Homework 12. Due by 11:59pm on Sunday 11/26.

A workflow for reproducible statistical research: combining Latex and R with knitr

There are numerous advantages to writing a statistics paper in such a way that the tables, figures and other quantitative results are automatically generated from chunks of code included in the document. These include:

1. Investigating the stability of conclusions. Questions like, "What happens to all my figures and tables if I omit the 5 smallest states from my panel of 50 states?" can be rapidly answered. The easier it is to investigate new analyses, the more things you explore.

2. Effective collaboration. All coauthors can read, run and modify all the code that produced the figures in the current version of a circulated draft.

3. Debugging. If your adviser asks "How exactly did this number get produced?" you can give a rapid, precise and accurate answer.

4. Updating. If you are presenting code (e.g., a lab presentation) and you want to make changes where necessary for a new software version, you simply re-run the document.

5. Revisions. 4 months after you submitted the paper, when the referee reports come back, you will be glad if you have your work organized in this way!

Rmarkdown and Jupyter are convenient platforms for exploratory investigations, but Rnw (a format associated with the R package knitr) is better placed to generate Latex for publication-quality pdf articles. This homework investigates a workflow, meaning a set of tools and procedures that together get research done effectively, used for the research project at `https://github.com/ionides/bagged_filters`. The code is somewhat complex and involves various features that may be new to you. You are welcome to ask your peers for help if you get stuck.

Edit the file `810f23/hw12.Rnw` with your answers, build a pdf and submit it to Canvas. To get started, clone the `bagged_filters` git repository to your laptop, as in Homework 9. There are two pdf files:

- `ms.pdf`, the main article.
- `si/si.pdf`, an online supplement for the article.

We will focus on ms.pdf.

1. The source file for ms.pdf is ms.Rnw, a file in R noweb format designed to be run using `knitr::knit("ms.Rnw")`, which requires the `knitr` package to be installed. Rnw files simply combine chunks of LaTeXwith chunks of R code.

   Rstudio runs `knit()` automatically when you ask it to build from an Rnw file, if you have set the relevant option:

   ```
   Tools -> Global Options -> Sweave -> Weave Rnw files using: knitr
   ```

   However, for workflows based on text commands it is good practice to call `knit()` directly from R.

   Have you used Rnw format before, either through Rstudio or not? Rnw has some similarities with the Rmarkdown (Rmd) format. Rmd is slightly simpler and quicker for some tasks, but Rnw is better for fine control of Latex.

   YOUR ANSWER HERE

2. There are various ways to compile ms.Rnw to ms.pdf. All of them need the necessary R packages, which you may need to install on your laptop or greatlakes or anywhere else you try running the code. Note that before installing `spatPomp` you will need to have `pomp` installed, for which you may need to consult the instructions at `https://kingaa.github.io/pomp/install.html`. The installation of `pomp` is nontrivial because this package carries out compilation of C code, so you need to have a C compiler installed and talking properly to R. Time spent figuring this out is not entirely wasted, since it demonstrates an approach to combining the computational efficiency of C with the statistical analysis environment of R.

   Now, in an R session running in the `bagged_filters` directory, you can run

   ```
   library(knitr)
   knit("ms.Rnw")
   ```

   If all is well, this will generate a file `ms.tex` which can be used to produce `ms.pdf` by running pdflatex. Likely, issues will arise that need to be solved to get this working. Spend a reasonable amount of time trying to get this working. Some debugging advice is posted on the class website at `hw12supp.html`. If you cannot get the code to run, the subsequent questions can be answered without this. Report on whether you were successful, what problems you overcame, and where you got stuck.

   YOUR ANSWER HERE.

3. Have you used `make` before? (`https://en.wikipedia.org/wiki/Make_(software)`) This is a standard tool for organizing scientific coding projects, and it is installed by default on Mac and Linux systems. The `bagged_filters` directory has a Makefile, so you can run

```
make ms.pdf
```

at a terminal prompt to build ms.pdf from ms.Rnw. This just runs `knit` followed by `pdflatex` so it cannot work unless the separate steps are working. For debugging, it can be better to run `knit` and `pdflatex` sequentially. Other things to experiment with if you are new to make:

```
make -n ms.pdf
make -B ms.pdf
make -nB ms.pdf
```

Try this, and report briefly

YOUR ANSWER HERE.

4. The manuscript can also be built on greatlakes, and this is appropriate for a production version having numerical calculations too extensive for a laptop. However, the version of `ms.Rnw` in the repository is set to run quickly, via the code `run_level <- 1`. This lets you run a preliminary version, for testing and debugging, fairly quickly on a laptop.

   Writing a reasonably large reproducible document combining text and code, you cannot avoid the issue of caching. You do not want to re-run all computations each time you edit any text in the document, so you must save (i.e., cache) results that do not need to be recomputed. Ideally, when we edit code we would re-run only the partial results that have changed as a consequence of the edit. Sadly, it is intractable to automate this in a foolproof way. The knitr code chunk option `cache=TRUE` re-runs a code chunk if that particular chunk is edited. Therefore, it may be necessary to delete all cached files (e.g., `rm -rf cache`) to force `knitr` to rebuild the cache correctly. In ms.Rnw, the `stew()` function from the `pomp` package is used to give additional manual control of caching the most time-consuming results. To remove all the results for `run_level=1` we can do

   ```
   rm -rf *_1
   ```

   Have you had any prior experience working with cache on reproducible documents?

   YOUR ANSWER HERE.

5. Workflows for writing manuscripts are built up over years, borrowed, shared, and modified for different purposes and evolving technologies. Compare this workflow with the range of techniques you already use.

   YOUR ANSWER HERE.

6. Notice how the random number generator seed is set to give reproducible results (e.g., search for "seed" in ms.Rnw). Subtle problems can arise when setting seeds for parallel computations. Can you think of any? This code attempts to deal with them via the `doRNG` R package.

   YOUR ANSWER HERE.