Digital Methods: Learning Journal

Christoffer Mondrup Kramer

Autumn 2020

Contents

1 02-10-2020: Start with R

For this assignment, please submit a page of your journal showing the following solutions:

1.1 Part 1: Use R to figure out how many elements in the vector below are greater than 2

```
rooms <- c(1, 2, 1, 3, 1, NA, 3, 1, 3, 2,
           1, NA, 1, 8, 3, 1, 4, NA, 1, 3,
           1, 2, 1, 7, 1, NA)
```

1.1.1 Steps

- **Action:** I created new project called "assignment_start_with_R". Then I created a folder called "scripts" for keeping track of my scripts. I also created a folder called "data" for keeping track of my data. Lastly, I copy-pasted the code above in to a script called "assignment1_2".

```
        rooms <- c(1, 2, 1, 3, 1, NA, 3, 1, 3,
           2, 1, NA, 1, 8, 3, 1, 4, NA, 1,
           3, 1, 2, 1, 7, 1, NA)
```

- **Action:** Then I created a variable, which would be a subvector of "rooms" but excluding all NA's.

```
        rooms_NA_removed <- rooms[!is.na(rooms)]
```

  "rooms_NA_removed" is my new variable. rooms[!is.na(rooms)] shows that I want a subsection of rooms but without NA's. This is done by using the operator ! which means NOT. If I removed the ! I would only get NA's.

- **Result:** It worked, I now had a new vector without any NA's.

- **Action:** I then used the logical operator "larger than" to only get a vector containing numbers larger than 2.

```
        rooms_NA_removed[rooms_NA_removed > 2]
```

  This worked since I got the following output:

```
        3 3 3 8 3 4 3 7
```

  Therefore, I just needed to count the length of this output.

- **Action:** I, therefore, used the function length() with my previous command as the parameter. This should count the length of the list.

```
length(rooms_NA_removed[rooms_NA_removed > 2])
```

- **Result:** This worked. The result is 8.

1.2 Part 2: What is the result of running median() function on the above 'rooms' vector? (again, best remove the NAs)

- **Action:** Since I already had the variable "rooms_NA_removed" without NA's I could just use the median() function with this variable as the parameter:

```
median(rooms_NA_removed)
```

- **Result:** This worked. The median of the rooms vector is 1.5.

1.3 Part 3

Inside your R Project (.Rproj), install the 'tidyverse' package and use the download.file() and read_csv() function to read the SAFI_clean.csv dataset into your R project as 'interviews' digital object. Take a screenshot of your RStudio interface showing a) the script you used to create the object b) the 'interviews' object in the Environment and the c) structure of your R project in the bottom right Files pane. Save the screenshot as an image and put it in your AUID_lastname_firstname repository inside our Github organisation (github.com/Digital-Methods-HASS). Place here the URL leading to the screenshot in your repository.

- **Click here for the screenshots of my script, the interview object, and my file structure**

1.4 Appendix

1.4.1 Source code for part 1 and 2

```
 #1) Use R to figure out how many elements in the vector below are greater than 2 .
#(You need to filter out the NAs first)

rooms <- c(1, 2, 1, 3, 1, NA, 3, 1, 3,
         2, 1, NA, 1, 8, 3, 1, 4, NA, 1,
         3, 1, 2, 1, 7, 1, NA)
```

```
#Create variable excluding NA from the vector
rooms_NA_removed <- rooms[!is.na(rooms)]

#Extract numbers over 2
rooms_NA_removed[rooms_NA_removed > 2]

#show the length of previous output
length(rooms_NA_removed[rooms_NA_removed > 2])

#What is the result of running median() function on the above 'rooms' vector?
#(again, best remove the NAs)
median(rooms_NA_removed)
```

2   25-09-2020: Shell

Your supervisor has shared a folder of photos on Sciencedata.dk with you (password is 2020CDS, folder is 500Mb and contains 189 images) and needs your help with a couple diagnostics:

2.1   Part 1: Identify the names and format of the 3 biggest files. Can you come up with a command to generate a numerically ordered list of 3 biggest files? (hint: consider using wc to gauge image size)

- **Answer:** The three biggest files, ordered numerically by size, are

    - "9240_Overview_S.RW2" (14.761.472 bytes)
    - "9247_Overview_SW.RW2" (14.733.312 bytes)
    - "9237_Overview_W.RW2" (14.713.856 bytes)

2.1.1   Steps

- **Action:** I started out by navigating to the HW folder, which is located in my data-shell folder on my desktop

    ```
    cd OneDrive/Skrivebord/data-shell/HW
    ```

    cd is the command for moving through the directories. Since my pc is backed up with OneDrive I need to move to my OneDrive folder first, then Dekstop (Skrivebord), then data-shell and lastly HW.

- **Action:** I then listed what was in the current directory, to see what files it contained:

```
ls
```

ls is the command for listing the contents of a directory. Since I'm currently in the HW directory, I do not need to specify the path.

- **Result:** There where a lot of .jpgs but also some other file-formats.

- **Action:** I needed to list the file-size. I knew that the wc command could do it, but I couldn't remember what flag to use, so I consulted the manual:

```
wc --help
```

- **Result:** I figured out, that I could use the -c flag to get the file-sizes.

- **Action:** I knew that I had to sort it numerically and in reverse. By doing this I could use the head command to get the first three lines:

```
wc -c * | sort -n -r | head -n3
```

The asterisk after -c indicates that I want to count the size of everything in this folder. With the second pipe I'm sorting the output from the previous command nummerically and in reverse (the largest numbers appear on top of the list). The last pipe takes the three first lines from the output of the second pipe.

- **Result:** It almost worked, however, the first line is the combined size of every file. I therefore needed to get the second, third and fourth line of the output, rather than the first, second and third line.

- **Action:** I changed head -n3 to head n-4 and added another pipe:

```
wc -c * | sort -n -r | head -n4 | tail -n3
```

The last pipe indicates that i want the 3 last lines from the previous output.

- **Action:** It worked, I now had a list of the three biggest files. Sorted numerically by size.

2.2 Part 2: Some of the image files are empty, a sign of error in the data processing or corruption. Can you find the empty photo files (0 kb size), count them, and generate a list of their filenames to make their later replacement easier?

- **Action:** I knew that I needed to somehow find files according to how big they were. I, therefore, thought that the command "find" would be the best place to start. Since, If I used the answer from part 1, I would have to count zero sized files manually. I used –help to see what flags I could use with find:

  ```
  find --help
  ```

- **Result:** It turns out that I can use the flag -size to find files according to their size.

- **Action:** I used find -size 0 to find all file containing zero bytes. Moreover, I also used the flag -type f to indicate, that I only wanted files (not directories). The last part wasn't that important, but it would make sure, that I only matched files.

  ```
  find -size 0 -type f
  ```

- **Result:** It worked. Now I only needed to count the amount of files. I noticed that all files was separated by line-breaks. Therefore, I could just use a pipe containing wc -l to count the number of lines from the previous output:

  ```
  find -size 0 -type f | wc -l
  ```

- **Result:** It worked. I had 73 files with 0 bytes. Now I only needed a list for later replacement.

- **Action:** Luckily this is easy. I can just use the "larger than" character to make a text-file containing the output:

  ```
  find -size 0 -type f > list_zero_byte_files.txt
  ```

  Result: It worked. I now have a file containing all file names with zero bytes called list_zero_byte_files.txt in the HW folder.

2.3 Appendix

2.3.1 Source code for part 1 (Copied from my terminal)

```
484  cd OneDrive/Skrivebord/data-shell/HW/
485  ls
486  wc --help
487  wc -c * | sort -n -r | head -n3
488  wc -c * | sort -n -r | head -n4 | tail -n3
489 history
```

2.3.2 Source code for part 2 (Copied from my terminal)

```
500  find --help
501  find -size 0 -type f
502  find -size 0 -type f | wc -l
503  find -size 0 -type f > list_zero_byte_files.txt
504  history
```

3   18-09-2020

3.1   Question 1: What are basic principles for using spreadsheets for good data organisation?

- Use consistent names, values and data types (Broman & Woo, , 2).

- Be careful with dates, since they are treated as numbers in excel. Data carpentry recommends splitting up day, month and year by different columns (Carpentry,).

- All names should be short but meaning-full. White-spaces should be replaced with either camel-Case or underscores (Broman & Woo, , 3).

- No cells should be left intentionally blank (ibid., 4). If a cell needs to be blank use either NA, na or BLANK as values (Carpentry,). This ensures that cells, which are actually missing data, can be properly dealt with.

- Never alter or use calculations in the raw/original data-set (Broman & Woo, , 7).

- Never use formatting as data. (Broman & Woo, , 7-8).

- Make back-ups, preferably with version control systems such as git(Broman & Woo, , 9).

- Keep track of changes (Wilson et al., , 12) Open-Refine is a great tool for this.

- Use Data validation to avoid errors (Broman & Woo, , 10).

- Have rich and usefull metadata, such as a Data dictionary (Broman & Woo, , 6).

- Never have more than one value in a cell (Broman & Woo, , 4).

- Save the data-set in a non-proprietary format such as .csv (Broman & Woo, ,).

- Every column should be a variable, every row should be an observation and every table should be one observational unit (Wickham, , 4).

3.2 Question 2: Does OpenRefine alter the raw data during sorting and filtering?

As far as I know, this is not the case. By looking at the "undo/redo" section it does not save sorting and filtering steps. Moreover, by exporting the data file and looking at it, sorting and filtering does not appear in the "history" section of the file.

3.3 Question 3: Fix the interviews data-set in OpenRefine enough to answer this question: "Which two months are reported as the most water-deprived/driest by the interviewed farmer households?"

October appears to be the most water-deprived month with 74 mentions, closely followed by September with 70 mentions.

3.4 Question 4: Describe briefly the steps you took to achieve the answer to point 3

- **Action:** I started by moving the column to the start of the tabel. That way it would be easier to find the column:

```
"op": "core/column-reorder",
"columnNames": [
  "months_no_water",
  "interview_date",
  "quest_no"
  "description": "Reorder columns"
```

"months no water" was moved in front of the column "interview date". (I have removed most of the columns from this code, in order to improve readability).

- I then removed the brackets, the white-spaces and the quotation marks with the following GREL function:

    value.replace("String pattern", "Replacement string")

  This resulted in the following GREL code:

    value.replace("[", "").replace("'", "").replace(" ", "").replace("]", "")

  This code finds to the opening bracket and replaces them with nothing. It then matches the quotation marks and replaces them with nothing. Then it matches the white-spaces and replaces them with nothing. Lastly, it finds the closing brackets and replaces them with nothing.

- **Result:** It worked like a charm. Now each month was separated by a semicolon.

- **Action:** I then created a custom text-facet, I used the following GREL code:

    value.split(";")

  This code matches each word separated by a semicolon as individual values, rather than the whole cell as one value.

- **Result:** It worked! now each month was displayed as a single value.

3.5   Final thoughts

It actually went quite well. I did not experience any problems. Maybe this data-science thing is not completely impossible.

3.6   Appendix: script

My script copied directly from OpenRefine:

```
  [
{
  "op": "core/column-reorder",
  "columnNames": [
    "months_no_water",
    "interview_date",
    "quest_no",
    "start",
    "end",
```

```
"province",
"district",
"ward",
"village",
"years_farm",
"agr_assoc",
"no_membrs",
"_members_count",
"remittance_money",
"years_liv",
"parents_liv",
"sp_parents_liv",
"grand_liv",
"sp_grand_liv",
"respondent_roof_type",
"respondent_wall_type",
"respondent_wall_type_other",
"respondent_floor_type",
"window_type",
"buildings_in_compound",
"rooms",
"other_buildings",
"no_plots",
"plots_count",
"water_use",
"no_group_count",
"yes_group_count",
"no_enough_water",
"period_use",
"exper_other",
"other_meth",
"res_change",
"memb_assoc",
"resp_assoc",
"fees_water",
"affect_conflicts",
"need_money",
"money_source",
"money_source_other",
"crops_contr",
"emply_lab",
"du_labour",
"liv_owned",
"liv_owned_other",
"liv_count",
"poultry",
```

```
          "du_look_aftr_cows",
          "items_owned",
          "items_owned_other",
          "no_meals",
          "months_lack_food",
          "no_food_mitigation",
          "gps_Latitude",
          "gps_Longitude",
          "gps_Altitude",
          "gps_Accuracy",
          "instanceID"
        ],
        "description": "Reorder columns"
      },
      {
        "op": "core/text-transform",
        "engineConfig": {
          "facets": [],
          "mode": "row-based"
        },
        "columnName": "months_no_water",
        "expression": "grel:value.replace(\"[\", \"\").replace(\"'\", \"\").replace(\" \", \"\")
        "onError": "keep-original",
        "repeat": false,
        "repeatCount": 10,
        "description": "Text transform on cells in column months_no_water using expression grel:
      },
      {
        "op": "core/text-transform",
        "engineConfig": {
          "facets": [],
          "mode": "row-based"
        },
        "columnName": "months_no_water",
        "expression": "grel:value.split(\";\")",
        "onError": "keep-original",
        "repeat": false,
        "repeatCount": 10,
        "description": "Text transform on cells in column months_no_water using expression grel:
      }
    ]
```

4   11-09-2020

4.1   Regex (Regular expressions): Stop-word list from Voyant! to R

9.30: I'm supposed to make a list with stop-words for Voyant! into a list for R. The R list contains words which are enclosed in quotation marks and separated by a comma and a white-space. For example: "an", "og". In the Voyant! list, all words are separated by a line-break and nothing else such as:
Han
Hun
Den

4.2   Actions and Results

- **Action:** I started out by trying to just match the words. I did this with the following expression:

  `^\w+`

  The circumflex indicates that the following expression should appear first in the string. That way I should not match any line-breaks. The w is used to match any word letter (including 0-9 and underscore). The + indicates that these character should appear one or more times.

- **Result:** This worked partially. However, it didn't match words containing æøå.

- **Action:** I, therefore, needed to account for Danish letters (æ-ø), which I did with the following expression:

  `^[\wæøå]+`

  The square brackets was added to indicate, that I wanted ANY letter inside them. The æøå were added so I could match danish letters. The + was moved outside of the square brackets to indicate, that any of the letters inside of the brackets should appear at least one time.

- **Result:** I thought that this made it possible to match all words in the list. But, it did not take case into account. It did not appear to be a problem in this particular list since all words were written in non-capital letters. However, if another list contained capital letters, it would create problems.

- **Action:** I therefore decided to make the expression case-insensitive. The list did not contain capital letters, so I wrote two test words ("Ælling" and "legoLand") and entered the following regex:

```
^(?i)[\wæøå]+
```

The parenthesis, the ? and the i was something Vojtech taught us yesterday. It makes the rest of the regex case-insensitive.

- **Result:** Ælling was not being matched. I suspected, that this was because of the letter æ.

- **Action:** I read about w and realised that it also includes capital letters from A-Z. Therefore, the problem had to be, that æøå was not being parsed as letters, but as special characters. I, therefore, entered the following regex:

```
^[\wæøåÆØÅ]+
```

I removed the code for case-insensitive, since the w took care of it for the letters A-Z. I then used ÆØÅ and æøå in the regex. This should ensure, that capital ÆØÅ would be matched as well.

- **Result:** While scrolling through the results, I noticed that certain characters, such as apostrophes and periods, were not being matched.

- **Action:** I needed to account for more special characters, so I added a period to the regex inside of the brackets. This should ensure, that I would also match special characters except for white-spaces and line-breaks:

```
^[\wæøåÆØÅ.]+
```

- **Result:** This did not work. Nothing changed.

- **Action:** I read about the period, and apparently it will literally only match periods, when it is in square brackets. So I moved it outside the brackets.

```
^[\wæøåÆØÅ].+
```

- **Result:** It actually kinda worked. But now the regex were not matching digits:

- **Action:** I realised that the + needs to be after the square brackets, and that the period needs to be optional. If the + is after the period it means, that the pattern needs to contain one or more letter characters (A-Z, a-z, 0-9 and underscore) or ÆØÅ or æøå AND one or more of any character except for white-space and line-breaks. In short, it meant that all matches needed to contain at least two characters. That is why I were not matching digits with one character. Therefore, the period needed to be optional. I, therefore, typed in:

```
^[\wæøåÆØÅ]+.?
```

The ? indicates that any character after the brackets is optional.

- **Result:** It did not work. I were only matching the first special character in these words.

- **Action**: I read about the ? and apparently it means between zero and one time. I, therefore, needed to use the asterisk:

  `^[\wæøåÆØÅ]+.*`

  The asterisk means that any character after the square brackets will be matched if it is used between zero and unlimited times (effectively it means that these characters are optional).

- **Result:** It worked. But maybe I should include it at the begging as well. That way I can match words which starts with a special character.

- **Action**: I added .* after the circumflex.

  `^.*[\wæøåÆØÅ]+.*`

  I tested this expression by writing YOLO with a hashtag in front of it.

- **Result:** It worked, I got a match.

- **Action:** Now I needed to substitute the line-breaks. Therefore, I made a group, so I could create variables:

  `(^.*[\wæøåÆØÅ]+.*)`

  The parenthesis indicates, that this is a group, which can be used later. I tried to put quotation marks around the variable and follow it with a comma and a white-space:

  `"$1",`

- **Result:** This worked partially, but it did not remove the line-break:

- **Action:** I tried to use "b" to mark word boundaries, since I suspected that I was matching line-breaks:

  `(\b.*[\wæøåÆØÅ]+.*\b)`

  The b indicates a word boundary (such as a white-space or a line-break):

- **Result:** This did not change anything, and I were not matching words which started with ÆØÅ. So I removed the boundary.

- **Action:** I then tried to match and group line-breaks separately. However the line-break needed to be optional. Otherwise it would not match the last word on the list, which was not followed by a line-break:

  `(^.*[\wæøåÆØÅ]+.*)(\r*)`

- **Result:** This worked. It is now a stop-word list for R. It means the following: Find a pattern where THE FIRST CHARACTERS, WHICH CAN BE ANYTHING EXCEPT FOR WHITE-SPACES AND LINE-BREAKS, APPEARS BETWEEN ZERO AND UNLIMITED TIMES. Followed by ONE OR MORE WORD LETTERS FROM A-Z, a-z, 0-9, ÆØÅ, æøå. Followed by ZERO OR MORE CHARACTERS, WHICH CAN BE ANYTHING EXCEPT FOR WHITE-SPACES OR LINE-BREAKS. Followed by ZERO OR MORE LINE-BREAKS.
  However, after writing this explanation, I realised that I could just use the following code, since my code is way too over-engineered:

  `(.+)(\r*)`

  The period captures all characters regardless of their capitalization including æøå, digits and special characters. It just means the following: Find a pattern WHICH CONTAINS ONE OR MORE CHARACTERS EXCEPT FOR WHITE-SPACES AND LINE-BREAKS followed by ZERO OR MORE LINE-BREAKS.

4.3 Final thoughts

It was actually quite fun, but also very frustrating, to work with regexes. Moreover, I realised, that it is very helpfull to write exactly what the regex is matching, since it made me realize, that there was a way more elegant solution.

5 11-09-2020

5.1 Regex (Regular expressions): Stop-word list from R to Voyant!

13.30: Same task as before but in reverse.

5.2 Actions and Results

- **Action:** I started with my previous and over-engineered regex:

  `(^.*[\wæøåÆØÅ]+.*)(\r)`

- **Result:** This did not work. The whole list was displayed as one match rather than multiple matches.

- **Action:** I tried to but a word boundary around the regex

  `(\b.*[\wæøåÆØÅ]+.*\b)(\r)`

- **Result:** No changes.

- **Action:** I suspected that this was because of the periods in the regex. I, therefore, removed them.

  `(\b[\wæøåÆØÅ]+\b)(\r)`

- **Result:** Now I was not matching words with special characters such as umlauts or periods. But I could not use the period since it would match the quotation marks and commas. Moreover, I could not specify each special character, since I did not know which where used.

- **Action:** After a break I realized that I should be trying to match the commas, the spaces and the quotation marks, since they are the ones I need to replace. So I made the following expression:

  `(", )`

  This regex matches a string which contains a quotation mark followed by a comma and then a white-space. This is how all stop-words on the R list are separated.

- **Result:** This almost worked, however, the quotation mark at the start of each word was not captured.

- **Action:** I, therefore, added another quotation mark after the white-space, that way I should be able to match the commas at the start of each word as well:

  `(", ")`

- **Result**: The first and last quotation mark in the list were not being matched. Moreover, a few words did not fit this pattern (probably an entry mistake).

- **Action:** I made different groups to match sequences, that did not fit the standard pattern (entry mistakes).

  `(", ")|(")|(,)|( )`

  The vertical line is a logic operator, which means OR. So this regex should catch either the quotation – comma – white-space sequence OR quotation marks OR commas OR white-spaces that do not follow this sequence (effectively all signs that are misplaced).

- **Result**: This worked, but left a line-break at the beginning and at the end of the list and in places which did not follow the sequence of quotation – comma – space.

5.3 Final Thoughts

18.00: I tried for hours to find a way to removed the extra line-breaks, but I could not. So for now, I will leave it as it is.

6 13-09-2020

6.1 Regex (Regular expressions): Stop-word list from R to Voyant! - Part 2

12.00: I will proofread my assignment and submit it. I do not think, that I can crack the last part of the assignment.

13.00: While proofreading my assignment, I suddenly realised the solution. I could just use a circumflex inside the brackets to indicate, which patterns I do not want to match.

6.2 Actions and Results

- **Action:** I started by specifying that, I do not want any patterns containing white-spaces, commas or periods.

  `([^" ,]+)`

  The + at the end ensures, that the characters are matched as a word, rather than individual letters.

- **Result:** It worked, now I am matching all words in the list without matching commas, white-spaces or quotation marks

- **Action:** Now I just need to match the commas, the white-spaces and the quotation marks. So I wrote the following regex:

  `([^" ,]+)([" ,]*)`

  This should match either a quotation mark, a white-space or a comma, if it is used between zero and unlimited times after the first group.

- **Result:** It almost worked. However, I'm not matching the first quotation mark on the list.

- **Action:** However, I can just set the regex to match zero or more quotation marks at the start of the list by adding a quotation mark and an asterisk at the start of the regex

  `("*)([^" ,]+)([" ,]*)`

  This should match the first quotation mark on the list.

- **Result:** It almost worked. There are two places, where there is a blank line-break. But that is probably because there are two line-breaks in the list. So I just need to match the line-breaks, as I did in the previous task.

- **Action:** I added two groups for matching line-breaks in the regex. One at the beginning, and one at the end. A line-break at the end should be enough. But, by having one at the beginning as well, this regex should catch all line-breaks. Lastly, I made these line-breaks optional using the asterisk.

    ```
    (\r*)("*)([^" ,]+)([" ,]*)(\r*)
    ```

- **Result:** It worked! This regex means the following: Find a pattern that starts with ZERO OR MORE LINE-BREAKS followed by ZERO OR MORE QUOTATION MARKS and ONE OR MORE CHARACTERS THAT DOES NOT INCLUDE A QUOTATION MARK, A COMMA OR A WHITE-SPACE followed by ZERO OR MORE QUOTATION MARKS, COMMAS OR WHITE-SPACES followed by ZERO OR MORE LINE-BREAKS.

6.3   Final Thoughts

I learned that sometimes it helps to just take a step back from the project and return later to review it. It was while re-reading my journal, that I discovered how to solve this problem.

References

Broman, K. W., & Woo, K. H. (2018). Data Organization in Spreadsheets. *The American Statistician*, *72*(1), 2–10. doi: 10.1080/00031305.2017.1375989

Carpentry, D. (n.d.). *Data organization in spreadsheets for social scientists.* Retrieved from `https://datacarpentry.org/spreadsheets-socialsci/`

Wickham, H. (2014). Tidy Data. *Journal of Statistical Software*, *59*(10), 1–22.

Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L., & Teal, T. K. (2017). Good enough practices in scientific computing. *PLoS Computational Biology*, *13*(6), 1–20. doi: https://doi.org/10.1371/journal.pcbi .1005510