Homework 10. Due by 5pm on Wednesday 11/11.

R packages

We have two goals for this class. One is to write a "hello world" function for an R package (`http://en.wikipedia.org/wiki/Hello_world_program`). Another is to discuss the role of R packages, and other similar software constructs, for developing and disseminating statistical research.

Please see how far you can get on the following. It would be helpful if you could send me a short email to say if you got all the way through or where you got stuck, or any other useful feedback. You're welcome to consult your peers. This assignment may be rather more time consuming (but also more instructive) if you have litle or no experience with Linux.

This homework guides you through the process of editing the `pomp` package, which just happens to be the one I know best. It is by no means a small package, but that may help us to think about not-so-small programming projects.

1. Log into a Linux machine. You can `ssh` into `bayes.stat.lsa.umich.edu`, which logs you onto one of the Statistics department bayes machines. Also, you can access a University Linux server at `scs.itd.umich.edu` via `ssh`.

   It is possible to carry out this assignment on Windows or Mac. As an optional extra task, you can try that as well. You will need to install the necessary source code compilers. For Windows, install Rtools from `https://cran.r-project.org/bin/windows/Rtools/`. For Mac, the C compiler is included in the Xcode toolset, available at `https://developer.apple.com/xcode/download/`. An R-compatible Mac Fortran compiler can be downloaded from `https://cran.r-project.org/bin/macosx/tools/`

2. Install the R package `pomp` using the R command

   ```
   install.packages("pomp")
   ```

   Depending on your exact setup, if you do not yet have a default location for personal R libraries to be installed, you may have to create a personal R library when prompted, as follows:

   ```
   Installing package into /usr/lib64/R/library
   (as lib is unspecified)
   Warning in install.packages("pomp") :
     'lib = "/usr/lib64/R/library"' is not writable
   Would you like to use a personal library instead?  (y/n) y
   Would you like to create a personal library
   ~/R/x86_64-redhat-linux-gnu-library/3.2
   to install packages into?  (y/n) y
   ```

Starting an R session and typing `require("pomp")` should load the package. This is the standard way to install and use an R package from CRAN (Comprehensive R Archive Network; `http://cran.us.r-project.org`). Typing `library(help=pomp)` gives some information about what is in the package, and more information is available in the reference manual and vignettes from `http://cran.r-project.org/web/packages/pomp/` though the functionality of this package is not relevant to us right now.

3. Installing the package as above does not give us access to the source code and does not let us edit the package. We will now see how to do these things. Create a directory (e.g., `mkdir ~/Rlib`). Download the `pomp` source code file `pomp_1.2.1.1.tar.gz` from `http://cran.r-project.org/web/packages/pomp`. Unzip the file, e.g.,

   ```
   tar xvfz pomp_1.2.1.1.tar.gz
   ```

   (type `man tar` to see more than you needed to know about tar files, or google "tar file format").

4. Poke around the resulting directory using commands like `cd`, `ls`, `more`.

5. In the directory `~/Rlib`, run

   ```
   R CMD INSTALL pomp
   ```

   This runs immediately for me, but dealing with related package dependencies may be specific to your particular R setup. Everything should work fine as long as you successfully completed Step 2.

6. Start an R session and load the newly installed version of the `pomp` package by

   ```
   require("pomp")
   ```

   You may have to deal with installing some other packages required by `pomp`. You could type `?pomp` to check that the package is loaded.

7. You are now in a position to edit the package. You need to choose a text editor (my preference is emacs), and so you could add code to define a `hello.world()` function by changing directory to `~/Rlib/pomp/R` and typing

   ```
   emacs hello.world.R &
   ```

   You can choose what your function should do!

8. You will have to include your new function in the `export` category of the `NAMESPACE` and the corresponding filename to the `collate` category of the `DESCRIPTION` file to make it accessible when you install the modified package.

9. Repeat Steps 5 and 6 on the edited package. Now you should be able to run

   ```
   hello.world()
   ```