individual assignment

Zhen Chen

1. Data Set Overview

1.1 Data source and context

I selected the dataset " **Birds Observed at Feeding Stations in USA**" sourced from the "TidyTuesday project" (tidytuesday/data/2023/2023-01-10 at main · rfordata-science/tidytuesday · GitHub). This dataset records bird species observed at feeding stations across the United States. The data originates from the **FeederWatch program**, a citizen science initiative managed by the **Cornell Lab of Ornithology and Birds Canada**. The program collects reports from thousands of volunteers during the winter months (November to April) to help researchers monitor bird population and trends over time.

1.2 Key variables

This study utilizes two datasets:

- **PFW_2021_public.csv**: Contains survey site variable (e.g. ID), spatial variables (e.g. coordinates, regional abbreviations), species-based variables (species code).
- **PFW_count_site_data_public.csv**: Record the information of survey sites, such as the characteristics of survey site and the way that citizen scientists to attract birds.

The objective of our project is to analyze and visualize bird observation at feeding stations in the USA using data wrangling e.g tidyr and dplyr and visualization e.g ggplot2.

To achieve these objectives, I selected the following key variables which include **temporal variables**, **spatial variables** and **species-based variables** from PFW_2021_public.csv Some variables such as "entry_technique","Data_Entry_Method","PROJ_PERIOD_ID" are not relevant for our analysis as they does not impact our observations, they only describe how data were entered and recorded.

In the second dataset "PFW_count_site_data_public.csv", I focused on **yard type** as a key variable. Although the dataset contains mutiple site-based variables (e.g habit types), this study specifically exmaines the influence of yead type on bird observations. The information of key variables are shown as follows:

Variable	Class	Description
loca_id	character	Unique identifier for each survey site
latitude	double	Latitude in decimal degrees for each survey site
longitude	double	Longitude in decimal degrees for each survey site

Variable	Class	Description
subnational1_code	character	Country abbreviation and State or Province abbreviation of each survey site. Note that the files may contain some "XX" locations. These are sites that were incorrectly placed by the user (e.g., site plotted in the ocean.)
Month	double	Month of 1st day of two-day observation period
Year	double	Year of 1st day of two-day observation period
species_code	character	Bird species observed, stored as 6-letter species codes
how_many	double	Maximum number of individuals seen at one time during observation period
effort_hrs_atleast	double	Participant estimate of survey time for each checklist
snow_dep_atleast	double	Participant estimate of minimum snow depth during a checklist
yard_type_xx	double	the type of yard, contains pavement, garden, landsca, woods, desert

1.3 Data Quality Assessment

- **Completeness**: The dataset is well-structured but contains some missing values, such as snow_dep_atleat.
- **Accuracy**: Since the data is sourced from citizen scientists, there might exit observer bias, affecting species count reliability. Additionally, bird observations are not conducted under standardized conditions - some observations last fewer minute while others span several hours. The variation in observation time may introduce bias, as longer observation times may result in higher recorded birds counts.
- **Timeliness**: The data is collected annually and it includes records at monthly, and daily level, which allows both long-term and short-term analysis. However, it only covers the period November to April, with no data available for Summer month.

2. Data Handling Section

2.1 Transformation of data (tidyr & dplyr)

2.1.1 Load package and import data

```
# install and load necessary package
library(readr)
library(dplyr)
library(ggplot2)
library(tidyr)
library(patchwork)
```

```
# for spatial visualization
install.packages("stringr",repos = "https://cloud.r-project.org/")
```

```
package 'stringr' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\znen0002\AppData\Local\Temp\RtmpOGhici\downloaded_packages
```

```
library(stringr)
```

```
# load data
PFW_2021_public <- read_csv("PFW_2021_public.csv")
PFW_count_site_data_public_2021 <-
 ↪  read_csv("PFW_count_site_data_public_2021.csv")
View(PFW_2021_public)
View(PFW_count_site_data_public_2021)
```

2.1.2 Select key variables

```
# combine two data set by loc_id variable
# there is no year variable in PFW_count_site_data_public_2021,so we extract
 ↪  year from the variable "proj_period_id by using "str_extract"

df <- PFW_2021_public %>% mutate(Year = as.character(Year)) %>%
```

```
  inner_join(PFW_count_site_data_public_2021 %>% mutate(Year =
   ↪  str_extract(proj_period_id
                                                                    ,
 ↪  "\\d{4}")) %>%
               filter(proj_period_id %in% c("PFW_2020", "PFW_2021"))
             # there are other years data in PFW_count_site_data_public_2021,
              ↪  so we filter the data in 2020 and 2021
                                    %>% select (Year,loc_id,
                                     ↪  yard_type_pavement,
                                     ↪  yard_type_garden,
                                     ↪  yard_type_landsca, yard_type_woods,
                                     ↪  yard_type_desert), by =
                                     ↪  c("Year","loc_id")) # use
                                     ↪  inner_join to retain only rows in
                                     ↪  both set


df_key_variable <- df %>% select(loc_id, Year, Month, latitude, longitude,
 ↪  subnational1_code, species_code, how_many, effort_hrs_atleast,
 ↪  snow_dep_atleast,  yard_type_pavement, yard_type_garden,
 ↪  yard_type_landsca, yard_type_woods, yard_type_desert) # select key
 ↪  variables
```

2.1.3 Handle missing values

```
#using na.omit to delete all Na values
df_key_variable <- df_key_variable %>% na.omit(df_key_variable)
```

2.1.4 Convert from wide to long format with pivot_longer

We use the **pivot_longer** function to convert the yard type variable from a wide format to a long format. In the original data, each yard type was stored as a separate column, which made it difficult to filter, group, and summarize the different yard types, converting to long format can simplify analysis.

```
# convert for variable "yard_type"
# use pivot_longer to convert data from wide to long format

df_long <- df_key_variable %>%
  pivot_longer(
    cols = starts_with("yard_type"),   # select all variables starts with
     ↪  yard_type_*
    names_to = "yard_type",   # new column name
    values_to = "type"          # values for new column
  ) %>% mutate(yard_type = gsub("yard_type_", "", yard_type)) # remove prefix
```

2.1.5 Group_by + summarize operations

- Understand bird observations trend over time. I am interested in how bird observations change over time, I aggregated bird observations by year and month, which allows for trend analysis.
- Spatial distribution of bird observations. I summarized bird observations at the province level, which supports the analysis of the regional distribution of bird observations
- Filter for top 10 species and provinces analysis. I selected the most frequently observed species per year and the top 10 provinces with the highest bird observations. These enables clear comparative bar plots.
- Species-specific patterns. I summarize the total bird observations of each species, which facilities the species-wise comparisons.

```
# check the distinct values for certain variables
n_distinct(df_long$loc_id)
```

```
[1] 11287
```

```
n_distinct(df_long$yard_type)
```

```
[1] 5
```

```
n_distinct(df_long$Month)
```

```
[1] 6
```

```
n_distinct(df_long$species_code)
```

```
[1] 322
```

```
n_distinct(df_long$effort_hrs_atleast)
```

```
[1] 4
```

```
# summarize total bird observation every year
yearly_observations <- df_long %>% group_by(Year) %>%
  summarize(total_birds_per_year = sum(how_many))
print(yearly_observations)
```

```
# A tibble: 2 x 2
  Year  total_birds_per_year
  <chr>                <dbl>
1 2020                299855
2 2021                941875
```

```
# summarize total bird observations of each month

monthly_observations <- df_long %>% group_by(Month) %>%
  summarize(total_birds_per_month = sum(how_many))
print(monthly_observations)
```

```
# A tibble: 6 x 2
  Month total_birds_per_month
  <dbl>                 <dbl>
1     1                314350
2     2                276420
3     3                219925
4     4                131180
5    11                113720
6    12                186135
```

```r
# summarize total observations of each bird species
species_observations <- df_long %>% group_by(species_code, Year) %>%
  # group by bird species
  summarize(total_bird_per_species = sum(how_many))
# count bird observations of each species
head(species_observations)
```

```
# A tibble: 6 x 3
# Groups:   species_code [3]
  species_code Year  total_bird_per_species
  <chr>        <chr>                  <dbl>
1 abetow       2020                      60
2 abetow       2021                     140
3 accipi       2020                      10
4 accipi       2021                      20
5 acowoo       2020                      80
```

```
6 acowoo       2021                    200
```

```r
# filter out top 10 bird species with highest bird observations

top_10_species <- species_observations %>%
  group_by(Year) %>%
  # select within group first
  slice_max(total_bird_per_species, n = 10)
# select the top 10 species with the highest total bird observation per year

# summarize average bird observations of each minimum snow depth
snow_depth_observations <- df_long %>% filter(type == 1) %>%
  # Only keep rows with type = 1
  group_by(snow_dep_atleast, yard_type) %>%
  # group by minimum snow depth and yard type
  summarize(average_bird_observations = mean(how_many))
# count bird observations of each species
head(snow_depth_observations)
```

```
# A tibble: 6 x 3
# Groups:   snow_dep_atleast [2]
  snow_dep_atleast yard_type average_bird_observations
             <dbl> <chr>                          <dbl>
1                0 desert                          4.82
2                0 garden                          3.56
3                0 landsca                         3.38
```

```
4            0     pavement                      3.12
5            0     woods                         3.41
6            0.001 desert                       11.3
```

```r
# summarize bird observation per year per province
yearly_province_observations <- df_long %>% group_by(Year, subnational1_code)
 ↪  %>% summarize(total_birds_per_year_province = sum(how_many),
 ↪  mean_birds_per_year_province = mean(how_many))

# filter out top 10 provinces with highest bird observations

top_10_yearly_province_observations <- yearly_province_observations %>%
  group_by(Year) %>%  # select within group first
  slice_max(total_birds_per_year_province, n = 10)
# select the top 10 provinces with the highest total bird observation per
 ↪  year
head(yearly_province_observations)
```

```
# A tibble: 6 x 4
# Groups:   Year [1]
  Year  subnational1_code total_birds_per_year_province mean_birds_per_year_pr~1
  <chr> <chr>                                    <dbl>                      <dbl>
1 2020  CA-AB                                     3290                       5.26
2 2020  CA-BC                                     6960                       4.41
3 2020  CA-MB                                     1370                       3.86
4 2020  CA-NB                                     1805                       7.22
```

```
5 2020  CA-NL                                          355                    7.1
6 2020  CA-NS                                         1530                    4.86
# i abbreviated name: 1: mean_birds_per_year_province
```

2.1.6 Use mutate to create new variables

When we categorize season based on month, we found that there is no observations in Summer.

```
## categorize seasons based on month


df_long <- df_long %>% mutate(season = case_when(
  Month %in% c(3, 4, 5)  ~ "Spring",
  Month %in% c(6, 7, 8)  ~ "Summer",
  Month %in% c(9, 10, 11) ~ "Fall",
  Month %in% c(12, 1, 2) ~ "Winter",
))
```

3. Visualizations Section

3.1 Individual figures with explanations

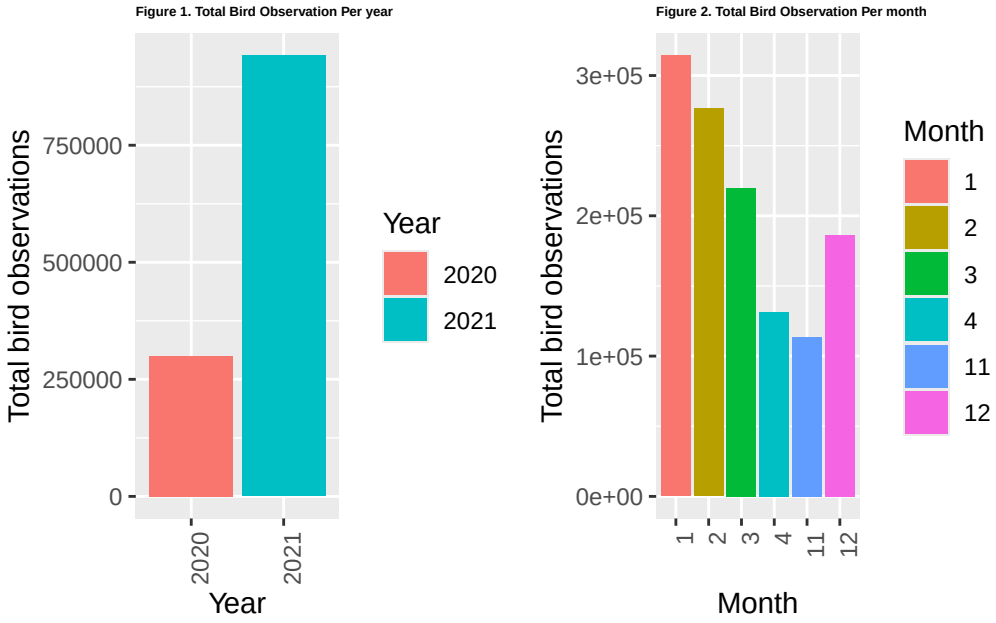
3.1.1 Create two bar plot to analyze the bird observations over time

We created two bar plots to analyze yearly and monthly trends in bird observations. As shown in the figure 1, the number of bird observations is much higher than that in 2020. When we

look into the figure 2, we found that the bird observations is highest in Jan while it is lowest in November. Regarding the color choice, we select distinct color for each year and month, which helps differentiate categorical data clearly and the color is very contrasting, which ensure th readability for colorblind users. AdditionallyAdditionally, we put the yearly and monthly figures together, which can provide both long-term and short-term patterns within a single visualization.

```
# total bird observation every year


# convert year from numeric to factor
yearly_observations$Year <- as.factor(yearly_observations$Year)


a <- ggplot(data = yearly_observations , mapping = aes (x = Year,
                                                        y =
                                                         ↪  total_birds_per_year,
                                                         ↪
                                                        fill = Year)) +
  geom_bar(stat = "identity") +
  #define the stats, as we have provided the y value, we need to use stat =
   ↪  "identity"
  labs(x = "Year", # x axis title
       y = "Total bird observations", # y axis title
       title = "Figure 1. Total Bird Observation Per year"
       # main title of figure
       ) + #title of legend
  theme(plot.title = element_text(size = 5, face = "bold"),
    axis.text.x = element_text(angle = 90, hjust = 1))
```

```r
# summarize total observations of each month

# convert month from numeric to factor
monthly_observations$Month <- as.factor(monthly_observations$Month)

b <- ggplot(data = monthly_observations , mapping = aes (x = Month, y =
 ↪  total_birds_per_month, fill = Month)) +
  geom_bar(stat = "identity") +
  #define the stats, as we have provided the y value, we need to use stat =
   ↪  "identity"
  labs(x = "Month", # x axis title
       y = "Total bird observations",
       # y axis title
       title = "Figure 2. Total Bird Observation Per month"
       # main title of figure
       ) + #title of legend
  theme(plot.title = element_text(size = 5, face = "bold"),
    axis.text.x = element_text(angle = 90, hjust = 1))

# combine plots

patchwork::wrap_plots(a,b)
```



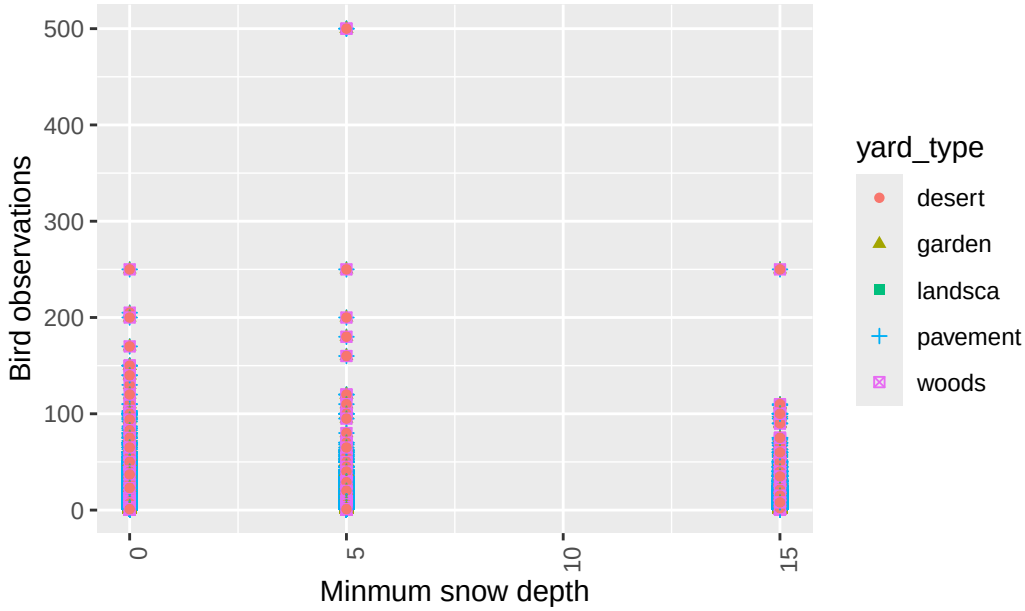
3.1.2 Plot a point plot containing snow_depth and bird observations

I tried to display the relationship between snow depth and the number of bird observations. Meanwhile, i use different colors and shapes to represent various types, but the figure seems look not so good, because many points are overlapped.

```
ggplot(data = df_long) + geom_point(aes(x = snow_dep_atleast, y= how_many,
                                        shape = yard_type, color=yard_type))
                                         ↪  +
  labs(x = "Minmum snow depth", # x axis title
       y = "Bird observations", # y axis title
       title = "Figure 3. Bird Observation vs. Minmum Snow Depth vs. Yard
        ↪  Type"
       # main title of figure
```

```
    ) + #title of legend
theme(plot.title = element_text(size = 10, face = "bold"),
  axis.text.x = element_text(angle = 90, hjust = 1))
```

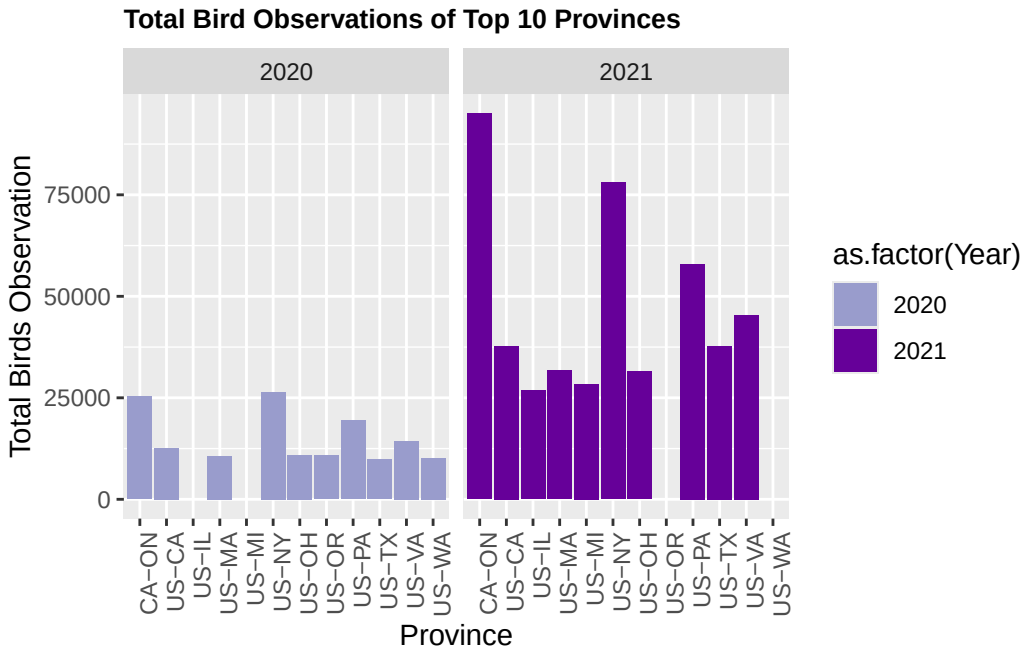
Figure 3. Bird Observation vs. Minmum Snow Depth vs. Yard Type



3.1.3 Plot top 10 provinces with the highest bird observations

We display regional differences in bird observations across provinces and we use faceting to provide a clear, per-year breakdown of the data. As shown in the figure, US-NY has the highest bird observations and it is followed by CA-ON. In 2021, CA-ON has the highest bird observations while US-IL has the lowest.

```r
# there are too many provinces in our dataset, we filter top 10 provinces and
 ↪  top 10 bird species


# top 10 province


ggplot(data = top_10_yearly_province_observations, mapping = aes(x =
 ↪  subnational1_code, y = total_birds_per_year_province, fill =
 ↪  as.factor(Year))) +
  geom_col() +
  facet_wrap(~Year) + # facet in two dimension
scale_fill_manual(values = c("2020" = "#999ccc", "2021" = "#660099"))+
  labs(title = "Total Bird Observations of Top 10 Provinces",
       x = "Province",
       y = "Total Birds Observation") +
  theme(plot.title = element_text(size = 10, face = "bold"),
    axis.text.x = element_text(angle = 90, hjust = 1))
```

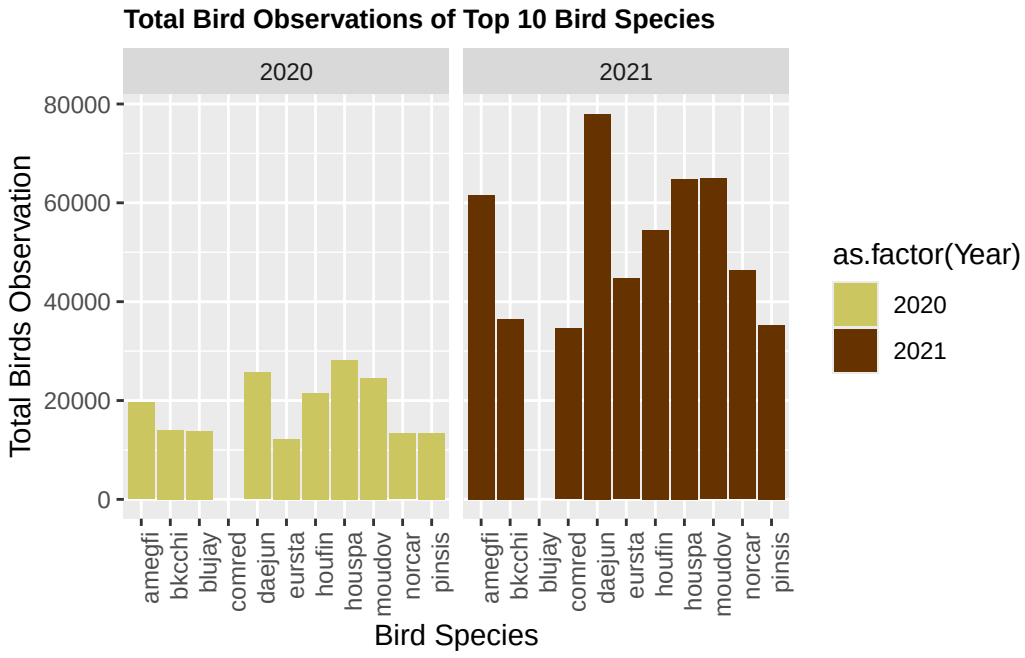


3.1.4 Plot top 10 provinces with the highest bird observations

We display species differences in bird observations and we use faceting to provide a clear, per-year breakdown of the data. As shown in the figure, the overall bird observations of top 10 species in 2020 is much lower than that in 2021. In 2020, houspa has the highest bird observations while comred is the highest in 2021.

```
# top 10 bird species


ggplot(data = top_10_species, mapping = aes(x = species_code, y =
 ↪  total_bird_per_species, fill = as.factor(Year))) + geom_col() +
  facet_wrap(~Year)+
  # facet in two dimension
```

```r
scale_fill_manual(values = c("2020" = "#CCC660", "2021" = "#663300"))+
  labs(title = "Total Bird Observations of Top 10 Bird Species",
       x = "Bird Species",
       y = "Total Birds Observation") +
  theme(
    plot.title = element_text(size = 10, face = "bold"),
    axis.text.x = element_text(angle = 90, hjust = 1))
```



3.2 One spatial visualization using R

3.2.1 Install and load necessary packages

```
library(sf) # vector data
install.packages("geodata", repos = "https://cloud.r-project.org/")
```

```
package 'geodata' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\znen0002\AppData\Local\Temp\RtmpOGhici\downloaded_packages
```

```
library(geodata) # vector of roads
install.packages("rnaturalearthhires", repos =
 ↪  "https://ropensci.r-universe.dev")
```

```
package 'rnaturalearthhires' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\znen0002\AppData\Local\Temp\RtmpOGhici\downloaded_packages
```

```
library(rnaturalearth) # shape of usa
library(ggspatial) # plot spatial in ggplot
pcks <- list("dplyr",
             "raster",    # used for raster data
```

```
             "terra",     # used for raster data
             "sf",        # used for handling sppatial data
             "ggspatial", # spatial for ggplot2
             "tidyverse")
sapply(pcks, require, char = TRUE) ## check whether these packages are loaded
 ↪  or not
```

```
[1] TRUE TRUE TRUE TRUE TRUE TRUE
```

3.2.2 Download vector data

We downloaded the vector data using "rnaturalearth" and save the shp locally. It can also be download from Natural Earth (https://www.naturalearthdata.com/downloads/)

```
## Download shapfile for the USA
usa <- ne_countries(scale = 10, country = "United States of America",
 ↪  returnclass = "sf")
# Check projection
sf::st_crs(usa)$proj4string  # Check if it's unprojected (WGS84)
```

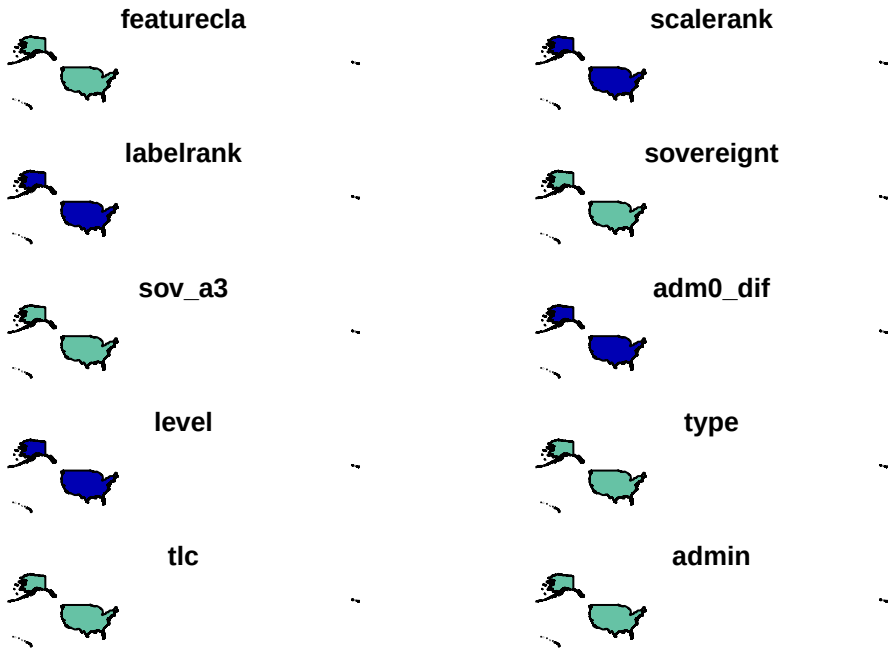
```
[1] "+proj=longlat +datum=WGS84 +no_defs"
```

```
class(usa)
```

```
[1] "sf"         "data.frame"
```

```
plot(usa)
```

```
Warning: plotting the first 10 out of 168 attributes; use max.plot = 168 to
plot all
```




```
plot(usa[,1]) # Plot the unprojected USA map
```


```
# Transform projection to a common US projection (Albers Equal Area)
usa_proj <- sf::st_transform(usa, crs = 5070)  # EPSG:5070 is Albers Equal
 ↪  Area
```

```
## load and transform water bodies for the USA


lakes <- ne_download(scale = 10, type = "lakes", category = "physical",
 ↪  returnclass = "sf")
```

```
Reading layer `ne_10m_lakes' from data source
  `C:\Users\znen0002\AppData\Local\Temp\RtmpOGhici\ne_10m_lakes.shp'
  using driver `ESRI Shapefile'
Simple feature collection with 1355 features and 41 fields
```

```
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -165.9656 ymin: -50.66967 xmax: 177.1544 ymax: 81.95521
Geodetic CRS:  WGS 84
```

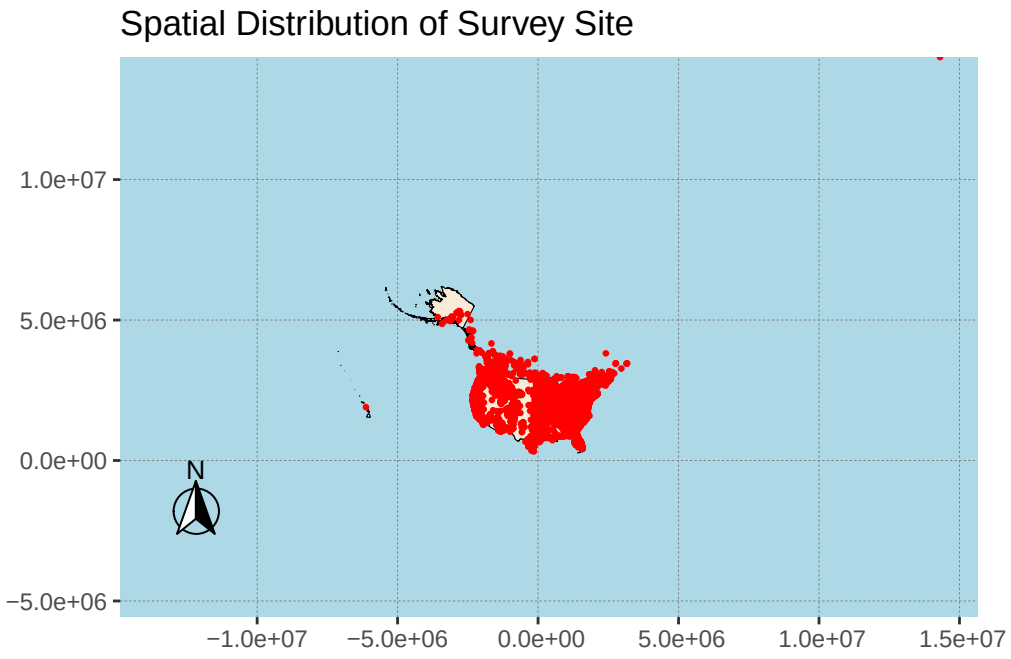
```r
lakes_proj <- st_transform(lakes, crs = 5070)


## extract coordinators for observational data
df_long <- as.data.frame(df_long)
observations <- df_long %>% dplyr::select(loc_id, latitude, longitude) %>%
  # extract coordinators for each loc_id
  distinct(loc_id,.keep_all = TRUE)
# remove duplicate and keep the first record of each loc_id



## Convert observational data to an SF object
observations_sf <- st_as_sf(observations, coords = c("longitude",
 ↪  "latitude"), crs = 4326)
observations_proj <- st_transform(observations_sf, crs = 5070)
# Transform to match map
```

3.2.3 Plot observations on the map using ggplot2

As shown in the map, the observations concentrated in the eastern and central regions. Pew observations appear in the western USA and Alaska.

```
ggplot() +
  # Plot USA boundaries
  geom_sf(data = usa_proj, fill = "antiquewhite", color = "black") +

  # Plot lakes in light blue
  geom_sf(data = lakes_proj, fill = "lightblue", color = NA) +

  # Add observation points
  geom_sf(data = observations_proj, color = "red", size = 0.5) +

  # Add title
  ggtitle("Spatial Distribution of Survey Site") +

  # Add north arrow and scale bar
  annotation_north_arrow(location = "bl", which_north = "true",
                         height = unit(1, "cm"), width = unit(1, "cm"),
                         pad_x = unit(0.5, "cm"), pad_y = unit(1, "cm"),
                         style = north_arrow_fancy_orienteering, rotation =
                          ↪  0) +

  # Keep new projection for ggplot, make sure ggplot uses the same projected
   ↪  coordinate system (EPSG: 5070) as the transformed spatial data, rather
   ↪  than defaulting back to WGS84.
  coord_sf(crs = st_crs(5070), datum = sf::st_crs(5070), expand = FALSE) +

  # Customize background and grid lines
```

```
theme(
  panel.grid.major = element_line(color = gray(.5), linetype = "dashed",
   ↪  linewidth = 0.1),
  panel.background = element_rect(fill = "lightblue")
)
```



4. Reflection Section

4.1 New skills acquired

- **Data wrangling.** I developed news skills of using tidyverse and dplyr to handle data, such as how to use select and filter functions to extract data, use group_by and sum-

marize to categorize data and analyze data, and use pivot_longer to convert wide data format to longer format.

- **Visualization analysis.** I gained experience in creating different mapping and spatial visualization.

4.2 Challenges encountered and solutions

- **Fix errors in code execution.** It is very common to encounter errors during data manipulation. Usually I ask help from AI.
- **Choose appropriate plot type.** I struggled with selecting the most suitable visualization for the dataset to convey insights effectively. This might be related to the experience and reading. reading and reviewing relevant studies can be a solution for improving understand of different plots
- **Generate map in R.** I followed the tutorial to create a map but it is still very struggling, I need more practice to be more familiar with different aesthetics and to create more informative maps.

4.3 Areas for future improvement

- Enhance ability to choose select the most appropriate plot type
- Enhance ability to create more informative maps