BIOS 560R "Introduction to R programming" lab: R package I

This lab focuses on R library. We will practice installing and building R packages.

1 System setup

One needs to install necessary software and setup system environment in order to use command line to build R packages. Before the lab please set everything up so that we can focus on R packages.

Setup command line tools:

- **On a Mac or Linux:** The only thing you need to do is to find your terminal program. On Mac, it can be found by typing "terminal" in spotlight. On Linux it varies. But usually there's a Terminal menu.
- **On Windows PC:** This is complicated. You'll need to setup the System environment and install a bunch of tools. Follow the instructions at `http://www.biostat.wisc.edu/~kbroman/Rintro/Rwinpack.html` to have them setup.

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.

2 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 you favorite package and install it by different ways. For following instructions I'll assume that your favorite package is `abc`.

- **Install from R console**.
  1. Type `install.packages("abc")` within R. 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.
  2. After installation is finished, type `library(abc)` in R to make sure it was installed correctly. Then type `help(package=abc)` to get a brief introduction of the package.
- **Install from R GUI.** Follow the instruction on the class notes.

- **Bonus: install from command line.**

    1. First click the link to the package on CRAN, and download a binary version for your operating system. Assume the package binary file is `abc_1.6.tgz`. Save it to a directory, for example, `C:\Rpkg` (on Windows) or `/home/myname/` (on Mac/Linux).
    2. Open your command window, go to the directory where the package is save (you'll need to do something like "`cd C:\Rpkg`". Type `R CMD INSTALL abc_1.6.tgz` to install.

3 Create an R package

3.1 Create an R package without document

In this practice, we'll create an R package with a single function "`quartile.nist`" from your second homework. Assume the package is called "`quartile`".

1. Fist choose a directory for your package. Assume it's `C:\myRpkg`.
2. In R, change current directory to `C:\myRpkg`, then type `package.skeleton("quartile")`. This will create a new directory called `quartile`. Go into that directory, you'll see the structure of the package are in place.
3. Open DESCRIPTION file in your text editor, and modify the necessary fields.
4. Copy the R file containing your quartile.nist function to the `R` directory. Note the file must have extension `.R` or `.r`.
5. Now the package has been created. Go to your terminal, change to the package directory (e.g., `C:\myRpkg`) and check the package by running `R CMD check quartile`. Look at the package checking logs for error and warning messages.
6. Build the package source. Run `R CMD build quartile` in the terminal This will create a zip file on Windows and a gz file on Mac/Linux. This is your package source.
7. Try to install the package to R, using the R GUI or command line.
8. load in the package, e.g., by doing `library(quartile)`, and try your `quartile.nist` function.

In R, type `?quartile` and you'll see there's no function help yet. We'll add function help and package vignette next class.