Stat 243: Problem Set 2, Due Friday Sep-23

September 14, 2016

Instructions

Please turn in (1) a copy on paper, as this makes it easier for us to handle AND (2) an electronic copy through bCourses so we can run your code if needed.

Your electronic solution should be in the form of a plain text file, with the shell code included.

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.

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 a file and turn in as part of your solution to the assignment. 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. Any downloading and processing of data should be done by using bash commands as well.

Problems

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.

The goal here is to practice writing shell code to download files, extract fields and subset datasets, and summarize information in those fields.

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 file of the data.

Deconstructing the javascript on the site, it is possible to find 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&s=countryName:asc,elementCode:asc,year:desc& c=2,3,4,5,6,7&
```

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 (e.g. `curl`) 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 (e.g. 515 for Apples, 366 for Artichokes).

a. Download the data for apricots (i.e. the zip file).

b. Unzip the file. This will be the *raw data file*.

c. Extract the data for regions of the world into one file `regions.csv`. Examples of such regions are `"Africa +"` or `"Asia +"`. This file should NOT include the names of the columns.

d. Extract the data for individual countries into one file `countries.csv`. Do not include the last 7 rows of the raw data which do not contain information about countries. Also, this file should exclude the names of the columns.

e. Find out how many countries are in `countries.csv`. Tip: you can use the following code, which uses a Unix utility called **sed** (used for pattern matching and replacement) to deal with commas and quotes in the data file. I didn't go into sed in class because we'll deal with regular expressions primarily through R. The first line removes the commas in any country names so that commas are used only for field delimiters. The second line gets rid of the double quotes so that we can sort by fields numerically.

```
sed 's/, / /g' file1.csv > file2.csv
sed 's/\"//g' file2.csv > file3.csv
```

f. Subset the country-level data to the year 2005 into one file `countries2005.csv`.

g. Based on the `"Area Harvested"` determine the five countries in 2005 using the most land to produce apricots. You'll need to look carefully at arguments of the Unix **sort** utility, in particular the options that will allow you to sort on a field that is not the first field.

h. Now use a bash for-loop to automate your analysis and examine the top five countries for 1965, 1975, 1985, 1995, and 2005. Have the rankings changed?

i. Write a bash function `download_item()` 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 another UNIX operation.

More shell practice

2) Your task here is to automatically download all the files ending in `.txt` from this website:

http://textfiles.com/food/.

Your shell script should provide a status message to the user, telling you the name of the file as it downloads each file. In addition, write commands to answer the following questions:

- How many `.txt` files were downloaded?
- Which are the top-5 largest files (sorted by size)?
- What files contain numbers in their names (e.g. `1st_aid.txt`)?
- How many files do not contain numbers in their names?