

MM916 Project 1: Exploratory analysis of Atlantic hurricanes

Background

Hurdat2 is a database of hurricanes and similar storms produced by the US National Oceanic and Atmospheric Administration (NOAA). It contains data on

- the track each storm followed (latitude-longitude coordinates)
- the wind speed in knots (1 knot = 0.52 m/s) and the classification of the storm at each point along its track.
- the point at which each storm crossed onto land if it did cross onto land (“landfall”)

and a number of other variables, all organised by a unique ID code for each storm and sometimes a name. Databases like this are crucial for understanding how climate change has affected patterns of extreme weather and predicting how the choices we make about carbon emissions shape extreme weather in the future.

The original database is stored in a somewhat awkward format, but we have tidied it for you (you’re welcome!), and you will find the full dataset for the Atlantic Ocean region in the dataframe `hurrs` inside the file `Hurdat2_tidy.RData`.

The documentation from NOAA is at <https://www.nhc.noaa.gov/data/hurdat/hurdat2-format-nov2019.pdf>. Background information on the Hurdat2 project and hurricane science in general are at <https://www.noaa.gov/education/resource-collections/weather-atmosphere/hurricanes> https://www.aoml.noaa.gov/hrd/hurdat/Data_Storm.html

We have removed some variables and simplified the classification system: we will use the word “storm” to mean “anything with a unique ID in the database” and the column `isHurricane` is TRUE/FALSE depending on whether the storm system had reached hurricane status (based on wind speed) at a given point in time. There aren’t any questions that depend on the distinction between “storm” and “hurricane”.

Instructions

Make a single Word, PDF, or RMarkdown file containing your answers to the questions below, including the R code you used to answer them, and the plots produced by your code. Submit this document via MyPlace.

It is not necessary to run any statistical tests of your interpretations of the data. This project is about *exploratory* data analysis and so guesses based on what you see by eye are sufficient (this is how statistical hypotheses are generated, not tested).

Please do not be discouraged or skip later questions if you have difficulty with early questions. We will give **partial credit** for partial solutions, including code that has the right idea but does not run, plots that include some but not all required elements, and so on.

You can make use of everything on MyPlace and also general web searches, but you may not work in groups or copy from a classmate: these are individual projects and everything below needs to be your own work.

continue to the next page

Question 1: What's in this dataset?

(2,5,1,2 marks)

(a) Load the dataframe. Pick one famous hurricane that you know the name of (Google “famous hurricanes” if you can’t think of any) and make a dataframe similar to `hurrs` but containing only the data for your chosen hurricane. (Note that the people who name hurricanes have reused names sometimes.)

For your own understanding, look at which dataframe columns are constant across the observations for this one hurricane and which ones vary. (You don’t have to write anything down for this, it’s just general advice.)

```
load("Hurdat2_tidy.RData")
katrina = hurrs %>% filter(name=="KATRINA" & year==2005)
```

(b) Write a function called `Hurdat2_summary` that takes the `hurrs` dataframe as an input and returns a **named list** containing

- the total number of storms
- the range of years covered
- a vector containing the number of track points for each storm

Test your function by running it for three cases:

- the full `hurrs` dataframe
- a version that has been filtered to contain only storms from a single year
- the one-hurricane dataframe you made in (a)

```
Hurdat2_summary = function(hurrs) {
  total = length(unique(hurrs$id))                      # total number of storms
  yearRange = c(min(hurrs$year),max(hurrs$year))        # range of years
  hurrs %>% group_by(id) %>%
    summarise(numPoints = n()) -> df                  # dataframe with number of points per storm

  list(total=total,
       yearRange=yearRange,
       numPoints=df$numPoints)
}

# test it
summary_full = Hurdat2_summary(hurrs)
summary_full$total # this is how many elements are in summary_full$numPoints--
```



```
## [1] 1924
# long output so not including it in the solutions
summary_1961 = Hurdat2_summary(hurrs %>% filter(year==1961))
summary_1961
```



```
## $total
## [1] 12
##
## $yearRange
## [1] 1961 1961
##
## $numPoints
## [1] 32 58 60 55 70 15 43 23 26 38 20 15
summary_katrina = Hurdat2_summary(katrina)
summary_katrina
```

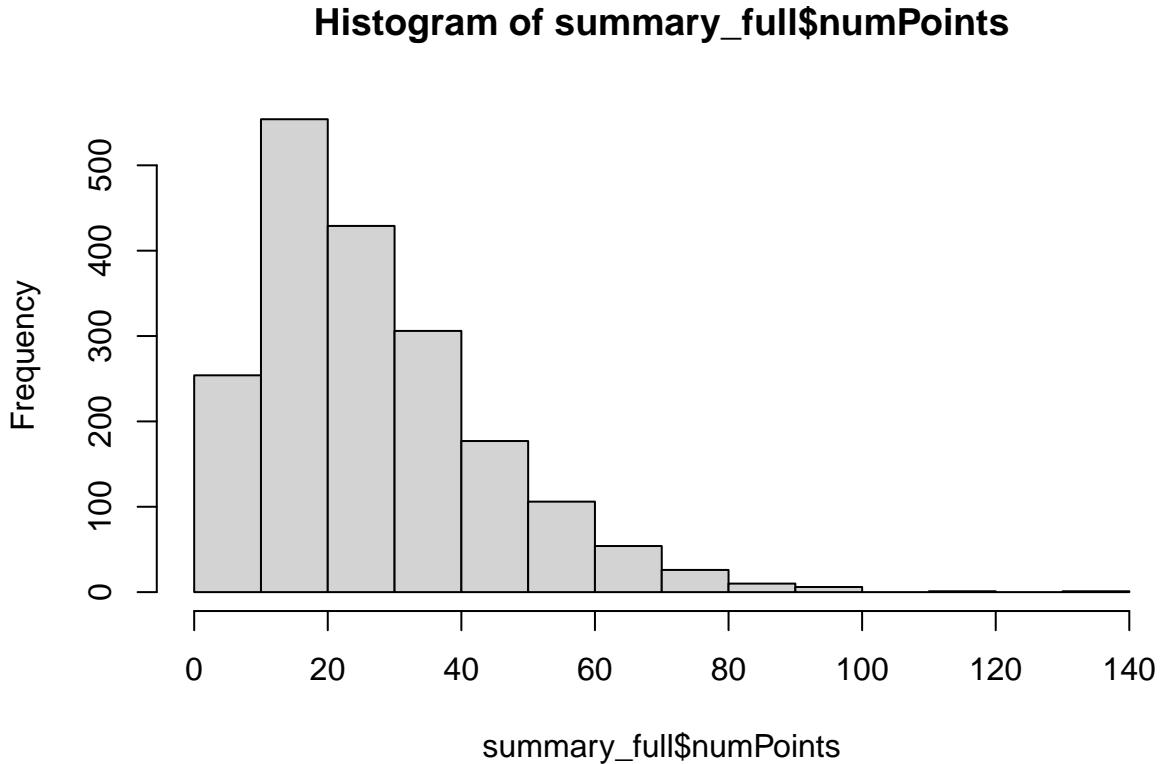
```

## $total
## [1] 1
##
## $yearRange
## [1] 2005 2005
##
## $numPoints
## [1] 34

```

(c) Make a histogram of the number of track points, using the output of your function for the `hurrs` case.

```
hist(summary_full$numPoints)
```



(d) What year did they begin naming hurricanes and other storms? (Answer this with R code and the `hurrs` dataframe—you can check if you’re right using Google).

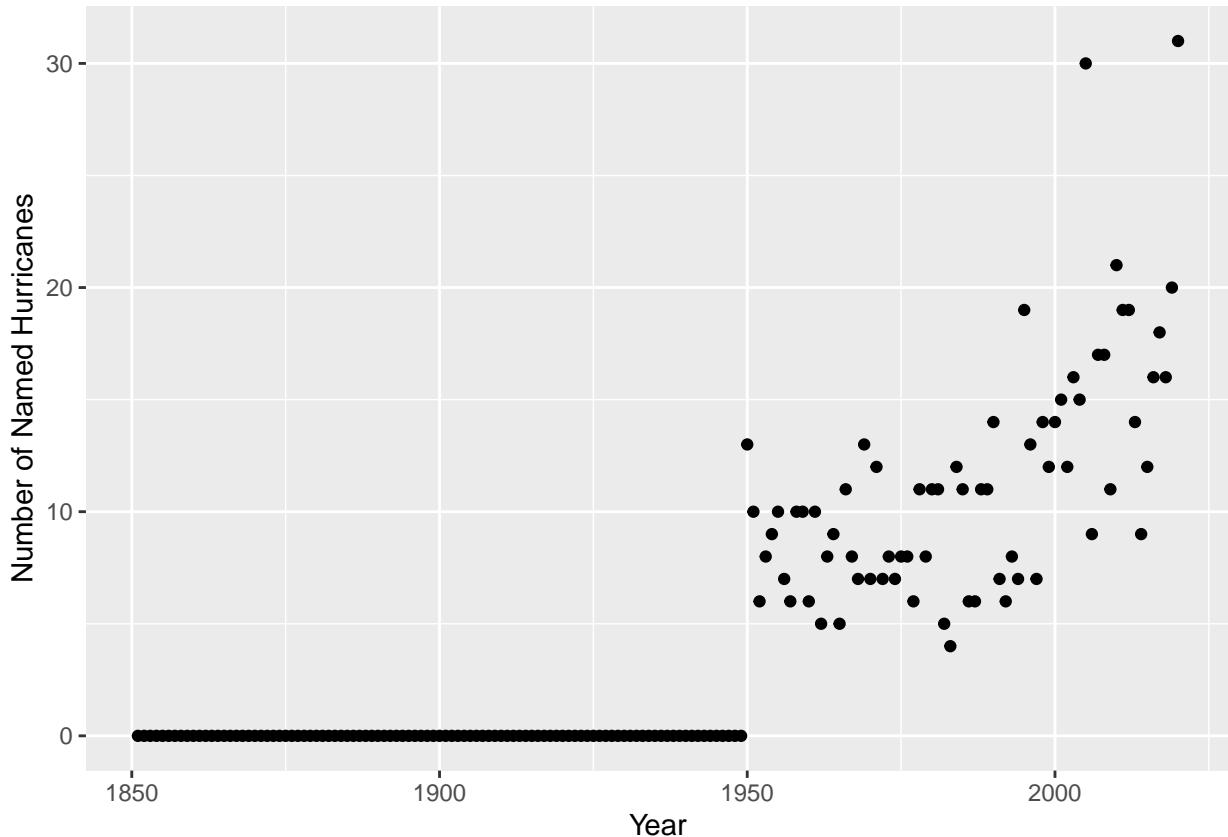
```

# quick answer:
named_hurrs = hurrs %>% filter(name != "UNNAMED")
min(named_hurrs$year)

## [1] 1950

# plot that makes this very clear:
hurrs %>% group_by(id) %>%
  summarise(isNamed = any(name != "UNNAMED"), year=first(year)) %>%
  group_by(year) %>% summarise(num_named = sum(isNamed)) %>%
  ggplot() + geom_point(aes(year,num_named)) +
  labs(x="Year", y="Number of Named Hurricanes")

```



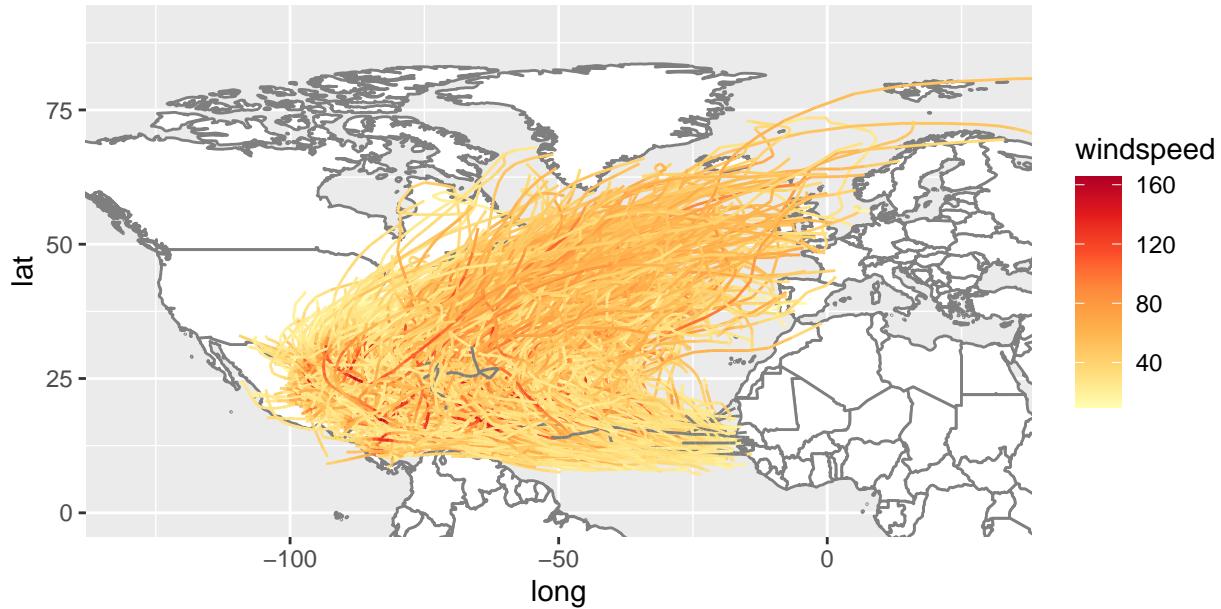
Question 2: Mapping

(4,3 marks)

Here is some code that plots all the storms in the database atop a simple world map.

```
# you might need to run
#   install.packages("maps")
# and restart R to make this work.
hurricaneBasemap = function() {
  library(maps)
  world = map_data("world")
  basemap = ggplot() +
    coord_fixed(xlim=c(-130,30),ylim=c(0,90)) +
    geom_polygon(data=world, aes(x=long,y=lat,group=group), color="gray50", fill="white")
  return(basemap)
}
hurricaneBasemap() +
  geom_path(data=hurrs, aes(x=longitude,y=latitude,group=id,color=windspeed)) +
  scale_color_distiller(type="seq",direction=1,palette="YlOrRd")

##
## Attaching package: 'maps'
## The following object is masked from 'package:purrr':
## 
##     map
```

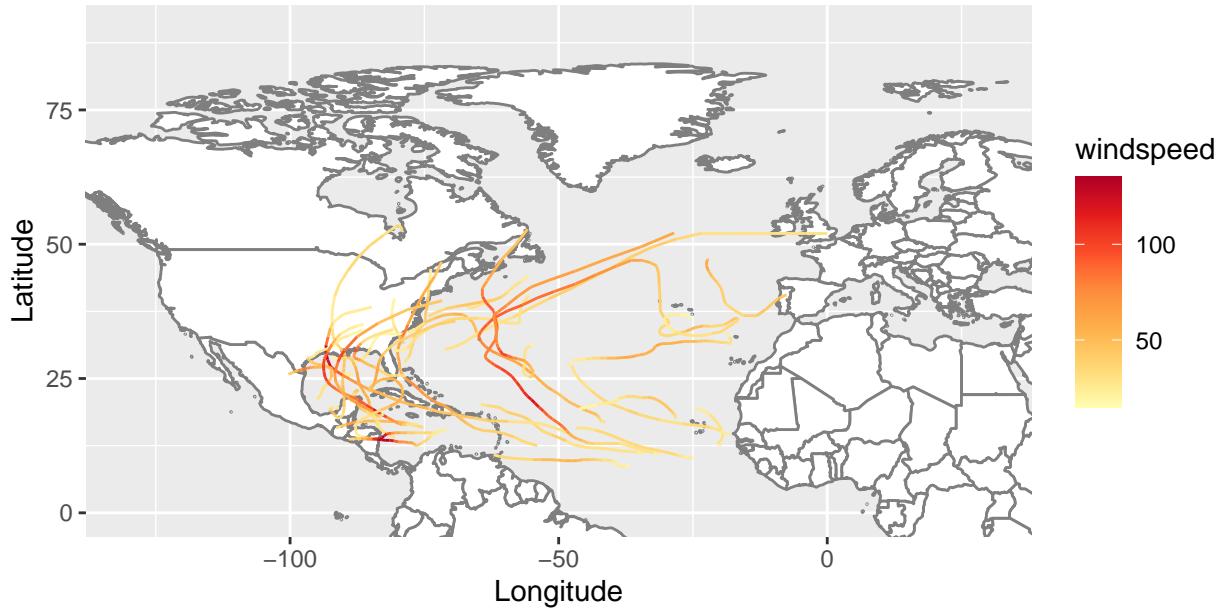


(a) Figure out how it works and add a helpful one-line comment to each of the 9 lines of code.

(b) Now write a function that takes a year as input and makes a similar plot for only the hurricanes during that year. The function doesn't need to return a value, but it does need to generate a plot. Test your function by making the plot for a year of your choice.

Making small changes to the code above is probably the easiest way to do this, but if you really want to you could start over with base R graphics. Please improve the labelling and give your plot a title.

```
oneYearOfHurricanes = function(yr) {
  hurricaneBasemap() +
    geom_path(data=hurrs %>% filter(year==yr),
              aes(x=longitude,y=latitude,group=id,color=windspeed)) +
    scale_color_distiller(type="seq",direction=1,palette="YlOrRd") +
    labs(x="Longitude",y="Latitude",paste("Atlantic hurricanes in ",yr))
}
oneYearOfHurricanes(2020)
```



Question 3: The local perspective (4,2,2,6 marks)

`world.cities` in the `maps` package gives information on lots of world cities including their latitude-longitude (lat-long) coordinates. You can access this data with the lines

```
library(maps)
cities = world.cities
```

Now `cities` is a dataframe in your Environment.

(a) The formula for calculating distance D in km from latitude-longitude coordinates is

$$D = 111.325 \sqrt{dx^2 + dy^2}$$

where

$$dx = (x_2 - x_1) \cos\left(\frac{\pi}{180} \frac{y_1+y_2}{2}\right)$$

$$dy = y_2 - y_1$$

and x is longitude and y latitude, in degrees. (The $\pi/180$ is there to convert from degrees to radians.) In other words, one degree of latitude is always 111.325 km, but the number of km in one degree of longitude varies as the cosine of the mean latitude.

Write a function that calculates distance between (lon1,lat1) and (lon2,lat2). Test it.

```
distance_km = function(lon1, lat1, lon2, lat2) {
  dx = (lon2-lon1) * cos(pi/180*0.5*(lat1+lat2));
  dy = (lat2-lat1);
  D = 111.325 * sqrt(dx^2 + dy^2);
  return(D)
}
# test cases
distance_km(-30,44,-30,45) # one degree of lat only: should be 111.325
## [1] 111.325
```

```

distance_km(-30,60,-29,60) # one degree of lon only, at 60N: should be 111.325/2 because cos(60) = 1/2

## [1] 55.6625
distance_km(c(-30,-30),c(44,60),c(-30,-29),c(45,60)) # check that it works for vector input

## [1] 111.3250 55.6625

```

(b) Adapt your function so that instead of (lon2,lat2) it takes the name of a city as input, finds its latitude-longitude coordinates, and returns the distance to (lat1,lon1). Test it.

You might want to give this function a new name and have it call the function from part (a). Either find a way to deal with the situation where multiple cities have the same name, or just pick examples that avoid that situation.

```

distanceToCity = function(lon1,lat1,cityName) {
  cities = world.cities;                      # need to call library(maps) before using this function
  theCity = cities %>% filter(name==cityName)
  D = distance_km(lon1,lat1,theCity$lon,theCity$lat)
  return(D)
}
# test it
testCity = cities %>% filter(name=="Vancouver" & country.etc=="Canada")
testLon = testCity$lon
testLat = testCity$lat
distanceToCity(testLon,testLat,"Seattle") # Google Maps says this should be roughly 200 km

## [1] 193.5664

```

(c) A common mistake in working with latitudes and longitudes is to swap them around. It isn't always possible to catch user errors of this sort, but one test is that longitude can take values from -180 to 180 while latitude is always -90 to 90. Modify your function to check whether it's likely that the user has swapped lat1 and lon1 and if so, provide a useful warning message.

```

isValidLat = function(x) {
  return(x >= -90 & x <= 90)
}

distanceToCity = function(lon1,lat1,cityName) {
  cities = world.cities;                      # need to call library(maps) before using this function
  theCity = cities %>% filter(name==cityName)
  if (isValidLat(lon1) & !isValidLat(lat1)) {
    warning("You might have swapped the longitude and latitude inputs.")
  }
  D = distance_km(lon1,lat1,theCity$lon,theCity$lat)
  return(D)
}
# test it
distanceToCity(testLat,testLon,"Seattle") # inputs swapped to check if we get the warning message

## Warning in distanceToCity(testLat, testLon, "Seattle"): You might have swapped
## the longitude and latitude inputs.

## [1] 24280.39

```

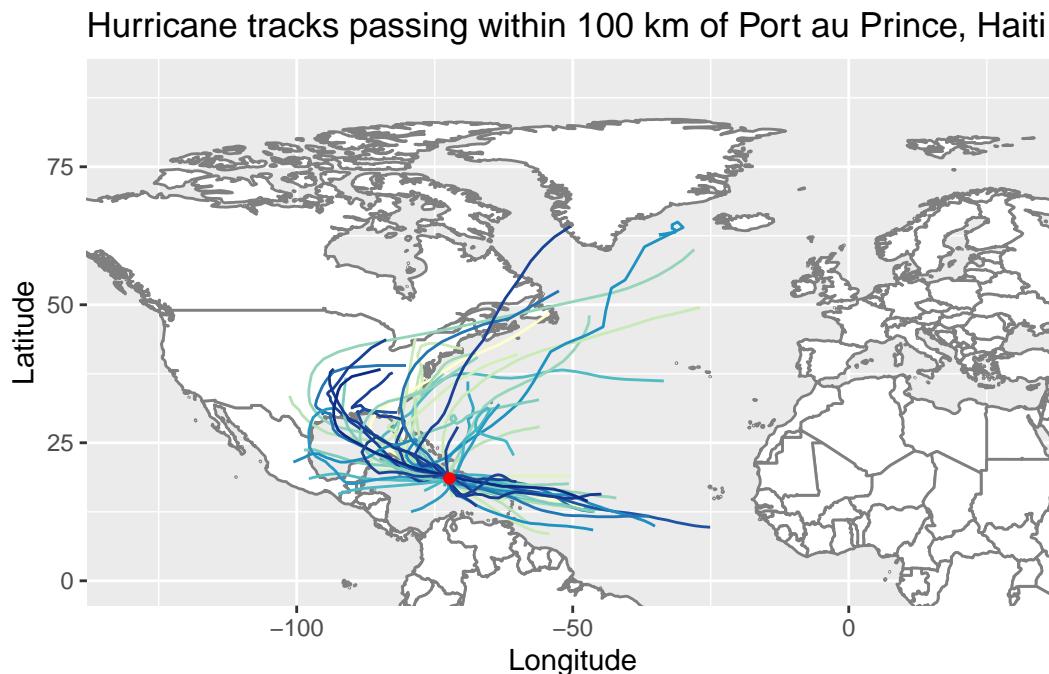
(d) Finally, choose a eastern North American or Caribbean city, and use your function to determine which storms passed within 100 km of it. Plot those storms on a map like those in Q1, with a dot on top showing where the city is.

```

cityName = "Port-au-Prince"
lonCity = (cities %>% filter(name==cityName))$lon
latCity = (cities %>% filter(name==cityName))$lat
distToCity = distanceToCity(hurrs$longitude,hurrs$latitude,cityName);

## Warning in if (isValidLat(lon1) & !isValidLat(lat1)) {: the condition has length
## > 1 and only the first element will be used
nearCity_ids = unique(hurrs$id[distToCity<100])
hurricaneBasemap() +
  geom_path(data=hurrs %>% filter(id %in% nearCity_ids),
            aes(x=longitude,y=latitude,group=id,color=year)) +
  scale_color_distiller(type="seq",direction=1,palette="YlGnBu") +
  geom_point(aes(x=lonCity,y=latCity),color="red") +
  labs(x="Longitude",y="Latitude",
       title="Hurricane tracks passing within 100 km of Port au Prince, Haiti")

```



Question 4: Climate change

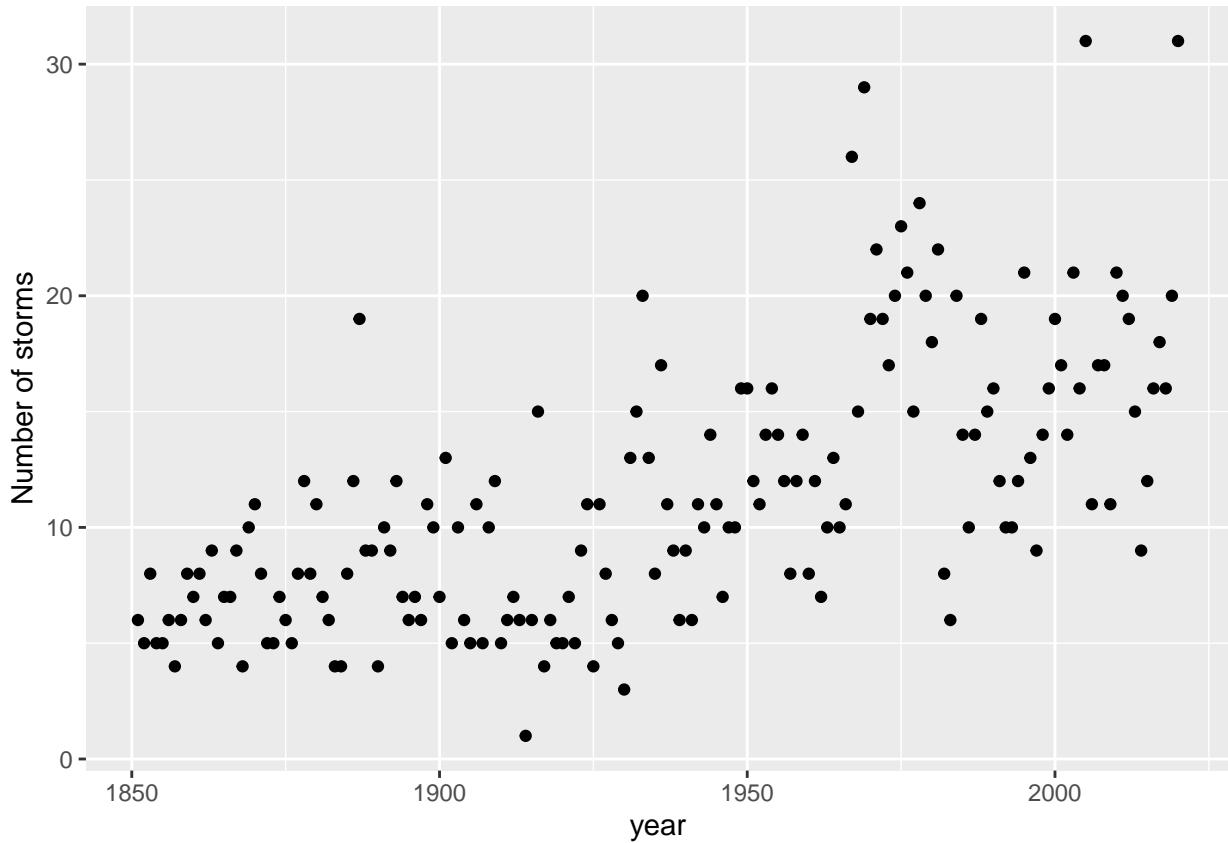
(3,4,3,3,2,4 marks)

(a) Globally, the number of hurricane-strength storms has been clearly increasing because of climate change. Is this true in the Atlantic specifically? Plot the number of storms in each year.

```

df = hurrs %>% group_by(year) %>% summarise(num=length(unique(id)))
ggplot(df) + geom_point(aes(x=year,y=num)) +
  labs(x="year",y="Number of storms")

```

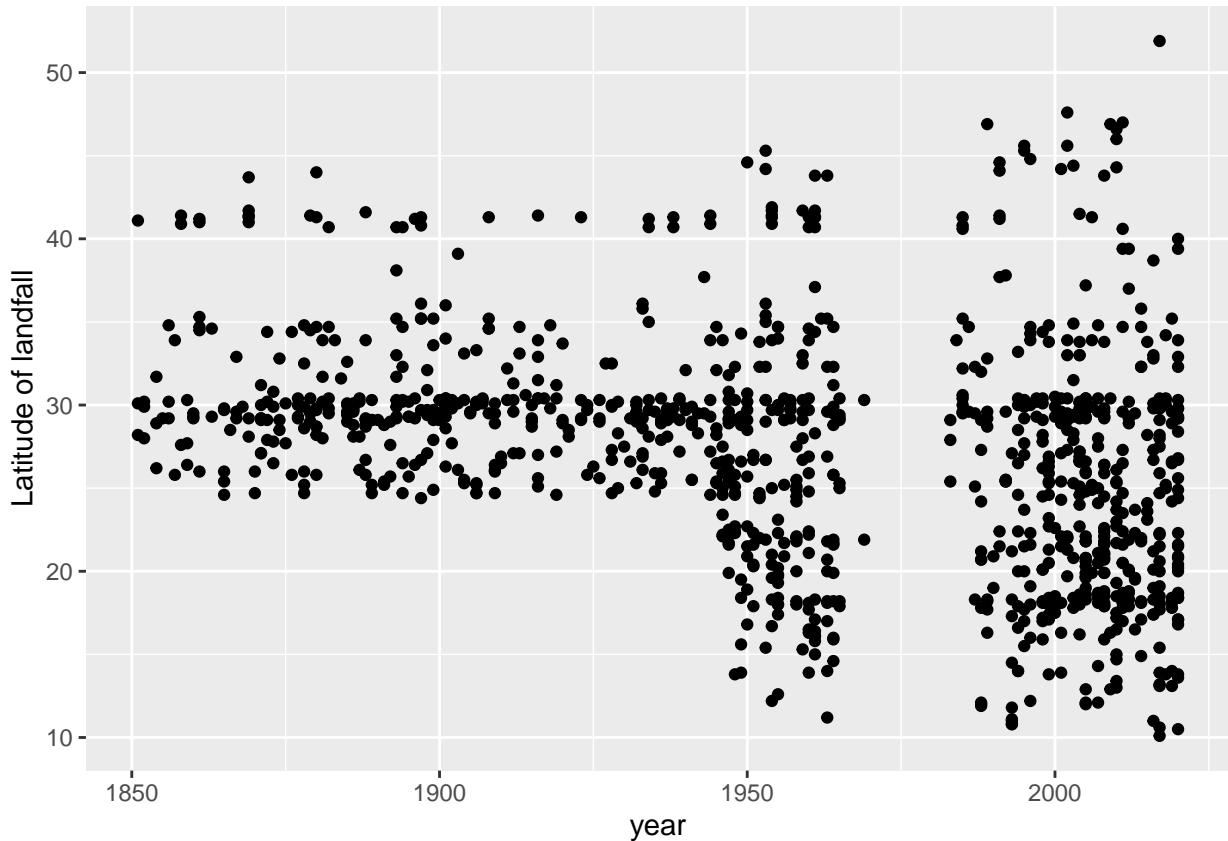


```
# Yes, the data are consistent with the hypothesis that Atlantic hurricanes and similar storms
# are increasing because of climate change.
```

(b) The total number isn't the only aspect of hurricane statistics that might change with climate change. What about the geographical area affected? Make a plot that lets you visually evaluate whether the latitude range where storms cross the coastline (make landfall) is expanding in recent decades.

Does your plot suggest any concerns about data quality and data coverage in this database?

```
ggplot(hurrs %>% filter(isLandfall)) +
  geom_point(aes(year,latitude)) +
  labs(x="year",y="Latitude of landfall")
```



```
# There might be a trend in recent decades toward an expanded range of landfall latitudes.
# The sudden change at 1950 is probably an artifact of how the database was constructed,
# and the data gap in the 1960s also doesn't make sense in light of the gradual change in part (a).
```

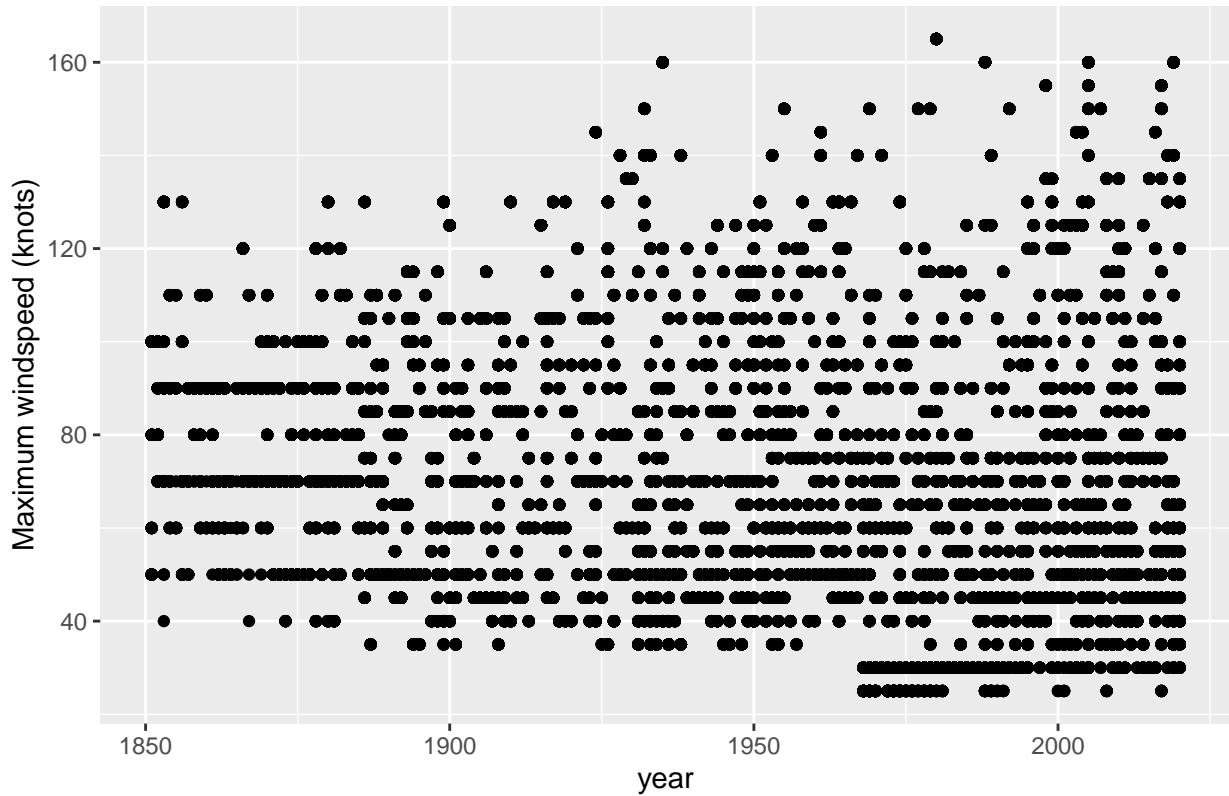
(c) Is peak wind speed increasing over time? Answer this using whatever kind of plot or table you think is most helpful.

```
library(dplyr)
library(ggplot2)

hurrs %>% group_by(id) %>% summarise(maxwind=max(windspeed),year) %>%
  ggplot() + geom_point(aes(x=year,y=maxwind)) +
  labs(x="year",y="Maximum windspeed (knots)",title="Peak hurricane wind speeds over time")

## `summarise()` has grouped output by 'id'. You can override using the `groups` argument.
## Warning: Removed 971 rows containing missing values (geom_point).
```

Peak hurricane wind speeds over time

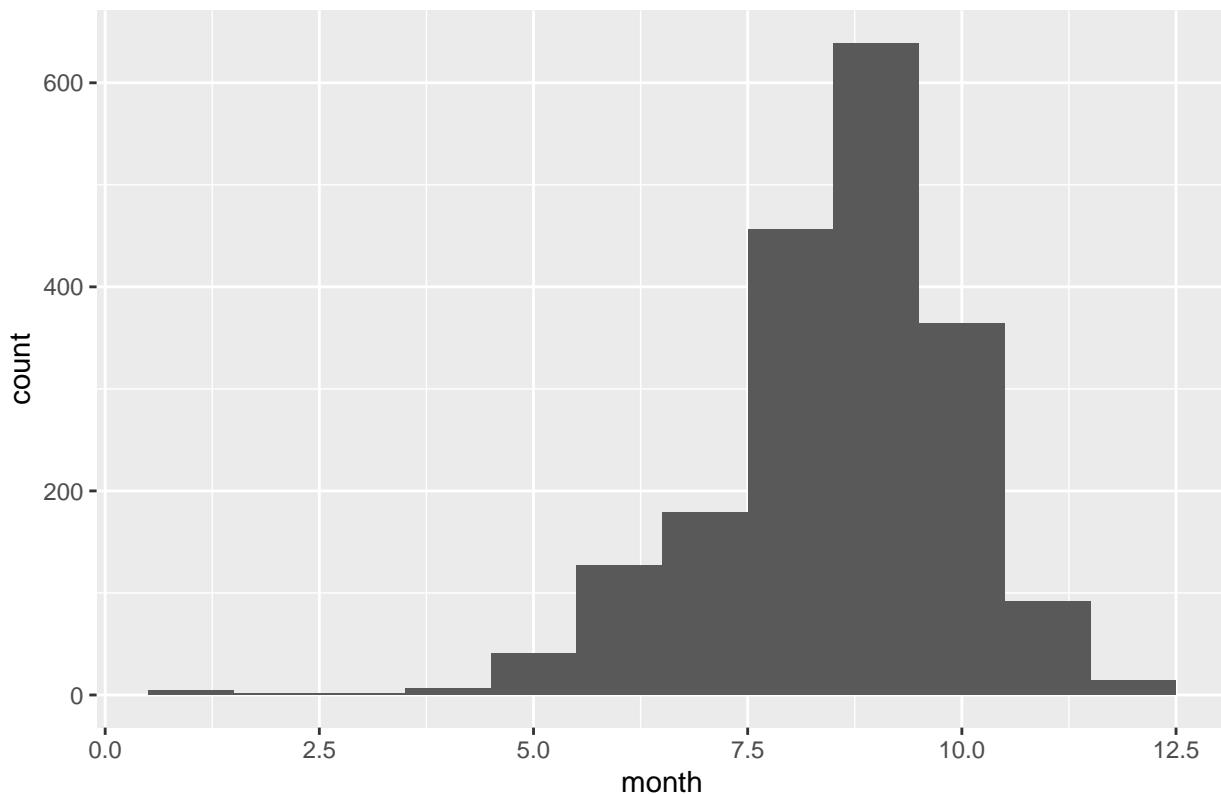


```
# Fortunately, there isn't much evidence that hurricanes are getting more severe over time,  
# just more frequent and more widespread.
```

(d) What about the seasonal timing of hurricanes? First, make a histogram of storms by month over the entire dataset. This should show that there is a distinct hurricane season—they tend to happen at a certain time of year. Use a bin width of 1 month.

```
hurrs %>% group_by(id) %>% summarise(month=first(month)) %>%  
  ggplot() + geom_histogram(aes(x=month), binwidth=1) +  
  labs(title="Atlantic hurricanes by month")
```

Atlantic hurricanes by month

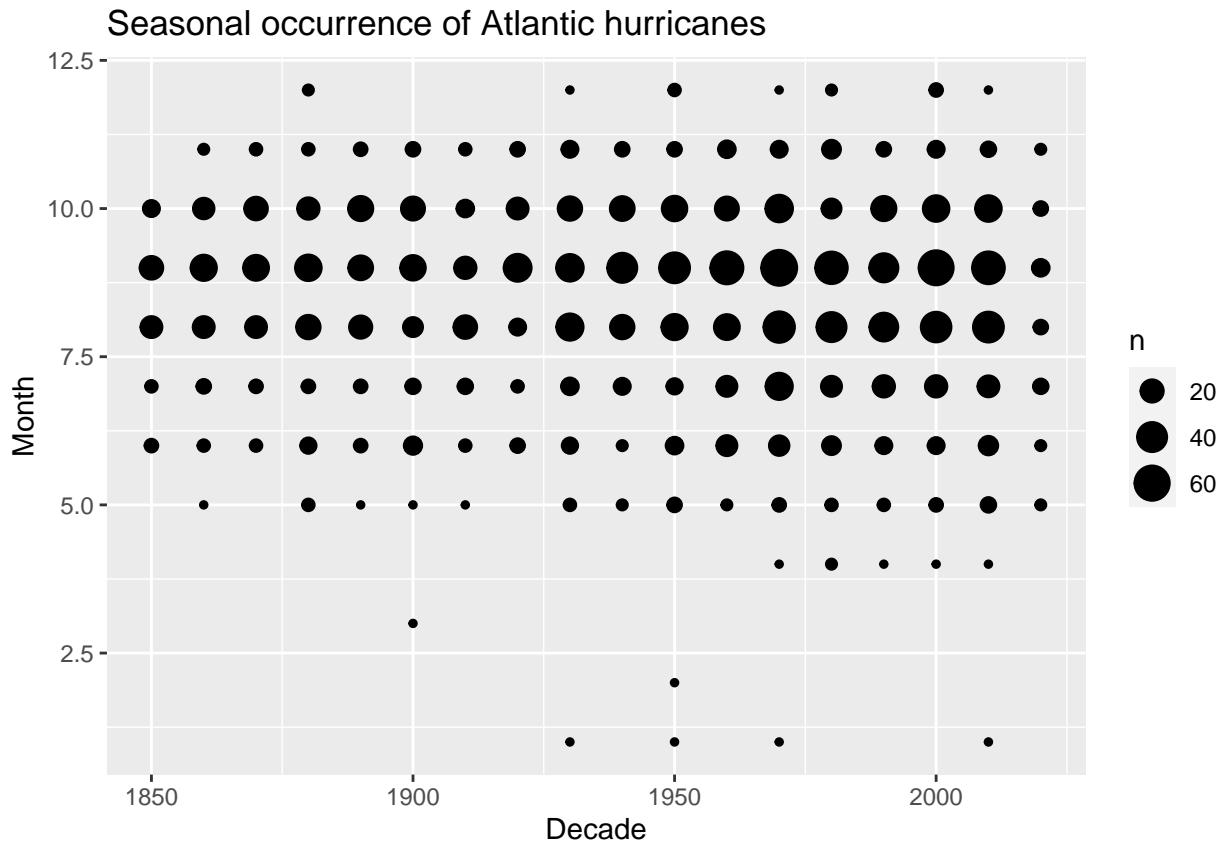


(e) Next, add a “decade” column to the `hurrs` dataframe. (You can calculate `decade` from `year` in one line of code: Google it if you have never seen this rounding trick.)

```
hurrs %>% mutate(decade = floor(year/10)*10) -> hurrs
```

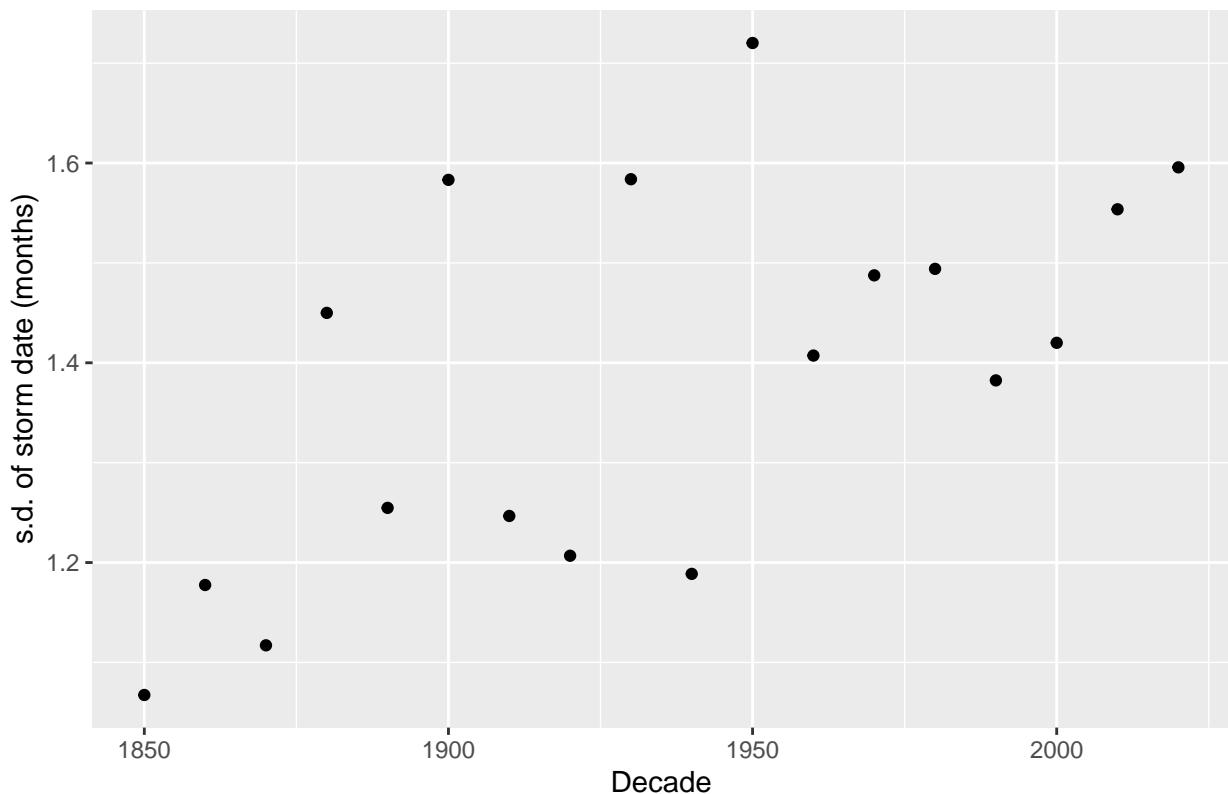
(f) Finally, make a plot that shows whether “hurricane season” is expanding into earlier and later months of the year over time. (How you do this is up to you. One approach would be to calculate standard deviation or other quantiles for each decade. Another would be to investigate `geom_count`. There are many possibilities.)

```
# the geom_count approach:  
hurrs %>% group_by(id) %>% summarise(decade=first(decade),month=first(month)) -> df  
ggplot(df) + geom_count(aes(x=decade,y=month)) +  
  labs(x="Decade",y="Month",title="Seasonal occurrence of Atlantic hurricanes")
```



```
# the s.d. approach:
hurrs %>% group_by(id) %>% summarise(decade=first(decade),month=first(month)) %>%
  group_by(decade) %>% summarise(month_sd = sd(month)) -> df
ggplot(df) + geom_point(aes(x=decade,y=month_sd)) +
  labs(x="Decade",y="s.d. of storm date (months)",
       title="Seasonal occurrence of Atlantic hurricanes")
```

Seasonal occurrence of Atlantic hurricanes



```
# Yes, the seasonal window appears to be getting wider (as ocean temperature increases,  
# it now sometimes reaches the level of heat input to the atmosphere that can  
# generate a hurricane even in winter or spring). However, the uneven data coverage  
# described in part b means that a proper statistical test of this hypothesis would have  
# to be quite carefully designed.
```