BIOS 545 “Introduction to R programming” lab: R package

In this lab, we will practice installing and building your own R package, using Rstudio.

0 System setup

Install LaTeX:

- **On Windows**, download and install **MiKTex** (`http://miktex.org/`).
- **On Mac**, download and install **MacTex** (`http://www.tug.org/mactex/`, the full installation package is ridiculously 2.4Gb).
- **On Linux**, it’s preinstalled so you don’t need to do anything.

Install an editor for LaTeX(optional):

It’s completely fine to use a **simple text editor** to write latex files (NEVER use word processor software such as MS Word for that purpose). For example you can use **Rstudio**, **TextEdit** on Mac or **notepad** on Windows. However it’ll be easier to use a GUI. I recommend **TexShop** (`http://pages.uoregon.edu/koch/texshop/`) on Mac, and **WinEdt** (`http://www.winedt.com`) on PC. WinEdt is a commercial software and one needs to pay (with one month free trial). Google “latex editor windows” to find other options.

Moreover you need to identify a **simple text editor** to modify text files. Don’t use word processor software such as MS Word for that purpose. For example you can use Rstudio, **TextEdit** on Mac or **Notepad** on Windows.

1 Install a new package

Go to the CRAN contributed library page at `http://cran.r-project.org/web/packages/available_packages_by_name.html`. Browse the package list and you will get an idea of how comprehensive the R packages are.

Now pick one of the package and install it by different ways. For following instructions I’ll assume that you choose `abc`.

- **Install from Rstudio.**
  - **Tools** ⇒ **Install Packages...** Type “abc” in **Packages** and click **Install**.
- **Install from R console**.

  - Type `install.packages("abc")` in the Console. You probably will see that it installs a bunch of other packages. Those are the ones required by `abc`. Just wait patiently until it finishes.

- **Check that it works.**

  - Load package with `library("abc")`.
  - Run `?abc` to get a brief introduction to the package.

2 Create an R package

2.1 Create an R package without documentation

In this practice, we'll create an R package called **coolpkg** that contains a single function called **myRowSums**. Here is that function:

```
myRowSums=function(x) {
  n0 = nrow(x)
  result = rep(0, n0)
  for(i in 1:n0) {
    result[i]=sum(x[i,])
  }
  return(result)
}
```

1. Paste the **myRowSums** code into a new R script, and save it somewhere as **myRowSums.R**. The file name doesn't really matter. It doesn't have to be the same as the function name. If you wanted to include several functions, you could include them all in this one .R file.

2. Create package skeleton using Rstudio, by selecting **File** ⇒ **New Project** ⇒ **New Directory** ⇒ **R package**. Under **Package name**, type "coolpkg". Under **Create package based on source files**, click **Add...** and select **myRowSums.R**. Under **Create project as a subdirectory of:**, click **Browse...** and navigate to wherever you want your package files to be created. Click **Create Project**.

3. Open the **DESCRIPTION** file, and modify the necessary fields to put in your information as a developer.

4. Go to **Build** ⇒ **Build Source Package**. This will result in a "tarball" file named something like `coolpkg_0.1.0.tar.gz` being written to your current directory. This is the package source file that you can share with other people or make public by submitting to CRAN.

5. Install the package to R by going to **Tools** ⇒ **Install Packages...**, choosing **Install from:** Package Archive File, finding and selecting the tarball, and clicking **Install**.
6. Load in the package, e.g., by running `library(coolpkg)`, and test it by trying to use the **myRowSums** function that it contains.

So far the function is created without any document. In R, type `?myRowSums` and you'll see there's no function help yet.

2.2 Write R function help

Now we will create a help file for your **myRowSums** function. This will dictate what appears when a user runs `?myRowSums`.

1. In Rstudio, change the working directory to **man** inside **coolpkg**. With `coolpkg` loaded, run `prompt(myRowSums)`. This will create a file called **myRowSums.Rd**, which is the template of an R function help file.
2. Open **myRowSums.Rd** and fill in necessary fields. (To avoid an error on re-build, you have to at least change the **Title** field.)
3. Re-build, re-install, and re-load the package as described previously.
4. Now when you run `?myRowSums`, your help file should appear.

Let's also create a help file for the package itself, which will provide information when the user runs `?coolpkg`. The procedure is very similar to the previous one.

1. In Rstudio, the working directory should still be **man** inside **coolpkg**. Run `prompt(coolpkg)`. This will create a file called **coolpkg.Rd**.
2. Open **coolpkg.Rd** and fill in necessary fields. Again, at the very least, the **Title** field needs to be modified.
3. Re-build, re-install, and re-load the package as described previously.
4. A package help file will now appear when you run `?coolpkg`.

Congratulations. You know how to build a real R package now!

3 Optional: Simple latex

Follow the steps below to create a pdf file using LaTeX:

1. Create a file called `coolpkg.tex`.
2. Copy/paste the latex content in the lecture slide to the file.
3. Compile the latex file and obtain the pdf.

4 Create a package vignette

1. Install the **knitr** package in RStudio, then go to **Tools** ⇒ **Global Options...**. Click the **Sweave** icon on the left, and under **Weave Rnw files using** select **knitr**.
2. Go to **File** ⇒ **New File** ⇒ **R Sweave**. Copy the simple LaTeXexample from class notes into this file, and add the following lines to the Results section:

```
<<echo = TRUE, eval = TRUE>>=
myRowSums(matrix(rnorm(100), ncol = 5))
@
```

3. Save this file as **coolpkg.Rnw** (again, doesn't really matter what the name is), then click **Compile PDF**. RStudio should build and pop up a PDF file. This is what your package vignette will look like when users access it.
4. Create a new subdirectory alongside **man** and **R** called **vignettes**, and put a copy of **mypkg.Rnw** there.
5. Re-build, re-install, and re-load, and now when you run `vignette("coolpkg")` your vignette should pop up. Time permitting, you can add some content to your vignette to make it more descriptive, following instructions from the class notes.