strawberry

Bingtian Ye

2023-10-11

##read data

```
strawberry=read.csv("strawberry.csv",header = T)
```

##Data Overview

```
summary(strawberry)
```

```
##    Program              Year         Period          Week.Ending
##  Length:4314        Min.   :2016   Length:4314        Mode:logical
##  Class :character   1st Qu.:2016   Class :character   NA's:4314
##  Mode  :character   Median :2018   Mode  :character
##                     Mean   :2018
##                     3rd Qu.:2019
##                     Max.   :2022
##
##   Geo.Level            State             State.ANSI    Ag.District
##  Length:4314        Length:4314        Min.   : 1.00   Mode:logical
##  Class :character   Class :character   1st Qu.: 6.00   NA's:4314
##  Mode  :character   Mode  :character   Median :12.00
##                                        Mean   :16.46
##                                        3rd Qu.:21.00
##                                        Max.   :55.00
##                                        NA's   :86
##  Ag.District.Code  County        County.ANSI    Zip.Code        Region
##  Mode:logical     Mode:logical   Mode:logical   Mode:logical   Mode:logical
##  NA's:4314        NA's:4314      NA's:4314      NA's:4314      NA's:4314
##
##
##
##
##
##  watershed_code Watershed       Commodity          Data.Item
##  Min.   :0      Mode:logical   Length:4314        Length:4314
##  1st Qu.:0      NA's:4314      Class :character   Class :character
##  Median :0                     Mode  :character   Mode  :character
##  Mean   :0
##  3rd Qu.:0
##  Max.   :0
##
##     Domain          Domain.Category       Value              CV....
##  Length:4314        Length:4314        Length:4314        Length:4314
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
```

```
head(strawberry)
```

```
##   Program Year Period Week.Ending Geo.Level  State State.ANSI Ag.District
## 1  CENSUS 2021   YEAR          NA     STATE ALASKA          2          NA
## 2  CENSUS 2021   YEAR          NA     STATE ALASKA          2          NA
## 3  CENSUS 2021   YEAR          NA     STATE ALASKA          2          NA
## 4  CENSUS 2021   YEAR          NA     STATE ALASKA          2          NA
## 5  CENSUS 2021   YEAR          NA     STATE ALASKA          2          NA
## 6  CENSUS 2021   YEAR          NA     STATE ALASKA          2          NA
##   Ag.District.Code County County.ANSI Zip.Code Region watershed_code Watershed
## 1               NA     NA          NA       NA     NA              0        NA
## 2               NA     NA          NA       NA     NA              0        NA
## 3               NA     NA          NA       NA     NA              0        NA
## 4               NA     NA          NA       NA     NA              0        NA
## 5               NA     NA          NA       NA     NA              0        NA
## 6               NA     NA          NA       NA     NA              0        NA
##      Commodity                                                   Data.Item
## 1 STRAWBERRIES               STRAWBERRIES, ORGANIC - OPERATIONS WITH SALES
## 2 STRAWBERRIES         STRAWBERRIES, ORGANIC - PRODUCTION, MEASURED IN CWT
## 3 STRAWBERRIES                STRAWBERRIES, ORGANIC - SALES, MEASURED IN $
## 4 STRAWBERRIES              STRAWBERRIES, ORGANIC - SALES, MEASURED IN CWT
## 5 STRAWBERRIES STRAWBERRIES, ORGANIC, FRESH MARKET - OPERATIONS WITH SALES
## 6 STRAWBERRIES  STRAWBERRIES, ORGANIC, FRESH MARKET - SALES, MEASURED IN $
##           Domain                      Domain.Category Value CV....
## 1 ORGANIC STATUS ORGANIC STATUS: (NOP USDA CERTIFIED)     2    (H)
## 2 ORGANIC STATUS ORGANIC STATUS: (NOP USDA CERTIFIED)   (D)    (D)
## 3 ORGANIC STATUS ORGANIC STATUS: (NOP USDA CERTIFIED)   (D)    (D)
## 4 ORGANIC STATUS ORGANIC STATUS: (NOP USDA CERTIFIED)   (D)    (D)
## 5 ORGANIC STATUS ORGANIC STATUS: (NOP USDA CERTIFIED)     2    (H)
## 6 ORGANIC STATUS ORGANIC STATUS: (NOP USDA CERTIFIED)   (D)    (D)
```

##Data preparing ###Remove columns with a single value in all columns (from giving qmd)

```
#define the function
drop_one_value_col <- function(df){
drop <- NULL
for(i in 1:dim(df)[2]){ #1:colume number
if((df |> distinct(df[,i]) |> count()) == 1){ #if only have one value, add i in drop
drop = c(drop, i)
} }

if(is.null(drop)){return("none")}else{

   print("Columns dropped:")
   print(colnames(df)[drop])
   strawberry <- df[, -1*drop]
}
}
#use function
strawberry_dropOneValue=drop_one_value_col(strawberry)
```

```
## [1] "Columns dropped:"
##  [1] "Week.Ending"      "Geo.Level"        "Ag.District"      "Ag.District.Code"
##  [5] "County"           "County.ANSI"      "Zip.Code"         "Region"
##  [9] "watershed_code"   "Watershed"        "Commodity"
```

```
head(strawberry_dropOneValue)
```

```
##   Program Year Period  State State.ANSI
## 1  CENSUS 2021   YEAR ALASKA          2
## 2  CENSUS 2021   YEAR ALASKA          2
## 3  CENSUS 2021   YEAR ALASKA          2
## 4  CENSUS 2021   YEAR ALASKA          2
## 5  CENSUS 2021   YEAR ALASKA          2
## 6  CENSUS 2021   YEAR ALASKA          2
##                                                       Data.Item         Domain
## 1              STRAWBERRIES, ORGANIC - OPERATIONS WITH SALES ORGANIC STATUS
## 2        STRAWBERRIES, ORGANIC - PRODUCTION, MEASURED IN CWT ORGANIC STATUS
## 3               STRAWBERRIES, ORGANIC - SALES, MEASURED IN $ ORGANIC STATUS
## 4             STRAWBERRIES, ORGANIC - SALES, MEASURED IN CWT ORGANIC STATUS
## 5 STRAWBERRIES, ORGANIC, FRESH MARKET - OPERATIONS WITH SALES ORGANIC STATUS
## 6  STRAWBERRIES, ORGANIC, FRESH MARKET - SALES, MEASURED IN $ ORGANIC STATUS
##                         Domain.Category Value CV....
## 1 ORGANIC STATUS: (NOP USDA CERTIFIED)     2    (H)
## 2 ORGANIC STATUS: (NOP USDA CERTIFIED)   (D)    (D)
## 3 ORGANIC STATUS: (NOP USDA CERTIFIED)   (D)    (D)
## 4 ORGANIC STATUS: (NOP USDA CERTIFIED)   (D)    (D)
## 5 ORGANIC STATUS: (NOP USDA CERTIFIED)     2    (H)
## 6 ORGANIC STATUS: (NOP USDA CERTIFIED)   (D)    (D)
```

###Overview the value of each colume.

```
value_unique=lapply(strawberry_dropOneValue, function(x) head(unique(x), 5))
value_unique
```

```
## $Program
## [1] "CENSUS" "SURVEY"
## 
## $Year
## [1] 2021 2019 2016 2022 2020
## 
## $Period
## [1] "YEAR"                "MARKETING YEAR"      "YEAR - AUG FORECAST"
## 
## $State
## [1] "ALASKA"      "CALIFORNIA"  "CONNECTICUT" "FLORIDA"     "GEORGIA"
## 
## $State.ANSI
## [1]  2  6  9 12 13
## 
## $Data.Item
## [1] "STRAWBERRIES, ORGANIC - OPERATIONS WITH SALES"
## [2] "STRAWBERRIES, ORGANIC - PRODUCTION, MEASURED IN CWT"
## [3] "STRAWBERRIES, ORGANIC - SALES, MEASURED IN $"
## [4] "STRAWBERRIES, ORGANIC - SALES, MEASURED IN CWT"
## [5] "STRAWBERRIES, ORGANIC, FRESH MARKET - OPERATIONS WITH SALES"
## 
## $Domain
## [1] "ORGANIC STATUS"        "TOTAL"                 "CHEMICAL, FUNGICIDE"
## [4] "CHEMICAL, HERBICIDE"   "CHEMICAL, INSECTICIDE"
## 
## $Domain.Category
## [1] "ORGANIC STATUS: (NOP USDA CERTIFIED)"
## [2] "NOT SPECIFIED"
## [3] "CHEMICAL, FUNGICIDE: (AZOXYSTROBIN = 128810)"
## [4] "CHEMICAL, FUNGICIDE: (BACILLUS AMYLOLIQUEFAC F727 = 16489)"
## [5] "CHEMICAL, FUNGICIDE: (BACILLUS AMYLOLIQUEFACIENS MBI 600 = 129082)"
## 
## $Value
## [1] "2"           " (D)"        "142"         "1,413,251"   "311,784,980"
## 
## $CV....
## [1] "(H)"  "(D)"  "19.2" "51.6" "46.0"
```

###Data processing of Value and CV…

```
#the value (D) means: Withheld to avoid disclosing data for individual operations.
#the value (H) means: Coefficient of variation or generalized coefficient of variation is greater than or eq
ual to 99.95 percent or the standard error is greater than or equal to 99.95 percent of the mean
straw_na <- strawberry_dropOneValue |> filter(CV....=="(H)"|CV....=="(D)"|Value=="(D)")
vals=strawberry_dropOneValue$Value
vals=sub(",","",vals)
vals=sub('""',"",vals)
vals=as.numeric(vals)
strawberry_dropOneValue["Value"]=vals
vals=strawberry_dropOneValue$CV....
vals=as.numeric(vals)
strawberry_dropOneValue["CV...."]=vals
```

###Classified by program

```
stb_census <- strawberry_dropOneValue |> filter(Program=="CENSUS")

## ## filter rows of California data from the SURVEY data
stb_survey <- strawberry_dropOneValue |> filter(Program=="SURVEY")

census_col <- colnames(stb_census)

survey_col <- colnames(stb_survey)
```

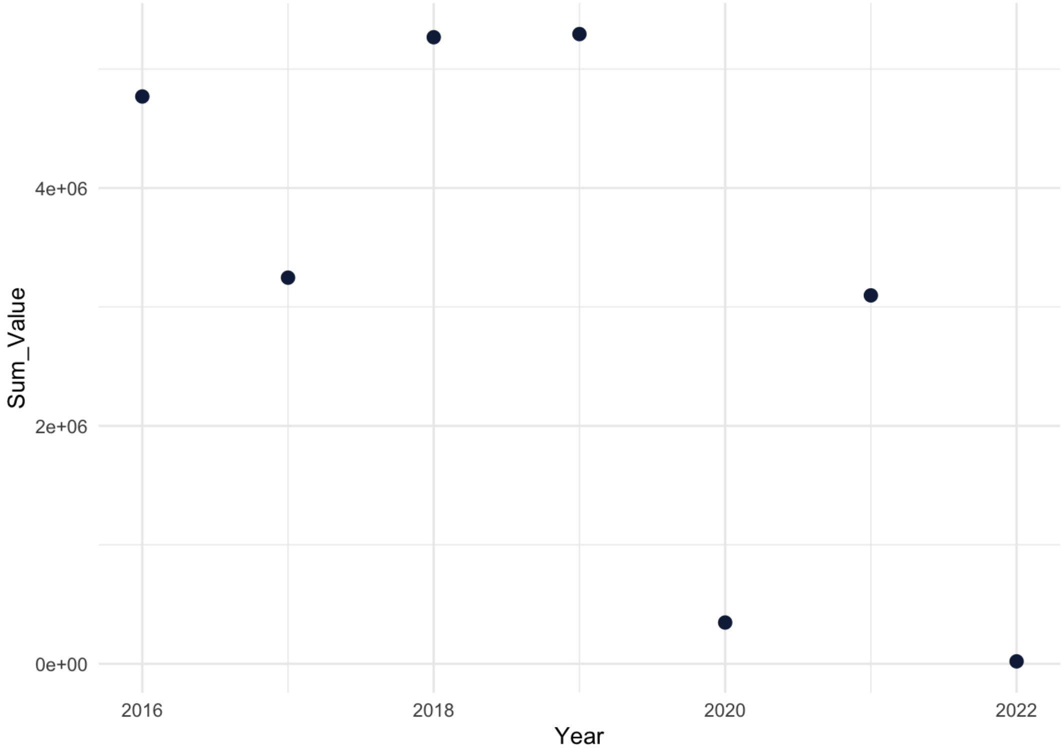
```
stb_census %>%
  group_by(State) %>%
  summarise(Total_Value = sum(Value, na.rm = TRUE))
```

```
## # A tibble: 46 × 2
##    State       Total_Value
##    <chr>             <dbl>
##  1 ALABAMA               6
##  2 ALASKA                4
##  3 ARIZONA               6
##  4 ARKANSAS              2
##  5 CALIFORNIA       444002
##  6 COLORADO          62236
##  7 CONNECTICUT      254148
##  8 FLORIDA          410406
##  9 GEORGIA           28065
## 10 IDAHO            205128
## # ℹ 36 more rows
```

```
stb_survey %>%
  group_by(State) %>%
  summarise(Total_Value = sum(Value, na.rm = TRUE))
```

```
## # A tibble: 11 × 2
##    State          Total_Value
##    <chr>                <dbl>
##  1 CALIFORNIA       11639437.
##  2 FLORIDA           3859748.
##  3 MICHIGAN                0
##  4 NEW YORK           422903
##  5 NORTH CAROLINA    2290141.
##  6 OHIO                    0
##  7 OREGON            2084918.
##  8 OTHER STATES       591108.
##  9 PENNSYLVANIA            0
## 10 WASHINGTON        1154029.
## 11 WISCONSIN               0
```

```
year_census <- stb_census %>%
  group_by(Year) %>%
  summarise(Sum_Value = sum(Value, na.rm = TRUE))
year_survey <- stb_survey %>%
  group_by(Year) %>%
  summarise(Sum_Value = sum(Value, na.rm = TRUE))
ggplot(year_survey) +
  aes(x = Year, y = Sum_Value) +
  geom_point(shape = "circle", size = 2.5, colour = "#112446") +
  theme_minimal()
```



Extract market names and chemical substances and their codes

```
stb_census <- stb_census %>%
  mutate(Data.Item = ifelse(
    str_detect(Data.Item, "MEASURED IN"),
    str_extract(Data.Item, "(?<=MEASURED IN ).*"),
    ifelse(str_detect(Data.Item, "SALES"), "SALES", Data.Item)
  ))
stb_survey <- stb_survey %>%
  mutate(
    Chemical = if_else(str_detect(Domain.Category, "\\(.*=.*\\)"),
                       str_extract(Domain.Category, "(?<=\\().*?(?=\\=)"),
                       NA_character_),
    Chemical_Code = if_else(str_detect(Domain.Category, "\\(.*=.*\\)"),
                            str_extract(Domain.Category, "(?<=\\=).*?(?=\\))"),
                            NA_character_)
  )
```

```
stb_census=subset(stb_census, !is.na(Value))
stb_survey=subset(stb_survey, !is.na(Value))
library(sf)
```

```
## Linking to GEOS 3.11.0, GDAL 3.5.3, PROJ 9.1.0; sf_use_s2() is TRUE
```

```
library(tools)
library(plotly)
```

```
##
## Attaching package: 'plotly'
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
# average_values <- stb_census %>%
#   group_by(State) %>%
#   summarise(Average_Value = mean(Value, na.rm = TRUE))
us_states <- st_read("https://eric.clst.org/assets/wiki/uploads/Stuff/gz_2010_us_040_00_5m.json")
```

```
## Reading layer `gz_2010_us_040_00_5m' from data source
##   `https://eric.clst.org/assets/wiki/uploads/Stuff/gz_2010_us_040_00_5m.json'
##   using driver `GeoJSON'
## Simple feature collection with 52 features and 5 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -179.1473 ymin: 17.92688 xmax: 179.7785 ymax: 71.35256
## Geodetic CRS:  WGS 84
```

```
capitalize_first <- function(string) {
  paste0(toupper(substr(string, 1, 1)), tolower(substr(string, 2, nchar(string))))
}

# df <- data.frame(State = sapply(average_values$State, capitalize_first),
#                  Value = average_values$Value)
stb_census_money=stb_census|>
  filter(Data.Item=="$")
values <- stb_census_money %>%
  group_by(State,Year) %>%
  summarise(Value = mean(Value, na.rm = TRUE))
```

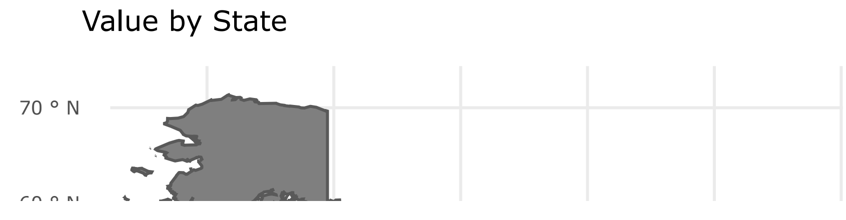
```
## `summarise()` has grouped output by 'State'. You can override using the
## `.groups` argument.
```

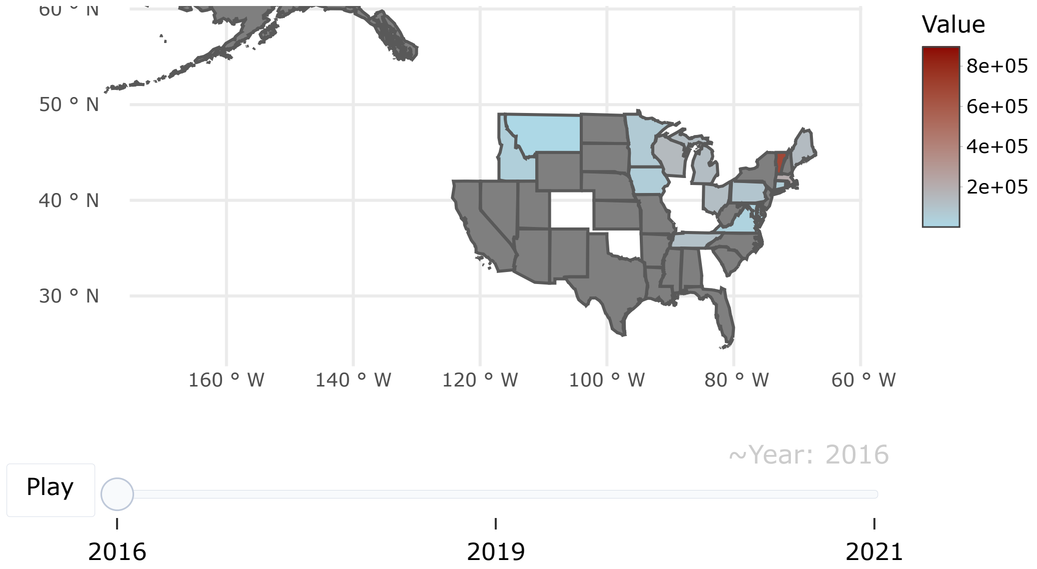
```
values$State<-sapply(values$State, capitalize_first)
merged_data <- left_join(us_states, values, by = c("NAME" = "State"))

p <- ggplot(data = merged_data) +
  geom_sf(aes(fill = Value, frame = Year)) +
  scale_fill_gradient(low = "lightblue", high = "darkred") +
  theme_minimal() +
  labs(title = "Value by State", fill = "Value") +
  coord_sf(xlim = c(-170, -65), ylim = c(25, 72))
```

```
## Warning in layer_sf(geom = GeomSf, data = data, mapping = mapping, stat = stat,
## : Ignoring unknown aesthetics: frame
```

```
plotly_map <- ggplotly(p)
plotly_map
```





```
stb_census_sales=stb_census|>
  filter(Data.Item=="SALES")
values <- stb_census_sales %>%
  group_by(State,Year) %>%
  summarise(Value = mean(Value, na.rm = TRUE))
```

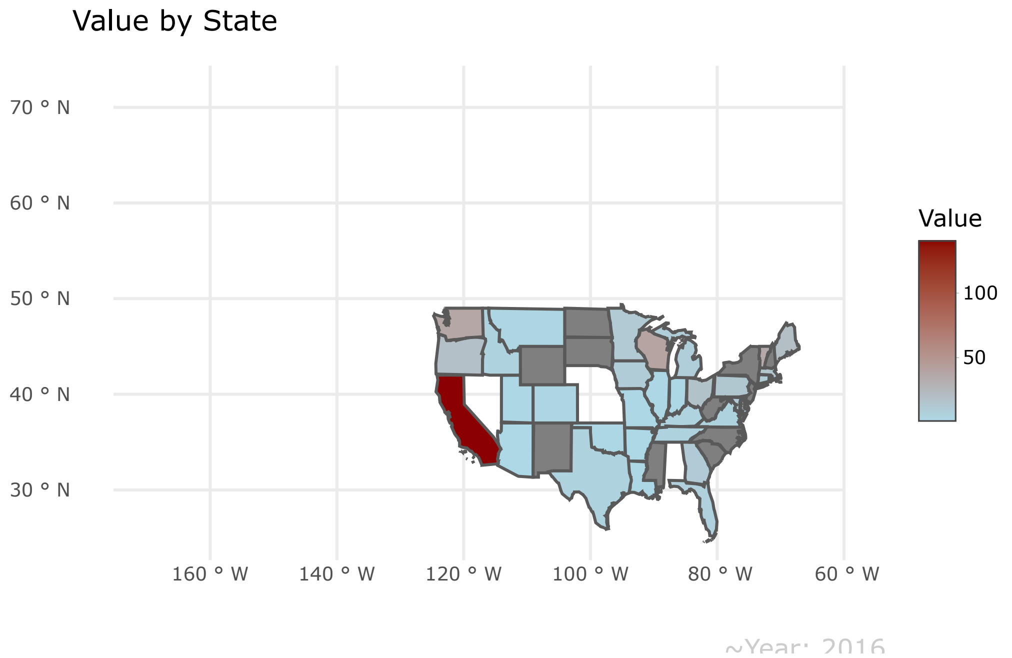
```
## `summarise()` has grouped output by 'State'. You can override using the
## `.groups` argument.
```

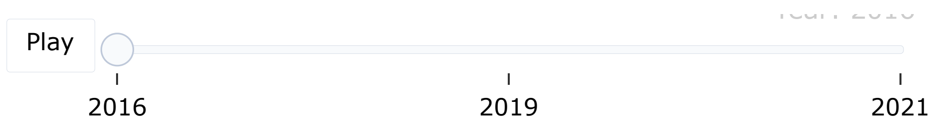
```
values$State<-sapply(values$State, capitalize_first)
merged_data <- left_join(us_states, values, by = c("NAME" = "State"))

p <- ggplot(data = merged_data) +
  geom_sf(aes(fill = Value, frame = Year)) +
  scale_fill_gradient(low = "lightblue", high = "darkred") +
  theme_minimal() +
  labs(title = "Value by State", fill = "Value") +
  coord_sf(xlim = c(-170, -65), ylim = c(25, 72))
```

```
## Warning in layer_sf(geom = GeomSf, data = data, mapping = mapping, stat = stat,
## : Ignoring unknown aesthetics: frame
```

```
plotly_map <- ggplotly(p)
plotly_map
```





```
stb_census_cwt=stb_census|>
  filter(Data.Item=="CWT")
values <- stb_census_cwt %>%
  group_by(State,Year) %>%
  summarise(Value = mean(Value, na.rm = TRUE))
```

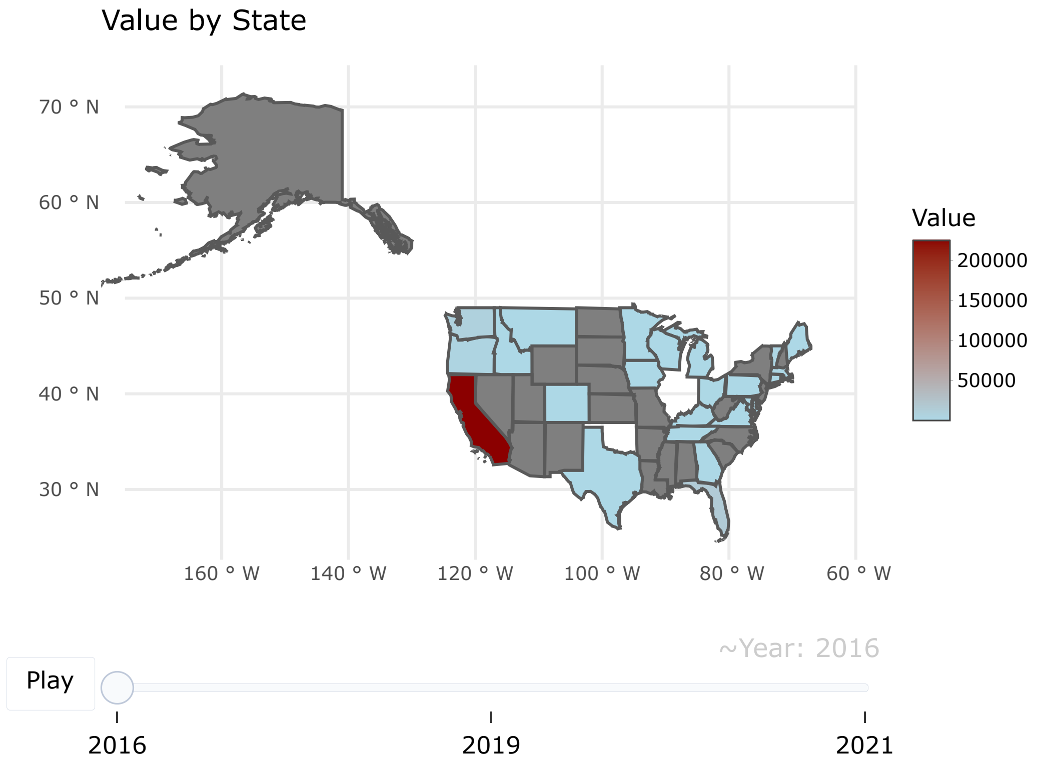
```
## `summarise()` has grouped output by 'State'. You can override using the
## `.groups` argument.
```

```
values$State<-sapply(values$State, capitalize_first)
merged_data <- left_join(us_states, values, by = c("NAME" = "State"))

p <- ggplot(data = merged_data) +
  geom_sf(aes(fill = Value, frame = Year)) +
  scale_fill_gradient(low = "lightblue", high = "darkred") +
  theme_minimal() +
  labs(title = "Value by State", fill = "Value") +
  coord_sf(xlim = c(-170, -65), ylim = c(25, 72))
```

```
## Warning in layer_sf(geom = GeomSf, data = data, mapping = mapping, stat = stat,
## : Ignoring unknown aesthetics: frame
```

```
plotly_map <- ggplotly(p)
plotly_map
```



```
# stb_survey$Chemical_Code_num <- as.numeric(stb_survey$Chemical_Code)
# stb_survey$Chemical_Code_str <- ifelse(is.na(stb_survey$Chemical_Code_num),
#                                         NA,
#                                         sprintf("%06d", stb_survey$Chemical_Code_num))
# library(httr)
# library(jsonlite)
# get_cas <- function(PC){
#     path <- paste0("https://ordspub.epa.gov/ords/pesticides/apprilapi/?q=%7b%22ais%22:%7b%22$instr%22:%22", PC,"%22%7d%7d")
#     r <- GET(url = path)
#     r_text <- content(r, as = "text", encoding = "UTF-8")
#     df <- fromJSON(r_text, flatten = TRUE)
#     df_strwb <- df$items[grepl("Strawberries", df$items$sites, fixed=T),]
#     ais <- df_strwb$ais[1]
#     pattern <- "\\(([^A-Za-z]+)\\/([0-9-]+)\\)"
#     text <- ais
#     matches <- regmatches(text, gregexpr(pattern, text))
#     cas <- sapply(matches, function(x) gsub(".*\\/([0-9-]+)\\)", "\\1", x))
#     if (is.character(cas)) {
#         return(cas[1])
# }
#     else {
#         return("can't find")
# }
# }
# unique_stb=unique(stb_survey$Chemical_Code_str)
# result=numeric()
# k=numeric()
# for(i in 1:length(unique_stb)){
#   result[i]=get_cas(unique_stb[i])
#   k[i]=unique_stb[i]
#   print(result[i])
# }
# data_save=data.frame(k,result)
# write.csv(data_save,"/Users/bingtianye/Desktop/data_save.csv",row.names = F)
```

```
data_save=read.csv("/Users/bingtianye/Desktop/bu_study/MA615 Data Science in R/exercise/data_save",header=T)
data_save$Chemical_Code_num <- as.numeric(data_save$k)
data_save$Chemical_Code_str <- ifelse(is.na(data_save$Chemical_Code_num),
                                         NA,
                                         sprintf("%06d", data_save$Chemical_Code_num))
po=read.csv("/Users/bingtianye/Desktop/bu_study/MA615 Data Science in R/exercise/CAS.csv",header=T)
merged_data <- merge(data_save, po, by.x="result", by.y="chemical", all.x=TRUE)
```