Pollution assignment

Oleksii Yehorchenkov

30 11 2020

This assignment is based on materials from Coursera course R Programming

Introduction

For this assignment you will write three functions that are meant to interact with dataset that accompanies this assignment. The dataset is contained in a zip file specdata.zip

Data

The zip file contains 332 comma-separated-value (CSV) files containing pollution monitoring data for fine particulate matter (PM) air pollution at 332 locations in the United States. Each file contains data from a single monitor and the ID number for each monitor is contained in the file name. For example, data for monitor 200 is contained in the file "200.csv". Each file contains three variables:

- Date: the date of the observation in YYYY-MM-DD format (year-month-day)
- sulfate: the level of sulfate PM in the air on that date (measured in micrograms per cubic meter)
- nitrate: the level of nitrate PM in the air on that date (measured in micrograms per cubic meter)

For this programming assignment you will need to unzip this file and create the directory 'specdata'. Once you have unzipped the zip file, do not make any modifications to the files in the 'specdata' directory. In each file you'll notice that there are many days where either sulfate or nitrate (or both) are missing (coded as NA). This is common with air pollution monitoring data in the United States.

Part 1

Write a function named 'pollutantmean' that calculates the mean of a pollutant (sulfate or nitrate) across a specified list of monitors. The function 'pollutantmean' takes three arguments: 'directory', 'pollutant', and 'id'. Given a vector monitor ID numbers, 'pollutantmean' reads that monitors' particulate matter data from the directory specified in the 'directory' argument and returns the mean of the pollutant across all of the monitors, ignoring any missing values coded as NA. A prototype of the function is as follows

```
pollutantmean <- function(directory, pollutant, id = 1:332) {

    file_list <- list.files(path=directory, pattern=".csv", full.names = TRUE)
    pollutant_values <- numeric()

    for(i in id){
        file_df <- read.csv(file_list[i])
```

```
        pollutant_values <- c(pollutant_values, file_df[[pollutant]])

    }

    mean(pollutant_values, na.rm=TRUE)


}
```

Output examples:

```
pollutantmean("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", "sulfate", 1:10)
```

```
## [1] 4.064128
```

```
## [1] 4.064128
```

```
pollutantmean("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", "sulfate", 55)
```

```
## [1] 3.587319
```

```
## [1] 3.587319
```

```
pollutantmean("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", "nitrate")
```

```
## [1] 1.702932
```

```
## [1] 1.702932
```

Part 2

Write a function that reads a directory full of files and reports the number of completely observed cases in each data file. The function should return a data frame where the first column is the name of the file and the second column is the number of complete cases. A prototype of this function follows

```
complete <- function(directory, id = 1:332) {
    # 'directory' is character vector of length 1
    # indicating the location of the csv file.

    # 'id' is an integer vector indicating the monitor ID
    # numbers to be used

    # Returns a data frame of the form:
    # id    nobs
    # 1     117
    # 2     1041
    # ...
    # where 'id' is the monitor number and 'nobs' is the
    # number of complete cases
```

```
    file_list <- list.files(path = directory, pattern = ".csv", full.names = TRUE)
    nobs <- numeric()
    for(i in id){
        file_df <- read.csv(file_list[i])
        nobs <- c(nobs, sum(complete.cases(file_df)))
    }
    data.frame(id, nobs)

}
```

Output examples:

```
complete("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", 1)
```

```
##   id nobs
## 1  1  117
```

```
##   id nobs
## 1  1  117
```

```
complete("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", c(2, 4, 8, 10, 12))
```

```
##   id nobs
## 1  2 1041
## 2  4  474
## 3  8  192
## 4 10  148
## 5 12   96
```

```
##   id nobs
## 1  2 1041
## 2  4  474
## 3  8  192
## 4 10  148
## 5 12   96
```

```
complete("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", 50:60)
```

```
##    id nobs
## 1  50  459
## 2  51  193
## 3  52  812
## 4  53  342
## 5  54  219
## 6  55  372
## 7  56  642
## 8  57  452
## 9  58  391
## 10 59  445
## 11 60  448
```

```
##   id nobs
## 1  50  459
## 2  51  193
## 3  52  812
## 4  53  342
## 5  54  219
## 6  55  372
## 7  56  642
## 8  57  452
## 9  58  391
## 10 59  445
## 11 60  448
```

Part 3

Write a function that takes a directory of data files and a threshold for complete cases and calculates the correlation between sulfate and nitrate for monitor locations where the number of completely observed cases (on all variables) is greater than the threshold. The function should return a vector of correlations for the monitors that meet the threshold requirement. If no monitors meet the threshold requirement, then the function should return a numeric vector of length 0. A prototype of this function follows

```
corr <- function(directory, threshold = 0) {
  # 'directory' is character vector of length 1
  # indicating the location of the csv file.

  # 'threshold' is numeric vector of length 1 indicating
  # the number of completely observed observations (on all
  # variables) required to compute the correlation between
  # nitrate and sulfate; the default is 0

  # Return a numeric vector of correlations
  # NOTE: do not round the result!

  file_list <- list.files(path = directory, pattern = ".csv", full.names = TRUE)
  directory_df <- complete(directory)
  id <- directory_df[directory_df["nobs"] > threshold, ]$id
  corr <- numeric()

  for(i in id){
    file_df <- read.csv(file_list[i])
    df <- file_df[complete.cases(file_df), ]
    corr <- c(corr, cor(df$sulfate, df$nitrate))
  }

  corr
}
```

For this function you will need to use the 'cor' function in R which calculates the correlation between two vectors. Please read the help page for this function via '?cor' and make sure that you know how to use it.

Output examples:

```
cr <- corr("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", 150)

head(cr); summary(cr)
```

```
## [1] -0.01895754 -0.14051254 -0.04389737 -0.06815956 -0.12350667 -0.07588814

##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.21057 -0.04999  0.09463  0.12525  0.26844  0.76313
```

```
## [1] -0.01895754 -0.14051254 -0.04389737 -0.06815956 -0.12350667 -0.07588814
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.21060 -0.04999  0.09463  0.12530  0.26840  0.76310
```

```
cr <- corr("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", 400)

head(cr); summary(cr)
```

```
## [1] -0.01895754 -0.04389737 -0.06815956 -0.07588814  0.76312884 -0.15782860

##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.17623 -0.03109  0.10021  0.13969  0.26849  0.76313
```

```
## [1] -0.01895754 -0.04389737 -0.06815956 -0.07588814  0.76312884 -0.15782860
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.17620 -0.03109  0.10020  0.13970  0.26850  0.76310
```

```
cr <- corr("C:/Users/Soumy/Documents/data/rprog_data_specdata/specdata", 5000)

head(cr); summary(cr) ; length(cr)
```

```
## numeric(0)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##

## [1] 0
```

```
## NULL
## Length  Class   Mode
##      0   NULL   NULL
## [1] 0
```

The function that you write should be able to approximately match this output. **Note that because of how R rounds and presents floating point numbers, the output you generate may differ slightly from the example output**.