Stat 243 Homework 3

Daniel Lee

October 5, 2016

Problem 1

```
#This function removes white spaces from a column
#made for cleaning up storm_name column
#input: vector of strings
#output: vector of strings with white space removed
storm_remove_space <- function(name){
  gsub(" ", "", name, fixed = TRUE)
}

#replace missing values for position confidence with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_pc <- function(pc){
  sapply(pc, function(x) ifelse(x == 9, NA_character_, x))
}

#replace missing values for T-number with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_tn <- function(tn){
  sapply(tn, function(x) ifelse(x == 99, NA_character_, x))
}

#replace missing values for CI-number with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_ci <- function(ci){
  sapply(ci, function(x) ifelse(x == 99, NA_character_, x))
}

#replace missing values for max avg wind speed with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_max_avg_wind_spd <- function(wind_sp){
  sapply(wind_sp, function(x) ifelse(x == 999, NA_character_, x))
}

#replace missing values for time interval for averaging wind speed with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_time_int_avg_wind_spd <- function(time_int){
  sapply(time_int, function(x) ifelse(x == 99, NA_character_, x))
}
```

```r
#replace missing values for max wind gust with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_max_wind_gust <- function(gust){
  sapply(gust, function(x) ifelse(x == 999, NA_character_, x))
}

#replace missing value for gust period with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_gust_period <- function(per){
  sapply(per, function(x) ifelse(x == 9, NA_character_, x))
}

#replace missing value for central pressure with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_central_pres <- function(cp){
  sapply(cp, function(x) ifelse(x == 9999, NA_character_, x))
}

#replace missing value for radius of maximum winds with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_max_winds <- function(max_wind){
  sapply(max_wind, function(x) ifelse(x == 999, NA_character_, x))
}

#replace missing value for threshold value for wind speed with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_thres_val <- function(thres_wind_spd){
  sapply(thres_wind_spd, function(x) ifelse(x == 999, NA_character_, x))
}

#replace missing value for second threshold value for wind speed with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_sec_thres_val <- function(second_thres_wind){
  sapply(second_thres_wind, function(x) ifelse(x == 999, NA_character_, x))
}

#replace missing value for cyclone type with NA_character_
#input: vector of characters
#output: vector of characters with missing values replaced
mv_cycl_type <- function(cyc_typ){
  sapply(cyc_typ, function(x) ifelse(x == 09, NA_character_, x))
}

#converts latitude indicator into proper labels
#input: vector of characters with values 1 or 2
#output: vector of characters with 1 replaced with "north" and 2 replaced with "south"
fix_latitude_indicator <- function(latitude) {
```

```
  sapply(latitude, function(x) ifelse(x == 1, "north", "south"))
}

#includes the negative value if latitude is south
#input: two vectors of characters - one represents "north" or "south for latitude
#the second vector is the vector of latitude coordinates
#output: a vector of latitude values with negative signs included
format_latitude <- function(latitude_ind, latitude) {
  for(i in 1:length(latitude_ind)){
    if(latitude_ind[i] == "south"){
      latitude[i] = -1 * as.numeric(latitude[i])
    }
  }
  return(as.numeric(latitude)/10)
}

#converts longitude indicator into the proper labels
#input: vector of characters with values 1 or 2
#output: vector of characters with 1 replaced with "west" and 2 replaced with "east"
fix_longitude_indicator <- function(longitude) {
  sapply(longitude, function(x) ifelse(x == 1, "west", "east"))
}

#includes the negative value if longitude is west
#input: two vectors of characters - one represents "west" or "east for longitude
#the second vector is the vector of longitude coordinates
#output: a vector of longitude values with negative signs included
format_longitude <- function(longitude_ind, longitude) {
  for(i in 1:length(longitude_ind)){
    if(longitude_ind[i] == "west"){
      longitude[i] = -1 * as.numeric(longitude[i])
    }
  }
  return(as.numeric(longitude)/10)
}

#replaces the indicator values of position confidence with their corresponding labels
#input: vector with values 1 - 3
#output: vector with 1-3 replaced with the proper labels
fix_position_confidence <- function(pc){
  sapply(pc, function(x) switch(x, "1" = "good", "2" = "fair", "3" = "poor", NA_character_))
}

#replaces the indicator values of quality code wind report
#with their corresponding labels
#input: vector with values 1 - 5
#output: vector with 1-5 replaced with the proper labels
fix_qc_wind_report <- function(qc_wind_report){
  sapply(qc_wind_report, function(x) switch(x, "1" = "Aircraft or Dropsonde observation", "2" = "Over w
}

#replaces the indicator values of quality code for radius maximum winds
#with their corresponding labels
```

```
#input: vector with values 1 - 5
#output: vector with 1-5 replaced with the proper labels
fix_qc_rmw <- function(qc_rmw){
  sapply(qc_rmw, function(x) switch(x, "1" = "Aircraft observation", "2" = "Radar with well-defined eye
}

#replaces the indicator values of quality code for wind threshold
#with their corresponding labels
#input: vector with values 1 - 4
#output: vector with 1-4 replaced with the proper labels
fix_qc_wind_threshold <- function(qc_threshold){
  sapply(qc_threshold, function(x) switch(x, "1" = "Aircraft observations", "2" = "Surface observations
}

#replaces the indicator values of cyclone type with their corresponding labels
#input: vector with values 1 - 8
#output: vector with 1-8 replaced with the proper labels
fix_cyclone_type <- function(cyclone_type){
  sapply(cyclone_type, function(x) switch(x, "01" = "tropics; disturbance", "02" = "<34 knot winds, <17
}

#replaces the indicator values of source code with their corresponding labels
#input: vector with values 1 - 12
#output: vector with 1-12 replaced with the proper labels
fix_source_code <- function(source_code){
  sapply(source_code, function(x) switch(x, "01" = "RSMC Miami-Hurrican Center", "02" = "RSMC Tokyo-Typ
}

#read_wmo function takes in a wmo file, processes the data, and returns the processed
#data frame.
#input: year
#output: processed dataframe from the downloaded wmo file for the appropriate year
read_wmo <- function(year){
  stopifnot(is.numeric(year) & (year > 1847) & (year < 2016))
  url <- paste("ftp://eclipse.ncdc.noaa.gov/pub/ibtracs/v03r08/wmo/wmo_format/year/Year.", year, ".ibtr
    wmo_data <- read.fwf(url, width = c(9,
    10, 4, 2, 2, 2, 1, 3, 2, 1, 4, 2, 1, 2, 2, 3, 1, 2, 3, 1, 1, 4, 1, 1, 3, 1, 3, 4, 4, 4, 4, 1, 3, 4,

  wmo_data$storm_name <- storm_remove_space(wmo_data$storm_name)

  #replace missing values for position confidence with NA_character_
  wmo_data$position_conf <- mv_pc(wmo_data$position_conf)

  #replace missing values for T-number with NA_character_
  wmo_data$t_number <- mv_tn(wmo_data$t_number)

  #replace missing values for CI-number with NA_character_
  wmo_data$ci_number <- mv_ci(wmo_data$ci_number)

  #replace missing values for max avg wind speed with NA_character_
  wmo_data$max_avg_wind <- mv_max_avg_wind_spd(wmo_data$max_avg_wind)

  #replace missing values for time interval for averaging wind speed with NA_character_
```

```
wmo_data$time_int_wind_speed <- mv_time_int_avg_wind_spd(wmo_data$time_int_wind_speed)

#replace missing values for max wind gust with NA_character_
wmo_data$max_wind_gust <- mv_max_wind_gust(wmo_data$max_wind_gust)

#replace missing value for gust period with NA_character_
wmo_data$gust_period <- mv_gust_period(wmo_data$gust_period)

#replace missing value for central pressure with NA_character_
wmo_data$central_pressure <- mv_central_pres(wmo_data$central_pressure)

#replace missing value for radius of maximum winds with NA_character_
wmo_data$radius_max_wind <- mv_max_winds(wmo_data$radius_max_wind)

#replace missing value for threshold value for wind speed with NA_character_
wmo_data$threshold_wind_speed <- mv_thres_val(wmo_data$threshold_wind_speed)

#replace missing value for second threshold value for wind speed with NA_character_
wmo_data$sec_thres_wind <- mv_sec_thres_val(wmo_data$sec_thres_wind)

#replace missing value for cyclone type with NA_character_
wmo_data$cyc_type <- mv_cycl_type(wmo_data$cyc_type)

#replaces indicator values with labels for latitude
wmo_data$lat_ind <- fix_latitude_indicator(wmo_data$lat_ind)

#formats the latitude column such that negative values are added to south
wmo_data$latitude <- format_latitude(wmo_data$lat_ind, wmo_data$latitude)

#replaces indicator values with labels for longitude
wmo_data$long_ind <- fix_longitude_indicator(wmo_data$long_ind)

#formats the longitude column such that negative values are added to west
wmo_data$long <- format_longitude(wmo_data$long_ind, wmo_data$long)

#replaces categorical values with labels for position confidence
wmo_data$position_conf <- fix_position_confidence(wmo_data$position_conf)

#replaces categorical values with labels for quality code for wind report
wmo_data$qc_wind <- fix_qc_wind_report(wmo_data$qc_wind)

#replaces categorical values with labels for quality code for radius maximum winds
wmo_data$qc_rmw <- fix_qc_rmw(wmo_data$qc_rmw)

#replaces categorical values with labels for quality code for wind threshold
wmo_data$qc_wind_threshold <- fix_qc_wind_threshold(wmo_data$qc_wind_threshold)

#replaces categorical values with labels for cyclone type
wmo_data$cyc_type <- fix_cyclone_type(wmo_data$cyc_type)

#replaces categorical values with labels for source code
wmo_data$source_code <- fix_source_code(wmo_data$source_code)
```

```
  return(wmo_data)
}

#example of code run for read_wmo()
dat <- read_wmo(2005)
head(dat)
```

```
##         cic storm_name year month day hour lat_ind latitude check_sum_lad
## 1 01 SI2005     PHOEBE 2004     9   1    0   south     -5.3
## 2 01 SI2005     PHOEBE 2004     9   1    6   south     -5.5
## 3 01 SI2005     PHOEBE 2004     9   1   12   south     -6.2
## 4 01 SI2005     PHOEBE 2004     9   1   18   south     -7.1
## 5 01 SI2005     PHOEBE 2004     9   2    0   south     -8.0
## 6 01 SI2005     PHOEBE 2004     9   2    6   south     -8.3
##   long_ind long check_sum_long position_conf t_number ci_number
## 1     east 90.0                         <NA>     <NA>      <NA>
## 2     east 90.4                         <NA>     <NA>      <NA>
## 3     east 90.5                         <NA>     <NA>      <NA>
## 4     east 90.8                         <NA>     <NA>      <NA>
## 5     east 91.2                         <NA>     <NA>      <NA>
## 6     east 91.7                         <NA>     <NA>      <NA>
##   max_avg_wind units_per_hour time_int_wind_speed max_wind_gust
## 1           35              1                  10          <NA>
## 2           35              1                  10          <NA>
## 3           35              1                  10          <NA>
## 4           35              1                  10          <NA>
## 5           35              1                  10          <NA>
## 6           40              1                  10          <NA>
##   gust_period qc_wind central_pressure qc_pressure unit_length
## 1        <NA>    NULL             1000           9           9
## 2        <NA>    NULL              998           9           9
## 3        <NA>    NULL              998           9           9
## 4        <NA>    NULL              995           9           9
## 5        <NA>    NULL              992           9           9
## 6        <NA>    NULL              992           9           9
##   radius_max_wind qc_rmw threshold_wind_speed radius_sec_1_1
## 1            <NA>   NULL                <NA>           9999
## 2            <NA>   NULL                <NA>           9999
## 3            <NA>   NULL                <NA>           9999
## 4            <NA>   NULL                <NA>           9999
## 5            <NA>   NULL                <NA>           9999
## 6            <NA>   NULL                <NA>           9999
##   radius_sec_2_1 radius_sec_3_1 radius_sec_4_1 qc_wind_threshold
## 1           9999           9999           9999              NULL
## 2           9999           9999           9999              NULL
## 3           9999           9999           9999              NULL
## 4           9999           9999           9999              NULL
## 5           9999           9999           9999              NULL
## 6           9999           9999           9999              NULL
##   sec_thres_wind radius_sec_1_2 radius_sec_2_2 radius_sec_3_2
## 1           <NA>           9999           9999           9999
## 2           <NA>           9999           9999           9999
## 3           <NA>           9999           9999           9999
```

```
## 4           <NA>           9999           9999           9999
## 5           <NA>           9999           9999           9999
## 6           <NA>           9999           9999           9999
##   radius_sec_4_2 qc_wind_threshold_88 cyc_type source_code
## 1           9999                    9     <NA>        <NA>
## 2           9999                    9     <NA>        <NA>
## 3           9999                    9     <NA>        <NA>
## 4           9999                    9     <NA>        <NA>
## 5           9999                    9     <NA>        <NA>
## 6           9999                    9     <NA>        <NA>
```

Problem 2

```
library(stringr)
```

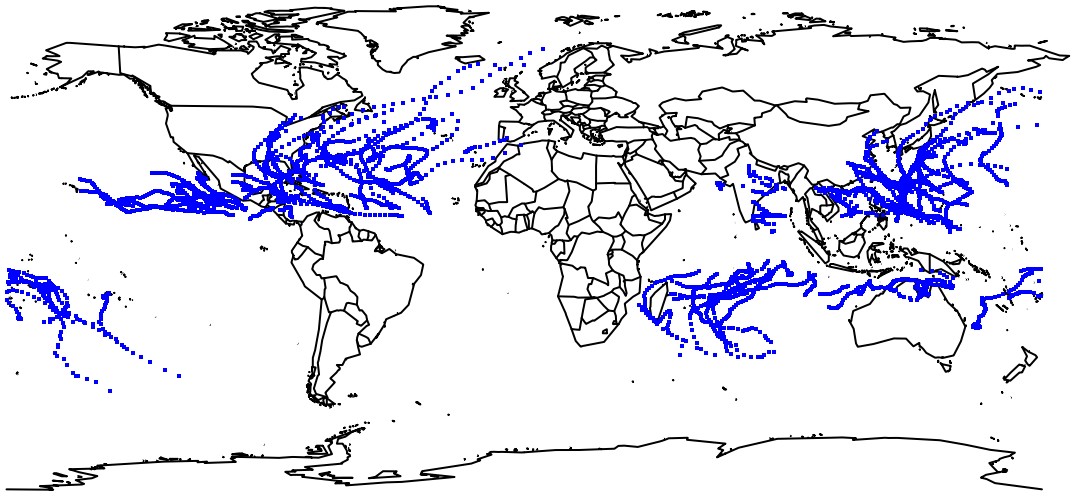
```
#Create and cbind a vector to the processed dataframe from part 1
#The new vector is called "Basin", which contains the two-letter code for basin
dat_2 <- cbind(dat, Basin = str_extract(dat$cic, "[A-Z]{2}"))

#Create a vector called Wind_Intensity
#This vector contains the categories of the max avg wind speeds
Wind_Intensity <- cut(as.numeric(dat_2$max_avg_wind), breaks = c(0, 10, 30, 55, 70, 85, 100, 120, 999),

#cbind Wind_intensity to the dataframe with processed data
dat_3 <- cbind(dat_2, "Wind Intensity" = Wind_Intensity)

#create the dataframe with processed information
#Desired columns are chosen
export <- dat_3[ , c("cic", "Basin", "storm_name", "year", "month", "day", "hour", "lat_ind", "latitude

#Export the dataframe stored in export to a csv file
write.table(export, file = "wmo_2005.csv", row.names=FALSE, sep = ",")
```

Problem 3

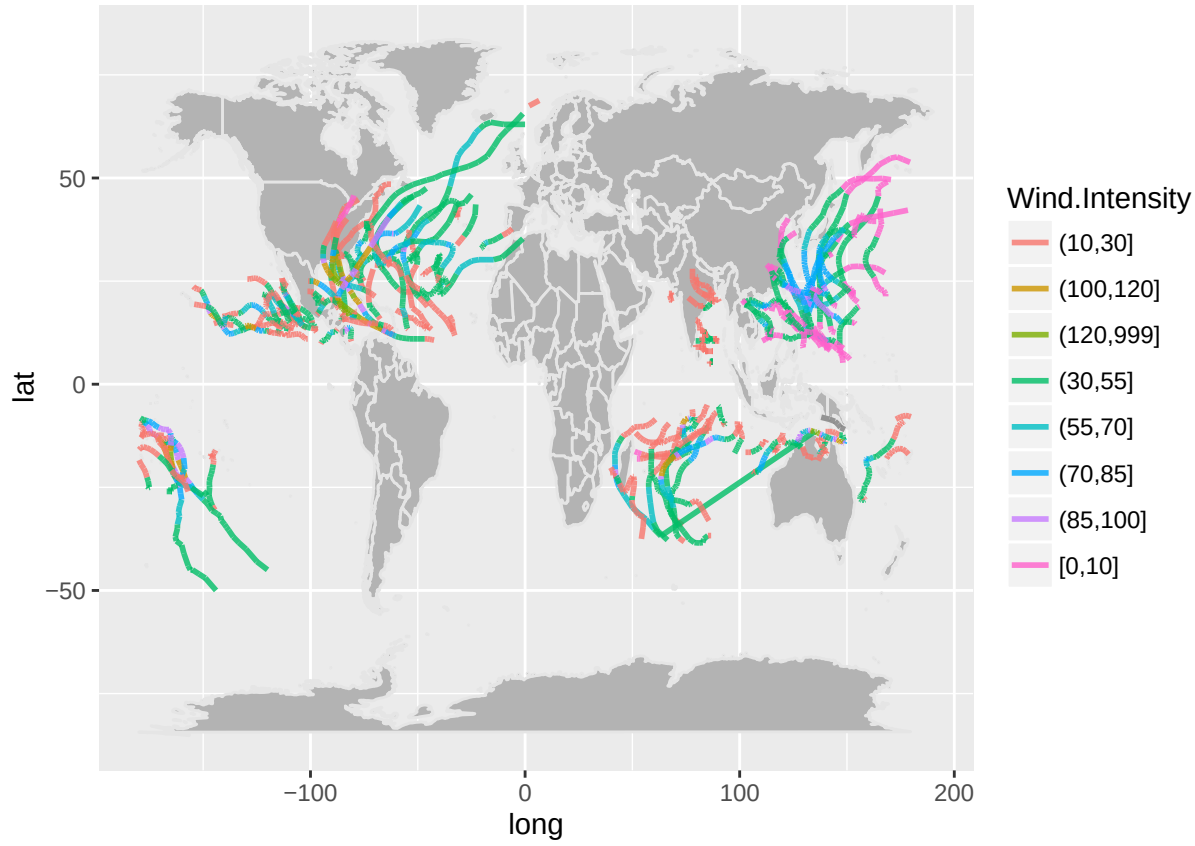
```
#Read in the exported file from Part 2
dat <- read.table("C:/Users/Daniel/Desktop/Fall 2016/Stat 243/wmo_2005.csv",
                  header = TRUE, sep = ",")
```

```
library(maps)
library(ggplot2)
```

```
# very basic map
map("world")
points(dat$long, dat$latitude, pch = ".", cex = 2, col = "blue")
```



```
world <- map_data("world")
# initialize "empty" ggplot object
p <- ggplot()
# add map layer
p <- p + geom_polygon(data = world,
aes(x=long, y=lat, group = group),
colour="gray90", fill="grey70" )
# for grouping purposes of storm trajectories
dat$group <- factor(paste0(dat$cic, dat$long_ind))
# add storm trajectories
p + geom_path(
data = dat,
aes(x = long, y = latitude, color = Wind.Intensity, group = group),
size = 1, alpha = 0.8)
```



Problem 4

I am going to use `wmo_2005.csv` that I exported from part 2 of this homework assignment.

```
#These are the bash codes for this particular problem.
cd /c/Users/Daniel/Desktop/Fall 2016/'Stat 243'

#How many (unique) storms were in the WMO file for 2005?

cut -f 3 -d "," wmo_2005.csv | tail -n +2 | sort | uniq | wc -l
100 lines

#Which storm has the most recorded values (i.e. lines)?

cut -f 3 -d "," wmo_2005.csv | tail -n +2 | sort | uniq -c | sort -nr | head -n 2
"NOTNAMED" occurs 183 times
"INGRID" occurs 82 times

#Display the number of recorded values per basin in decreasing order.

cut -f 2 -d "," wmo_2005.csv | tail -n +2 | sort | uniq -c | sort -nr
935 "NA"
868 "WP"
605 "SI"
```

```
458 "EP"
419 "SP"
248 "NI"

#How would you subset all the lines for basin NA (North Atlantic) and save it into a file NA-storms-200

grep '"NA"' wmo_2005.csv > NA-storms-2005.csv

#Write a for-loop to subset the lines for each basin in a separate file.

for basin in NA NI SA SI SP WP
do
  grep '"'$basin'"' wmo_2005.csv > $basin-storms-2005.csv
done
```

Problem 5

Write an R function, sample_file(), which takes a year number (e.g. 2010) and an arbitrary set of indices of observations of a .wmo file (e.g. indices = c(5, 9, 1000, 5000)) and which returns a data frame (a character vector of unprocessed fields would also be ok) with just those rows. The function should NOT read the entire .wmo file into memory and then subset. This is for sampling very large files. You should practice with the WMO data files for one year. One way this could be used is to take a random sample of observations from a file, based on using sample() to determine the indices.

```
#I used bash to obtain number of lines for a document with variable year.
function count_lines() {
  curl "ftp://eclipse.ncdc.noaa.gov/pub/ibtracs/v03r08/wmo/wmo_format/year/Year.$1.ibtracs_wmo.v03r08.w
}
```

```
library(readr)
```

```
#I tried to use the function system2() to obtain the number of lines directly from R. But this did not
#nrow_wmo_file <- system2("curl ftp://eclipse.ncdc.noaa.gov/pub/ibtracs/v03r08/wmo/wmo_format/year/Year

#Number was obtained using bash. I tried to do this using system2() function but did not work.
nrow_wmo_file <- 3533

#Function sample_file() that takes in a year and prints out a random sample of lines
#from a wmo file with the given year.
sample_file <- function(year){
  #stop code in case year is not numeric or out of bounds
  stopifnot(is.numeric(year) & (year > 1847) & (year < 2016))

  #processing the url address with paste
  url_year <- paste("ftp://eclipse.ncdc.noaa.gov/pub/ibtracs/v03r08/wmo/
                    wmo_format/year/Year.", year, ".ibtracs_wmo.v03r08.wmo", sep = "")

  #take random samples from nrow_wmo_file
  #sample size is 4. I chose it arbitrarily.
  sample_row_index <- sample(1:nrow_wmo_file, size = 4, replace = FALSE)
```

```r
  # create connection (in read mode)
  infile <- url(url_year, open = "r")

  sample_matrix <- matrix(data = NA, nrow = length(sample_row_index))
  j <- 1
  # loop over the lines of the file
  for(i in 1:max(sample_row_index)) {
    # read one line at a time
    one_line <- readLines(infile, n = 1)
    # are there any more lines to read?
    if (length(one_line) == 0) {
      break
    } else {
        if(sum(i == sample_row_index) > 0){
          sample_matrix[j,] <- one_line
          j <- j + 1
        }
    }
  }
  # close connection when done
  close(infile)

  return(as.data.frame(sample_matrix))
}

#Example of sample_file()
sample_file(2010)
```

```
##   V1
## 1 NA
## 2 NA
## 3 NA
## 4 NA
```