Stat243: Problem Set 1, Due Friday Sept. 11

August 31, 2015

Comments:

- This covers UNIX and the bash shell as well as practice with some of the tools we'll use in the course (VM, knitr/R Markdown).
- It's due at the start of class on Sept. 11.
- Please note my comments in the syllabus about when to ask for help and about working together.
- Finally, you should start early on this as the problems may take you some time. Particularly if you are getting used to bash, it may be hard to get everything to work at the last minute.

Formatting requirements

As discussed in the syllabus, please turn in (1) a copy on paper, as this makes it easier for us to handle AND (2) an electronic copy through Git following Harold's instructions.

Your electronic solution should be in the form of a plain text file named *ps1.txt*, with the bash shell code included. Ideally (but not required for this first problem set), your file would instead be a LaTeX+knitr (named *ps1.Rtex*) or R Markdown file (named *ps1.Rmd*), with bash code chunks included in the file. If you do choose to use LaTeX/knitr or R Markdown for the entire problem set, see the dynamic documents tutorial for how to create a PDF file using the *knitr* package in R.

For problems 1 and 2, your solution should start with a brief textual description of how you solved the problem, with the code following, including description of what your code does interspersed with the code. Do not just give us raw code.

Technical requirements for your solutions to Problems 1 and 2

All of your operations should be done using UNIX tools (i.e., you are not allowed to read the data into R or Python or other tools). Also, ALL of your work should be done using shell commands that you save in your solution file. So you can't say "I downloaded the data from such-and-such website" or "I unzipped the file"; you need to give us the code that we could run to repeat what you did. This is partly for practice in writing shell code and partly to enforce the idea that your work should be replicable and documented.

You will probably need to use *sed* in a basic way as we have used it so far in class and in the tutorial on bash. You should not need to use more advanced functionality nor should you need to use *awk*, but you may if you want to.

Problems

Hint: a simple piece of regular expression syntax standing for "any number of arbitrary characters" is ".*". To refer to an actual period or an actual quote, you need to 'escape' it: "\." and "\"".

1. This problem provides practice in downloading and manipulating data using shell scripting. We'll use United Nations Food and Agriculture Organization (FAO) data on agricultural production. If you go to http://data.un.org/Explorer.aspx?d=FAO and click on "Crops", you'll see a bunch of agricultural products with "View data" links. Click on "apricots" as an example and you'll see a "Download" button that allows you to download a CSV of the data. I've inspected the HTTP requests that the site handles and found out that you can download a file directly via URL in the following format:
`http://data.un.org/Handlers/DownloadHandler.ashx?DataFilter=itemCode:526&DataMartId=FAO&Format=csv&c=2,3,4,5,6,7&s=countryName:asc,elementCode:asc,year:desc`
That downloads the data for Item 526 (apricots). Note that you may need to put the http address inside double quotes when using a UNIX shell command to download it. Also make sure there are no carriage returns in the http address (I had to break the address above to fit on the page). You can see the item ID for other products by hovering over "View Data" link for the relevant product.

   (a) Download the data for apricots. Extract the data for individual countries into one file and for regions of the world into another file. Then subset the country-level data to the year 2005. Based on the "area harvested" determine the five countries using the most land to produce apricots. You'll need to look carefully at arguments to the *sort* utility, in particular the "key" will allow you to sort on a field that is not the first field. Now automate your analysis and examine the top five countries for 1965, 1975, 1985, 1995, and 2005. Have the rankings changed?

   (b) Write a bash function that takes as input a single item code (e.g., 526 for apricots, 572 for avocados) and prints out to the screen the data stored in the CSV file, such that the information could be piped on to UNIX another operation.

   (c) (Extra credit) Find the FAO metadata online that relate item codes to names of agricultural products. Modify your function in (b) so that the user can pass the name of the product instead of the code. Your function would then presumably query a file (i.e., a file that you have created in a separate step) that relates the codes to the names.

2. Your task here is to automatically download all the files ending in *.txt* from this National Climate Data Center website: http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/. Your shell script should provide a status message to the user, telling you the name of the file as it downloads each file. You should be able to use UNIX utilities to extract the individual file names from the HTML index file linked to above. Alternatively you may use tools from Unit 3, but an use of R should be done directly from the command line and should only involve a line or two of R code. Do not hard code the names of the .txt files into your code.

3. As preparation for future problem sets, this problem explores embedding R code and output in a PDF or HTML document.
Here is some R code that creates a plot and prints some output to the screen.

```
hist(LakeHuron)

lowHi <- c(which.min(LakeHuron), which.max(LakeHuron))
```

```
yearExtrema <- attributes(LakeHuron)$tsp[1]-1 + lowHi
```

Your task in this problem is to produce a single-page of PDF that looks like the last page of this assignment, using either the knitr package in R with LaTeX or R Markdown. If you are a Statistics student you should use LaTeX.

Requirements for your solution:

(a) Your solution should consist of the LaTeX+knitr or R Markdown syntax that produces the PDF, where the syntax embeds the necessary R code.

(b) Your resulting PDF output should look like the last page of the PDF of this assignment (it does not have to be exactly the same in terms of formatting and the actual prefacing text), which I created using LaTeX+knitr.

(c) Your resulting PDF document should be less than a page and your figure should be small enough that it only takes up half the width of the page (the *fig.width* argument may be helpful).

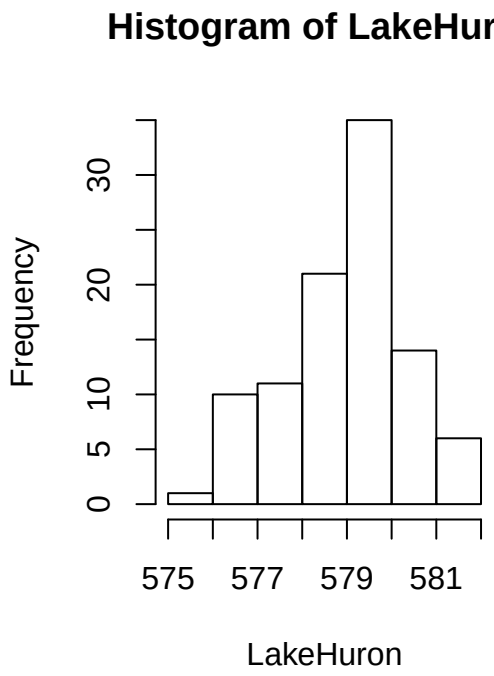
(d) In your solution, you should NOT manually type '1875' or '1972' in your document, rather embed an R expression that returns '1875' and '1972' using *rinline* or the equivalent for R Markdown.

The tutorial on dynamic documents provides information and example/template files that you can make use of to create your LaTeX/knitr or R Markdown file. Ask us questions if you get stuck - this question is just intended to ensure that you are up to speed on how to deal with formatting for future problem sets.

The result of your solution to Problem 3 should look like this page

The height of the water level in Lake Huron fluctuates over time. Here I 'analyze' the variation using R. I show a histogram of the lake levels for the period 1875 to 1972.

```
hist(LakeHuron)
```




```
lowHi <- c(which.min(LakeHuron), which.max(LakeHuron))
yearExtrema <- attributes(LakeHuron)$tsp[1]-1 + lowHi
```