Module 14 In-Class Coding Exercises Dates and Times

Jeffrey Orcutt

2021-05-17

This assignment is your Dates and Times assignment. Once you have placed all your code responses in the code chunks, knit the file to a pdf. Upload the pdf to the Dates and Times Assignment link in Module 14 in Canvas.

There are 5 problems and each problem is worth 9 points each. The due date is Wednesday May 5 by 11:59pm CST.

Assignment of Problems for Student Presenters:

There are 5 problems and 5 student presenters for the Monday May 3 class. Students should present the following based on the row number in the Presentation Calendar Sign-Up Sheet:

student in row 46: problem 1 student in row 47: problem 2 student in row 48: problem 3 student in row 49: problem 4 student in row 45: problem 5

The lubridate and ggridges packages have been installed in this project.

1a. There is a csv file in the data folder of this project that contains daily weather readings at three different airports in Atlanta. The name of the file is weather\_atlanta.csv. Import the data into R using read\_csv(). Examine the data frame and notice that the Date column is not imported as a date variable.

weather\_atl <- read.csv("/cloud/project/data/weather\_atlanta.csv")

1. Perform the following with the atlanta weather data.

* Create a new data frame entitled weather\_atlanta\_hart that includes only the weather readings from the Hartsfield Airport.
* Use mutate() to overwrite the Date column with a parsed version of Date.
* Use str() to see how the parsed date is now a date in the R data frame weather\_atlanta\_hart.

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(lubridate)

##   
## Attaching package: 'lubridate'

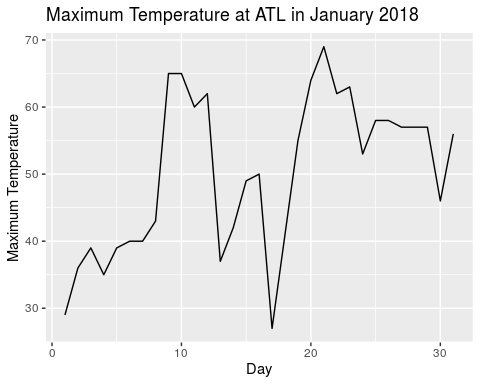
## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(ggplot2)  
weather\_atlanta\_hart <- NULL  
  
weather\_atlanta\_hart <- weather\_atl %>% filter(AirportCode == "ATL") %>% mutate(Date= mdy(Date))

1. Construct a line plot that has each day in January on the x-axis and the maximum temperature on the y-axis for the Hartsfield Atlanta airport.

Create a meaningful title and ensure there are meaningful labels on each axis.

weather\_atlanta\_hart %>% ggplot(aes(x=day(Date), y=TempMax)) + geom\_line() + labs(x="Day", y="Maximum Temperature", title="Maximum Temperature at ATL in January 2018")



2a. There is a csv file in the data folder of this project that contains information about storms (hurricanes, typhoons, and cyclones) that has been collected from 1980 to 2016. Every six hours the wind speed of the storm has been recorded. The name of the data file is storm\_hourly.csv. Import the data into R using read\_csv(). Examine the data frame and notice that the ISO\_time column is not imported as a date variable.

storm\_hourly <- read.csv("/cloud/project/data/storm\_hourly.csv")

1. Perform the following using the hourly storm data.

-Create a new data frame entitled storm\_hourly\_Katrina that includes hurricane Katrina. -Use mutate() to create a new column datetime which is the parsed version of ISO\_time. -Use str() to see how the parsed date is now a date in the R data frame storm\_hourly\_Katrina.

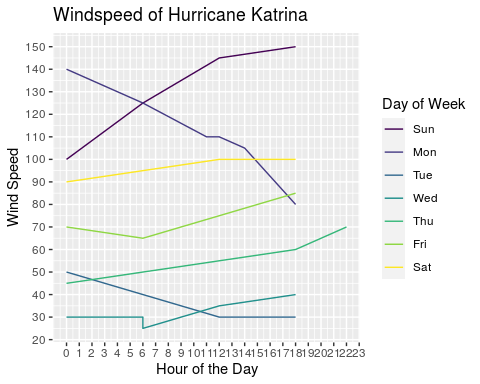
storm\_hourly\_Katrina <- storm\_hourly %>% filter(Name=="KATRINA") %>% mutate(datetime = dmy\_hms(ISO\_time))

1. Construct a line plot that has each hourly time increment on the x-axis and the wind speed on the y-axis for Hurricane Katrina.

Note: You do not have to perform any more parsing of the data as long as the datetime column was parsed correctly in part b.

Create a meaningful title and ensure there are meaningful labels on each axis.

ggplot(storm\_hourly\_Katrina, aes(x=hour(datetime), y=Wind, color=wday(datetime, label=TRUE))) + geom\_line() + scale\_x\_continuous(breaks = seq(0,24, by=1)) + scale\_y\_continuous(breaks = seq(20, 160, by=10)) + labs(y="Wind Speed", x="Hour of the Day", title = "Windspeed of Hurricane Katrina", color="Day of Week")



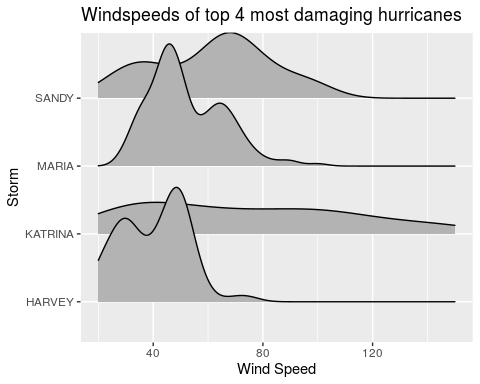
Problem #3a uses the storm\_hourly data.

3a. The five most costly storms in US history were Katrina, Harvey, Maria, and Sandy. Use the storm\_hourly data and perform the following.

* Create a ridgeline plot (formerly known as a joy plot) for these five storms where the wind speed is displayed on the x-axis and the name of the storms are on the y-axis.
* Create a meaningful title and ensure there are meaningful labels on each axis.

library(ggridges)  
graph\_this <- storm\_hourly %>% filter(Name %in% c("KATRINA", "HARVEY", "MARIA", "SANDY"))  
  
ggplot(graph\_this, aes(x=Wind, y= Name, height=..density..)) + geom\_density\_ridges(stat="density") + labs(title="Windspeeds of top 4 most damaging hurricanes", y="Storm", x="Wind Speed")

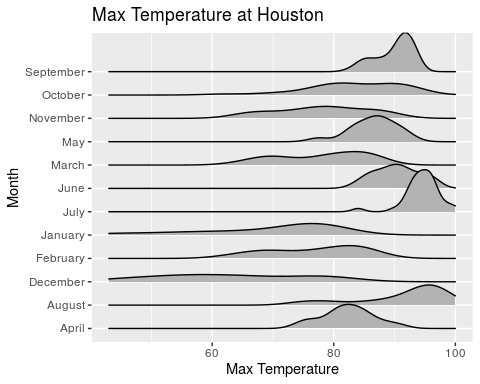
## Warning: Removed 49 rows containing non-finite values (stat\_density).



1. There is a csv file in the data folder of this project that contains daily rain and temperature readings at the airport in Houston, TX. The name of the file is weather\_houston.csv. Perform the following.

* Import the data into R using read\_csv().
* Examine the data frame and notice that the Date column is not imported as a date variable.
* Parse the Date column appropriately.
* Create a ridgeline plot that has maximum temperature on the x-axis and the months on the y-axis. Ensure the months are labeled as Jan, Feb, etc…
* Create a meaningful title and ensure there are meaningful labels on each axis.

data3b <- read.csv("/cloud/project/data/weather\_houston.csv")  
data3b <- data3b %>% mutate(Date=dmy(Date)) %>% dplyr::arrange(Date)  
ggplot(data3b, aes(x=TempMax, y= months(Date), height=..density..)) + geom\_density\_ridges(stat="density") + labs(title="Max Temperature at Houston", y="Month", x="Max Temperature")



1. There is a csv file in the data folder of this project that contains a description and damage estimates (adjusted for inflation) for storms in the US with damages greater than one billion dollars. The name of the file is storm\_damage.csv. Perform the following.

* Import the data into R using read\_csv().
* Examine the data frame and notice that the Date column is not imported as a date variable.
* Create a new column Date which is the parsed version of Date. Date represents the day the storm made landfall.
* Create a new column AssessmentDate that is one month past the date of landfall. Explain whether the function you selected is a duration or period and why you selected the function you did.
* Create a new column Anniversary that is the date of the 10-year anniversary for each storm. Explain whether the function you selected is a duration or period and why you selected the function you did.

library(lubridate)  
data4 <- read.csv("/cloud/project/data/storm\_damage.csv")  
data4$Date <- as.Date(data4$Date, origin=as.Date("1960-01-01"))  
  
#both functions use a period due to being relative times  
data4$AssessmentDate <- data4$Date + months(1)  
data4$Anniversary <- data4$Date + years(10)

1. There is a csv file in the data folder of this project that contains hourly rain amounts for the Panther Junction, TX, station located in Big Bend National Park. The DateTime column contains date/time values. The name of the file is np\_hourlyrain.csv. Perform the following:

* Import the data into R using read\_csv().
* Create a report/table/graph that you think would be useful if you were working for an analyst at the US National Park Service. Your final output must incorporate the use of some component of the DateTime column.

data4b <- read.csv("/cloud/project/data/np\_hourlyrain.csv")  
  
data4b <- data4b %>% mutate(DateTime = dmy\_hms(DateTime))   
  
#this didn't show anything of value  
#graph4b <-ggplot(data4b, aes(x=Rain, y=months(DateTime), height=..density..)) + geom\_ridgeline(stat="density")  
  
graph4b <-ggplot(data4b, aes(y=Rain, x=months(DateTime)))+geom\_bar(stat="identity") +labs(title="Sum of Rainfall by Month", x="Month", y="Inches of Rain")+  
 theme(axis.text.x = element\_text(angle = 90))  
#I need to sort this, but am out of time to figure that out.   
  
graph4b

## Warning: Removed 11 rows containing missing values (position\_stack).

