Problem Set 1

Stats 506, F19

Due: Monday, September 23 at 5pm

Instructions

- Submit the assignment by the due date via canvas. Assignments may be submitted up to 72 hours late for a 5 point reduction.
- All files read, sourced, or referred to within scripts should be assumed to be in the same working directory (`./`).
- Your code should be clearly written and it should be possible to assess it by reading it. Use appropriate variable names and comments. Your style will be graded using the style rubric (./StyleRubric.html) [10 points].
- Some of these exercises may require you to use commands or techniques that were not covered in class or in the course notes. You can use the web as needed to identify appropriate approaches. Part of the purpose of these exercises is for you to learn to be resourceful and self sufficient. Questions are welcome at all times, but please make an attempt to locate relevant information yourself first.

Questions

Question 1 [25 points]

In this question you will demonstrate your understanding of important *Pmny| wl i ppwonppv* by writing short shell commands to work with the 2015 RECS data (https://www.eia.gov/consumption/residential/data/2015/index.php?view=microdata). When instructed to use a “one-liner”, utilize pipes “|” and file redirection to format your answer as a single string of commands. If needed, assume the commands are being written in a Bash shell with only the default functionality on the servers in the login.itd.umich.edu pool.

Submit your answer to this question as a single shell script named `ps1_q1.sh` ; also submit the same file with a txt extension `ps1_q1.txt` for viewing on Canvas. Use comments to clearly delineate each part. Also, be sure to include a descriptive header and “shebang” (`#!/` …).

a. [5 pts] Create a variable ‘file’ with the name of the csv file containing the RECS data. Check if this file exists in the local directory and, if not, download it.
b. [5 pts] Write a one-liner to extract the header row of the RECS data, translate the commas to new line characters, and write the results to a file ‘recs_names.txt’. [*Lmx>} syvvspyxsr wl sypb f i esri 1pni vf yx} sy q e} { nwl xs mngpyhi m} syvvgvnxe gshi f psgo xl exhi pixi w* `recs_names.txt` *njnxl i Ăpi epi eh} i |mww.*]

c. [10 pts] Write a one-liner that uses 'recs_names.txt' to find the column positions for the id and replicate weight columns in the RECS data and then re-formats these positions as a single, comma-separated string. [*Lmx>Jsvxl i Ăr epwi t 0ywi xl i 1wer h 1h st xmsr wxs xl i gsq q er h t ewi* .]

d. [5 pts] Store the result from the previous one-liner in the variable 'cols'. Use this variable to write a one-liner that extracts the id and replicate weight columns from the recs data and writes them to `recs_weights.csv` . [*Lmx>Mxl i Ăwx wi t 0ywi e gsr wxygxmsr wygl ew* `cols=$(...)` .]

Question 2 [15 points]

In this question you will extend your knowledge of the Linux shell by modifying your solution to question 1 to write a short command line program to be named `cutnames` . This program should extract those columns from a csv file having headers matching a regular expression.

Your command/script should accept the arguments "file" and "expression" by position. It should reproduce the output from question 1 if called as below:

```
bash ./cutnames.sh ./recs2015_public_v4.csv 'DOEID|^BRR' > recs_weights.csv
```

It is also acceptable if it works instead if called as below:

```
bash ./cutnames.sh ./recs2015_public_v4.csv 'DOEID\|^BRR' > recs_weights.csv
```

In either case, be sure to include comments explaining how the command line arguments are used by the script.

Name your script `cutnames.sh` and, as before, submit to Canvas as both `cutnames.sh` and `cutnames.txt` .

Challenge: Modify your script to recognize and work with gzip compressed files when the file name passed has extension '.gz'. ***Xl mwmwyr kvehi h erh wl syph rsxfi wyf q mxi h2***

Question 3 [30 points]

In this question you will write several R functions for working with "mouse-tracking" data as described in this manuscript (https://psyarxiv.com/zuvqa/). Briefly, suppose you have data in the form of a series of triples (x, y, t) representing the position (x, y) of a mouse cursor in the plane (i.e. monitor) at time t during a trial in which participants click one of two buttons in response to a prompt. The buttons are arranged horizontally on opposite sides of the screen to allow the experimenter to garner information about how decisively each response is given.

For each part, write an R function to accomplish the stated task. Name each function using an informative verb. Use comments to document the arguments and output of each function. Be sure to clearly state any assumptions on the inputs.

Submit your answers to parts a-e as a single executable R script `ps1_q3.R`. Also submit a pdf created using Rmarkdown with answers to all parts including your function definitions.

a. [5pts] Write a function that accepts a n $n \times 3$ matrix representing the trajectory (x, y, t) and *xverwqexi w*it to begin with time zero at the origin.

b. [5pts] Write a function that *gsqtyxiwxli erkpi* θ formed by the secant line connecting the origin and the final position in the trajectory. Your answer should be an angle between $[-\pi, \pi]$. Be sure your your solution works for a trajectory ending in any of the four quadrants.

c. [5pts] Write a function to *vsxexi* the (x, y) coordinates of a trajectory so that the final point lies along the positive x-axis.

d. [2pts] Combine the three parts above into a single function that *rsvqepmi w*an $n \times 3$ trajectory matrix to begin at the origin and end on the positive x-axis.

e. [8pts] Write a function that accepts a normalized trajectory and computes the following metrics describing its curvature:
   i. the total (Euclidean) distance traveled,
   ii. the maximum absolute deviation from the secant connecting the starting and final positions,
   iii. the average absolute deviation of the observed trajectory from the direct path,
   iv. the (absolute) area under the curve for the trajectory relative to the secant line using the trapezoidal rule to integrate. [Hint: Allow cancellation in “x” but not “y”.]

f. [5pts] Apply your function to the sample trajectories at the Stats506_F19 repo on GitHub and check your solutions against the sample measures. Then, compute the metrics above for the test trajectories and report your results in a nicely formatted table.