Programming Assignment 1

Anastasiia Morozova

8/13/2021

Programming Assignment 1 – R Programming by Johns Hopkins University

For this first programming assignment I wrote three functions that are meant to interact with dataset that accompanies this assignment. The dataset is contained in a zip file specdata.zip (https://d396qusza40orc.cloudfront.net/rprog%2Fdata%2Fspecdata.zip)

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) 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.

We can preview what the binded file (of 332 csv files) would look like with a tidyverse solution, although it is not necessary

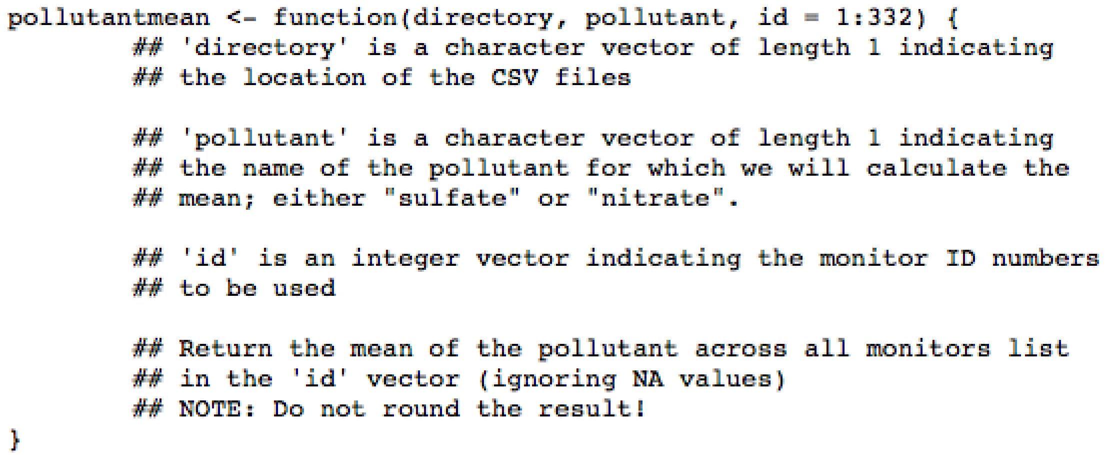
```
head(specdata_df)
```

```
## # A tibble: 6 × 4
##   Date       sulfate nitrate    ID
##   <date>       <dbl>   <dbl> <dbl>
## 1 2003-01-01      NA      NA     1
## 2 2003-01-02      NA      NA     1
## 3 2003-01-03      NA      NA     1
## 4 2003-01-04      NA      NA     1
## 5 2003-01-05      NA      NA     1
## 6 2003-01-06      NA      NA     1
```

Don’t forget to reverse the working directory back to the main one containing the specdata zip file before implementing the next set of functions.

Part 1

For Part 1, the task is to 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) {
        ## 'directory' is a character vector of length 1 indicating
        ## the location of the CSV files

        ## 'pollutant' is a character vector of length 1 indicating
        ## the name of the pollutant for which we will calculate the
        ## mean; either "sulfate" or "nitrate".

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

        ## Return the mean of the pollutant across all monitors list
        ## in the 'id' vector (ignoring NA values)
        ## NOTE: Do not round the result!
}
```

Prototype for “pollutantmean” function

The resulting function is:

```
pollutantmean <- function(directory, pollutant, id=1:332) {
  filePaths <- list.files((paste(getwd(), "/", directory, "/", sep = "")), "\\.csv$", fu
ll.names = TRUE)
  result <- do.call(rbind, lapply(filePaths, read.csv))
  sub_id <- NULL
  for (i in id) {sub_id <- rbind(result[result$ID == i,], sub_id)}
  mean(sub_id[[pollutant]], na.rm=TRUE)
}
```

Sample output is below:

```
pollutantmean("specdata", "sulfate", 1:10)
```

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

```
pollutantmean("specdata", "nitrate", 70:72)
```

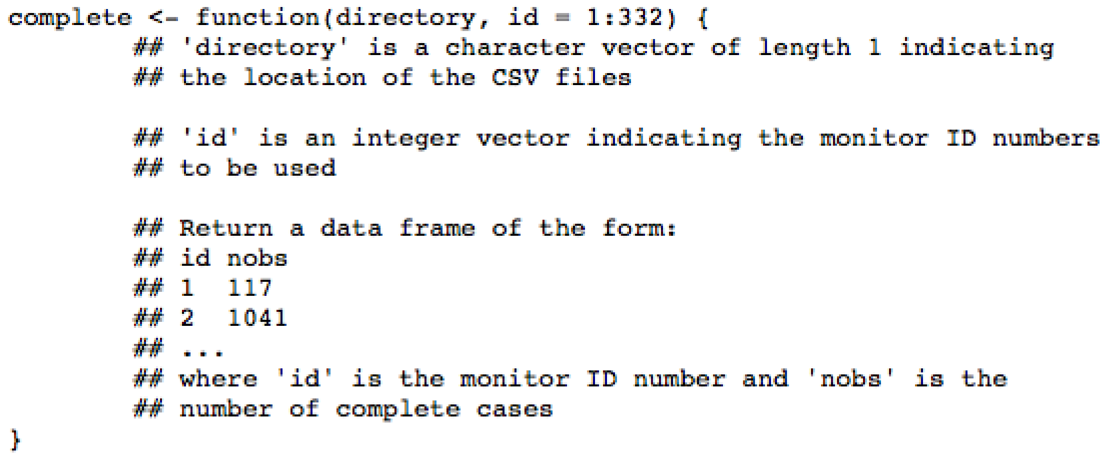
```
## [1] 1.706047
```

```
pollutantmean("specdata", "nitrate", 23)
```

```
## [1] 1.280833
```

Part 2

For part 2 the task is to 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 a character vector of length 1 indicating
        ## the location of the CSV files

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

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

Prototype for “complete” function

The resulting code as follows:

```
complete <- function(directory, id=1:332) {
 compl <- data.frame("id" = numeric(),
                     'nobs' =  numeric())
            for (i in id) {
                    path <- paste(getwd(), "/", directory, "/", sprintf("%03d", i), ".c
sv", sep = "")
                    monitor_data <- read.csv(path)
                    j<- which(id==i)
                     list_j <- c(i, sum(complete.cases(monitor_data)))
                      compl[j,] <- list_j
    }
  compl
}
```

And the sample output is below:

```
complete("specdata", 1)
```

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

```
complete("specdata", c(2, 4, 8, 10, 12))
```

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

```
complete("specdata", 30:25)
```

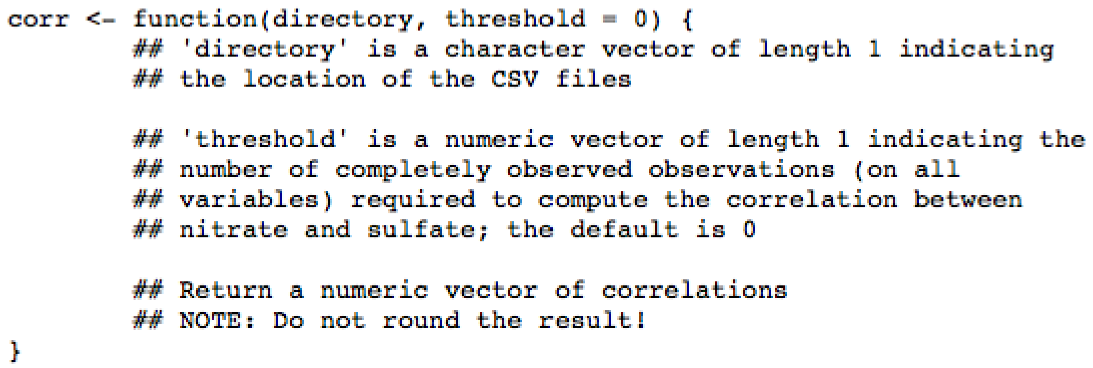
```
##   id nobs
## 1 30  932
## 2 29  711
## 3 28  475
## 4 27  338
## 5 26  586
## 6 25  463
```

```
complete("specdata", 3)
```

```
##   id nobs
## 1  3  243
```

Part 3

For the last part, the task is to 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 a character vector of length 1 indicating
        ## the location of the CSV files

        ## 'threshold' is a 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!
}
```

Prototype for “corr” function

The resulting function is as follows:

```
corr <- function(directory, threshold=0) {
  store <-numeric()
  for (i in 1:332) {
    path <- paste(getwd(), "/", directory, "/", sprintf("%03d", i), ".csv", sep = "")
    monitor_data <- read.csv(path)
    if ((sum(complete.cases(monitor_data)))>threshold)
    {store[(length(store) + 1)] <- (cor(monitor_data$nitrate, monitor_data$sulfate, use
 = "na.or.complete"))}
  }
  store
  }
```

And the sample output is below:

```
cr <- corr("specdata", 150)
head(cr)
```

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

```
summary(cr)
```

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

```
cr <- corr("specdata", 400)
head(cr)
```

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

```
summary(cr)
```

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

```
cr <- corr("specdata", 5000)
summary(cr)
```

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

```
length(cr)
```

```
## [1] 0
```

```
cr <- corr("specdata")
summary(cr)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -1.00000 -0.05282  0.10718  0.13684  0.27831  1.00000
```

```
length(cr)
```

```
## [1] 323
```