R Programming - Assignment #1

Myron Keith Gibert Jr

January 21, 2020

Contents

Correspondence

Please address any questions to Myron Keith Gibert Jr at mkgibertjr@msn.com. Code for this project is stored in a GitHub repository.

Introduction

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 that is included in the GitHub repository.

Set Parameters

```
#set output directory? Default: outputdir <- "assignment1outputs"
outputdir <- "assignment1outputs"
#Overwrite contents of the output directory? Default: deleteoutputs <- FALSE
deleteoutputs <- TRUE
#Delete specdata/ directory after completing the analysis? Default: deletespec <- TRUE
deletespec <- TRUE
```

Debug

The debug chunk will prevent the script from running if any of the dependent variables for this analysis do not exist. This should prevent the program from erroring out after a long runtime without producing any results due to a missing variable. If modifying the input .csv and .xlsx files, it is important to leave all header information and column names intact, as the program uses this information to extract relevant data. Columns are intuitively labeled to end user convenience.

```
if (dir.exists(outputdir) && deleteoutputs == FALSE ){
  stop("Your output directory already exists!  Please delete/move
       this folder from your working directory.  Alternatively, you
       can set 'deleteoutputs' to TRUE to auto-delete this folder
       for every run. You may also choose an alternative output
       directory.")
}else{
  unlink(outputdir,recursive = TRUE)
}

if (!exists("outputdir")){
stop("outputdir variable is not defined.  Please ensure that all
     parameters in the r parameters chunk are defined.")
}

if (!exists("deleteoutputs")){
stop("deleteoutputs variable is not defined.  Please ensure that all
     parameters in the r parameters chunk are defined.")
}

if (!exists("deletespec")){
stop("deletespec variable is not defined.  Please ensure that all
     parameters in the r parameters chunk are defined.")
}

if (!dir.exists(outputdir)){dir.create(outputdir)}
```

Data

The zip file containing the data can be downloaded here:

specdata.zip 2.4MB

I have renamed the file to "ASN1_rprog_data_specdata.zip" for organization.

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)

Unzipping the data

For this programming assignment I needed to unzip this file and create the directory 'specdata'. Once I unzipped the zip file, I did 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.

```
if(!dir.exists("specdata")){
  unzip("ASN1_rprog_data_specdata.zip")
}
```

Part 1

I first wrote 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. My final version of this function is as follows:

```
pollutantmean <- function(directory, pollutant, id = 1:332){
  #id <- 1:10
  #directory <- "./specdata"
  #pollutant <- "sulfate"
  filenames <- list.files(directory)
  data <- data.frame()

  i <- 1
  for(i in 1:length(id)){
    data <- rbind(data,read.csv(paste(directory,filenames[i],sep = "/")))
  }

  mean_pollutant <- mean(data[,pollutant],na.rm = TRUE)
  write.csv(mean_pollutant,file = paste(outputdir,"/",pollutant,"_mean_pollutant.csv",sep=""))
  mean_pollutant
}
```

```
# ExamplesS
pollutantmean("./specdata","sulfate",1:10)
```

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

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

```
## [1] 0.8599547
```

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

```
## [1] 0.5499098
```

Part 2

I then wrote 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. The final version of this function is as follows:

```
complete <- function(directory,id = 1:332){
  #i = 1
  #id <- 1
  #directory <- "./specdata"
  filenames <- list.files(directory)
  data <- data.frame()
  for(i in id){
    df <- read.csv(paste(directory,filenames[i],sep = "/"))
    rbinder <- data.frame(filenames[i],sum(complete.cases(df)))
    data <- rbind(data,rbinder)
  }
colnames(data) <- c("id","nobs")
write.csv(data,paste(outputdir,"completedcases.csv",sep="/"))
data
}

# Examples
complete("./specdata",1)
```

```
##        id nobs
## 1 001.csv  117
```

```
complete("./specdata",c(2,4,6,8,10,12))
```

```
##        id nobs
## 1 002.csv 1041
## 2 004.csv  474
## 3 006.csv  228
## 4 008.csv  192
## 5 010.csv  148
## 6 012.csv   96
```

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

```
##        id nobs
## 1 030.csv  932
## 2 029.csv  711
## 3 028.csv  475
```

```
## 4 027.csv  338
## 5 026.csv  586
## 6 025.csv  463
```

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

```
##        id nobs
## 1 003.csv  243
```

Part 3

Lastly, I wrote 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. The final version of this function is as follows:

```
corr <- function(directory,threshold = 0){
#directory <- "./specdata"
#threshold <- 150
#i=1
cor_results <- as.numeric(vector())
complete_cases <- complete(directory)
complete_cases <- complete_cases[complete_cases$nobs>=threshold, ]

if(nrow(complete_cases)>0){
    for(caseid in complete_cases$id){
          path <- paste(directory,caseid, sep = "/")
          data <- read.csv(path)
          narm.data <- data[(!is.na(data$sulfate)), ]
          narm.data <- narm.data[(!is.na(narm.data$nitrate)), ]
          cor_results <- c(cor_results, cor(narm.data$nitrate, narm.data$sulfate))
        }
    }
    cor_results
}

# Examples
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 -0.05147  0.09333  0.12401  0.26836  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.     NA's
## -1.00000 -0.05282  0.10719  0.13684  0.27831  1.00000        9
```

```
length(cr)
```

```
## [1] 332
```

Quiz

After completing the programming assignment, I had to complete a quiz using the provided code with each question. I first had to run the provided code, and then select my output from the multiple choice options. This quiz was used to verify whether my three functions can effectively answer the data science questions from this data set. My final score was a 100%.

1. What value is returned by the following call to pollutantmean()? You should round your output to 3 digits.

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

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

2. What value is returned by the following call to pollutantmean()? You should round your output to 3 digits.

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

```
## [1] 0.8599547
```

3. What value is returned by the following call to pollutantmean()? You should round your output to 3 digits.

```
pollutantmean("specdata", "sulfate", 34)
```

```
## [1] 3.880701
```

4. What value is returned by the following call to pollutantmean()? You should round your output to 3 digits.

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

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

5. What value is printed at end of the following code?

```
cc <- complete("specdata", c(6, 10, 20, 34, 100, 200, 310))
print(cc$nobs)
```

```
## [1] 228 148 124 165 104 460 232
```

6. What value is printed at end of the following code?

```
cc <- complete("specdata", 54)
print(cc$nobs)
```

```
## [1] 219
```

7. What value is printed at end of the following code?

```
RNGversion("3.5.1")
set.seed(42)
cc <- complete("specdata", 332:1)
use <- sample(332, 10)
print(cc[use, "nobs"])
```

```
##  [1] 711 135  74 445 178  73  49   0 687 237
```

8. What value is printed at end of the following code?

```
cr <- corr("specdata")
cr <- sort(cr)
RNGversion("3.5.1")
set.seed(868)
out <- round(cr[sample(length(cr), 5)], 4)
print(out)
```

```
## [1]  0.2688  0.1127 -0.0085  0.4586  0.0447
```

9. What value is printed at end of the following code?

```
cr <- corr("specdata", 129)
cr <- sort(cr)
n <- length(cr)
RNGversion("3.5.1")
set.seed(197)
out <- c(n, round(cr[sample(n, 5)], 4))
print(out)
```

```
## [1] 243.0000   0.2540   0.0504  -0.1462  -0.1680   0.5969
```

10. What value is printed at end of the following code?

```
cr <- corr("specdata", 2000)
n <- length(cr)
cr <- corr("specdata", 1000)
cr <- sort(cr)
print(c(n, round(cr, 4)))
```

```
## [1]  0.0000 -0.0190  0.0419  0.1901
```

Cleanup

This final command removes the unzipped "specdata" directory if the deletespec variable is set to TRUE. This reduces the overall storage burden of this project by removing the files that we no longer need access to. The zipped file remains in the working directory, so "specdata" will be recreated anyways using the command in line 93 (unzip) if it is deleted here.

```
if(deletespec == TRUE){unlink("specdata")}
```