SISBID 2017 Module 3 Homework

Keith Baggerly and Karl Broman

7/14/2017

Contents

General Background

The current US recommended dietary allowance (RDA) for vitamin D was set in 2011 at 600 international units (IU) per day, based on an overview report from the Institute of Medicine (IOM). The RDA is intended to be an intake sufficient to meet the dietary requirements of 97.5% of the population (mean + 2 standard deviations). "Meeting requirements" is typically assessed by relating some type of measureable physiologic property/response (e.g., bone density) to serum levels of the circulating form of vitamin D, 25(OH)D, where serum levels are measured in units of ng/mL (mostly US studies) or nmol/L (mostly European studies); 1 ng/mL = 2.5 nmol/L. After relating serum level to response, serum level is mapped as a function of intake based on data from studies (ideally randomized controlled trials, RCTs) showing how serum levels change in response to a fixed supplement dose.

We'll be looking at a few datasets involved in this process, which are posted on the GitHub site for SISBID Module 3. These include

- a dataset relating physiologic response to serum level (priemelDataReconstruction.csv).
- a dataset summarizing results from several clinical trials relating dietary intake levels in IU to serum levels in nmol/L, mostly transcribed from Table 5-4 in the IOM report (iomReview.csv). Careful with this one; it was assembled for human readability, and has blank lines.
- metadata for the dose response dataset (vitD.R).

The physiologic data come from a study by Priemel et al (2010). Direct measurements of bone are rare, because sampling bone hurts. In this study, bone samples from the ilaic crest (pelvis) were taken from 675 autopsy cases in Hamburg, and measurements were related to serum levels (because 25(OH)D doesn't degrade that quickly). What was measured about the bone was the fraction of a given volume of bone which was still osteoid (not yet mineralized and hardened). Generally, we want the ratio of osteoid volume (OV) to bone volume (BV) to be quite low, say 2% or less. OV/BV ratios much above this are associated with development of osteoporosis. The dataset here contains serum values in ng/mL and corresponding OV/BV values as percentages (i.e., the good/bad cutoff value is 2).

Using R Markdown (Day 1, Session 2)

Set up a folder to hold files for an initial data analysis. Add subfolders you think are appropriate.

Using a named chunk, plot the physiologic response (OV/BV) as a function of serum level.

Describe the trend.

Using logistic regression in a named code chunk, fit the probability that requirements are met (OV/BV is 2 or less) as a function of serum level. Does the answer change much if you restrict the set of patients to exclude those with serum values far from the target level (e.g., 10 nmol/L or less)? If so, which estimate would you trust more?

Describe the fit in text. State what the coefficient values are inline, rounded to 3 decimal places.

In a new code chunk, first invoke the earlier plot chunk to set things up, and then superimpose the regression fit. Use `abline` to add a horizontal line at your estimate of the serum level at which 97.5% of the people would have their requirements met.

Produce html, pdf, and word versions of your report.

Writing R Packages (Day 1, Sessions 3/4)

Using RStudio, open a new package project, giving it a name you find relevant (e.g., "sisbidVitD").

load the `devtools`, `roxygen2`, `rmarkdown`, and `knitr` libraries.

Edit the `DESCRIPTION` file.

Look at (but do not edit!) the `NAMESPACE` file.

Look at the contents of the `man/` and `R/` folders to see what's there.

In the `R/` folder, add a new R script describing the package, using `roxygen2` format.

Invoke `document()`. Check how the contents of the `man/` folder and the `NAMESPACE` and `DESCRIPTION` have changed.

Using the RStudio keyboard shortcut (`Cmd-Shift-B`), build and load your package. Check under the packages tab to confirm that your previously loaded packages (`devtools`, etc) are still loaded, and that your new package is listed and loaded.

Ask R about your package, using `?yourPackageNameHere`

Add a short function to your `R/` folder. Use `document()` to update the `man/` folder. Recompile your package and check the links.

Using the supplied R and csv files (`iomReview.csv` and `vitD.R`), add this table as a data object to your package. Document and recompile.

Using the above files as templates, document the second csv file (`priemelDataReconstruction.csv`) as a data object and add that to your package. Document and recompile.

Create a short vignette using `use_vignette()` in which you plot the physiologic response data as a function of serum level. Estimate the serum level at which you think 97.5% of people would have their requirements met (OV/BV values below 2). Invoke `build_vignette()` to produce an output file. Document and recompile.

Using `use_readme_md()`, add a `README` file to the top level of your package. Invoke `knitr` to produce a `.md` file from this. Document and recompile.

Using Git on Your Machine (Day 2, Session 1)

Create a git repository for either your R Markdown report or your package from yesterday (or both), using either RStudio or the command line.

Add and commit the files you deem relevant.

Create a `.gitignore` file and add files you don't want to track regularly (e.g., pdf, word, or html output files) to this. Add and commit the `.gitignore` file.

Add a new vignette in which you plot serum level achieved (`IOMLevel`) as a function of intake (`IOMIntake`). Add and commit this to your repository.

As it turns out, checking the initial source papers gives rise to slightly different numbers than those reported in the columns above; the revised values are in `AchievedLevel` and `TotalIntake`.

`checkout` the version of the repo from before you added the vignette.

Create a new branch, "alternate", starting from this version. Use `git status` to see which branch you're working on (also look at the git pane in RStudio). Add a different version of the same vignette as above, with the same name, differing only with respect to which columns of values are used. Add and commit this to the alternate branch.

Use `git merge` to fuse the two branches back together. This will cause a conflict, which `git` will ask you to resolve. Edit the vignette to choose your preferred version and include a note that the other version was tried as well, and comment on whether the differences (if any) looked meaningful.

Using Git with Your GitHub Account (Day 2, Session 2)

Create a new repository at GitHub for your R package.

Link the package folder on your computer to the GitHub repo using

```
git remote add origin https://github.com/username/packagename
```

```
git push -u origin master
```

(or do this using the RStudio links)

Confirm that the repo is now on GitHub

Add a new function to your package on your computer. Document and recompile. When it works, upgrade the version number slightly. Add and commit your changes.

Push the latest version of your repo to your GitHub repo. Confirm that it now shows the latest version.

Close the current project in RStudio, and open a new (empty) one. Clone a copy of your GitHub repo to this project.

Repeat the above process. This time we want to work on the earlier version of the repo. How can you get this from GitHub? Can you get this from what you *can* clone?

Collaborating with Others Using GitHub (Day 2, Sessions 3/4)

Get together with your neighbor (hi neighbor!)

Follow the instructions in Karl's homework posting on GitHub,

https://github.com/kbroman/Tools4RR/blob/master/05_Git_Lab/git_lab.md

Writing Templates (Day 3, Session 1)

Is there a common structure you want your reports to have (e.g., Introduction, Data and Methods, Results, and Conclusions)? Create a new package and include a template report in this form. Once the package is loaded, use RStudio to create a new file in the desired form from the template.

Is there a folder structure you want to use repeatedly? How might you automate this process? You may want to browse ProjectTemplate http://projecttemplate.net/getting_started.html or the keithUtils package at GitHub https://github.com/kabagg/keithUtils.

Note: more support for templates is being developed by RStudio https://rstudio.github.io/rstudio-extensions/rstudio_project_templates.html but this currently requires the daily builds, as opposed to the stable (production) builds.

What types of reports are commonly used in your group/lab/department/institution? What would be required to produce reports in that form?