615 Assignment Strawberry 3

Yiming Chen

2024-10-21

Preparing data for analysis —— Strawberries

read and explore the data

```
library(knitr)
library(kableExtra)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter()     masks stats::filter()
## x dplyr::group_rows() masks kableExtra::group_rows()
## x dplyr::lag()        masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(dplyr)
library(readr)
library(tidyr)
library(stringr)
library(ggplot2)
```

Read in the dataset and take a first look.

```
strawberry <- read_csv("strawberries25_v3.csv", col_names = TRUE)
```

```
## Rows: 12669 Columns: 21
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (15): Program, Period, Geo Level, State, State ANSI, Ag District, County...
## dbl  (2): Year, Ag District Code
## lgl  (4): Week Ending, Zip Code, Region, Watershed
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(strawberry)
```

```
## # A tibble: 6 x 21
##   Program  Year Period `Week Ending` `Geo Level` State   `State ANSI`
```

```
##   <chr>   <dbl> <chr>  <lgl>         <chr>       <chr>   <chr>
## 1 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 2 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 3 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 4 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 5 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 6 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## # i 14 more variables: `Ag District` <chr>, `Ag District Code` <dbl>,
## #   County <chr>, `County ANSI` <chr>, `Zip Code` <lgl>, Region <lgl>,
## #   watershed_code <chr>, Watershed <lgl>, Commodity <chr>, `Data Item` <chr>,
## #   Domain <chr>, `Domain Category` <chr>, Value <chr>, `CV (%)` <chr>
```

Remove the (D) term in Value and CV% columns

```
strawberry <- strawberry %>%
  mutate(
    Value = ifelse(Value == "(D)", NA, Value),
    `CV (%)` = ifelse(`CV (%)` == "(D)", NA, `CV (%)`)
  )
```

Do data cleaning for the Domain column, rearrange the info in this column into three columns: chemical category, name and number

```
strawberry <- strawberry %>%
  mutate(Category = case_when(
    Domain == "Total" ~ NA_character_,
    str_detect(Domain, "CHEMICAL") ~ str_trim(str_remove(Domain, "CHEMICAL, ")),
    TRUE ~ Domain
  ))
unique(strawberry$Category)
```

```
## [1] "TOTAL"          "AREA GROWN"     "ORGANIC STATUS" "FUNGICIDE"
## [5] "INSECTICIDE"    "OTHER"          "HERBICIDE"      "FERTILIZER"
```

Clean and transform the strawberry dataset by creating new columns based on specific conditions and regex extraction from the Domain Category field while handling special cases, such as missing values or unspecified categories.

```
strawberry <- strawberry %>%
  mutate(
    Name = case_when(
      Category == "TOTAL" ~ NA_character_,
      str_detect(`Domain Category`, fixed(Category)) & str_detect(`Domain Category`, "\\(.*=.*\\)") ~
        str_extract(`Domain Category`, "(?<=\\().*?(?=\\s?=)"),
      str_detect(`Domain Category`, fixed(Category)) & str_detect(`Domain Category`, "\\(.*\\)") ~
        str_extract(`Domain Category`, "(?<=\\().*?(?=\\))"),
      TRUE ~ NA_character_
    ),
    Number = case_when(
      Category == "TOTAL" ~ NA_real_,
      str_detect(`Domain Category`, fixed(Category)) & str_detect(`Domain Category`, "\\(.*=.*\\)") ~
        as.numeric(str_extract(`Domain Category`, "(?<=\\=\\s?).*?(?=\\))")),
      str_detect(`Domain Category`, fixed(Category)) & str_detect(`Domain Category`, "\\(.*\\)") ~
        NA_real_,
      TRUE ~ NA_real_
    )
  )
```

```
strawberry <- strawberry %>%
  mutate(Category = case_when(
    `Domain Category` == "NOT SPECIFIED" ~ NA_character_,
    TRUE ~ Category
  ))
```

data cleaning for AREA GROWN, the numerical intervals of the planted area are reintegrated inside the new columns, respectively, with the column names of Min and Max

```
strawberry <- strawberry %>%
  mutate(
    Min = case_when(
      str_detect(Name, "100 OR MORE ACRES") ~ 100,
      str_detect(Name, "TO") ~ as.numeric(str_extract(Name, "^[0-9.]+")),
      TRUE ~ NA_real_
    ),
    Max = case_when(
      str_detect(Name, "100 OR MORE ACRES") ~ "MORE",
      str_detect(Name, "TO") ~ str_extract(Name, "(?<=TO )^[0-9.]+"),
      TRUE ~ NA_character_
    )
)
```

Create a new column 'Unit' by extracting the substring after 'MEASURED'. Create a new column 'Type' by extracting either 'BEARING' or 'ORGANIC'. Create a new column 'Operation' by extracting the remaining parts of the string, Removing the 'MEASURED' part, the Unit and the Type, keeping the rest. Create a new column 'Operation' by extracting the remaining parts of the string, Removing the 'MEASURED', 'BEARING', 'ORGANIC', and 'STRAWBERRIES' parts.

```
strawberry <- strawberry %>%
  mutate(Unit = str_extract(strawberry$`Data Item`, "(?<=MEASURED ).*"))

strawberry <- strawberry %>%
  mutate(Type = str_extract(strawberry$`Data Item`, "BEARING|ORGANIC"))

strawberry <- strawberry %>%
  mutate(Operation = str_replace_all(strawberry$`Data Item`, "MEASURED.*|BEARING|ORGANIC", "") %>%
           str_trim())

strawberry <- strawberry %>%
  mutate(Operation = str_replace_all(strawberry$`Data Item`, "MEASURED.*|BEARING|ORGANIC|STRAWBERRIES(,
           str_replace_all("[-,]", "") %>%
           str_trim())
```

Export the cleaned dataset as a CSV file.

```
write.csv(strawberry, "cleaned_strawberries.csv", row.names = FALSE)
```

EDA

Check data types

```
str(strawberry)
```

```
## tibble [12,669 x 29] (S3: tbl_df/tbl/data.frame)
##  $ Program         : chr [1:12669] "CENSUS" "CENSUS" "CENSUS" "CENSUS" ...
```

```
##  $ Year            : num [1:12669] 2022 2022 2022 2022 2022 ...
##  $ Period          : chr [1:12669] "YEAR" "YEAR" "YEAR" "YEAR" ...
##  $ Week Ending     : logi [1:12669] NA NA NA NA NA NA ...
##  $ Geo Level       : chr [1:12669] "COUNTY" "COUNTY" "COUNTY" "COUNTY" ...
##  $ State           : chr [1:12669] "ALABAMA" "ALABAMA" "ALABAMA" "ALABAMA" ...
##  $ State ANSI      : chr [1:12669] "01" "01" "01" "01" ...
##  $ Ag District     : chr [1:12669] "BLACK BELT" "BLACK BELT" "BLACK BELT" "BLACK BELT" ...
##  $ Ag District Code: num [1:12669] 40 40 40 40 40 40 40 40 40 40 ...
##  $ County          : chr [1:12669] "BULLOCK" "BULLOCK" "BULLOCK" "BULLOCK" ...
##  $ County ANSI     : chr [1:12669] "011" "011" "011" "011" ...
##  $ Zip Code        : logi [1:12669] NA NA NA NA NA NA ...
##  $ Region          : logi [1:12669] NA NA NA NA NA NA ...
##  $ watershed_code  : chr [1:12669] "00000000" "00000000" "00000000" "00000000" ...
##  $ Watershed       : logi [1:12669] NA NA NA NA NA NA ...
##  $ Commodity       : chr [1:12669] "STRAWBERRIES" "STRAWBERRIES" "STRAWBERRIES" "STRAWBERRIES" ...
##  $ Data Item       : chr [1:12669] "STRAWBERRIES - ACRES BEARING" "STRAWBERRIES - ACRES GROWN" "STRA
##  $ Domain          : chr [1:12669] "TOTAL" "TOTAL" "TOTAL" "TOTAL" ...
##  $ Domain Category : chr [1:12669] "NOT SPECIFIED" "NOT SPECIFIED" "NOT SPECIFIED" "NOT SPECIFIED" .
##  $ Value           : chr [1:12669] NA "3" NA "1" ...
##  $ CV (%)          : chr [1:12669] NA "15.7" NA "(L)" ...
##  $ Category        : chr [1:12669] NA NA NA NA ...
##  $ Name            : chr [1:12669] NA NA NA NA ...
##  $ Number          : num [1:12669] NA NA NA NA NA NA NA NA NA NA ...
##  $ Min             : num [1:12669] NA NA NA NA NA NA NA NA NA NA ...
##  $ Max             : chr [1:12669] NA NA NA NA ...
##  $ Unit            : chr [1:12669] NA NA NA NA ...
##  $ Type            : chr [1:12669] "BEARING" NA "BEARING" "BEARING" ...
##  $ Operation       : chr [1:12669] "ACRES" "ACRES GROWN" "ACRES NON" "OPERATIONS WITH AREA" ...
```

Convert 'Value' to numeric, removing non-numeric characters. Convert 'CV (%)' to numeric, removing non-numeric characters (including %, parentheses). Check if conversion was successful. Check for any NAs introduced after conversion. Summary statistics for 'Value' and 'CV (%)'. Check for missing values in 'Value' and 'CV (%)'. Histogram for 'CV (%)'.

```r
strawberry$Value <- as.numeric(gsub("[^0-9.]", "", strawberry$Value))

strawberry$`CV (%)` <- as.numeric(gsub("[^0-9.]", "", strawberry$`CV (%)`))

str(strawberry$Value)
```

```
##  num [1:12669] NA 3 NA 1 6 5 NA NA 2 2 ...
```

```r
str(strawberry$`CV (%)`)
```

```
##  num [1:12669] NA 15.7 NA NA 52.7 47.6 NA NA 55.7 52.7 ...
```

```r
sum(is.na(strawberry$Value))
```

```
## [1] 4744
```

```r
sum(is.na(strawberry$`CV (%)`))
```

```
## [1] 7934
```

```r
summary(strawberry$Value)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
## 0.000e+00 2.000e+00 4.000e+00 1.123e+07 2.100e+01 3.584e+09      4744
```

```r
summary(strawberry$`CV (%)`)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.60   29.50   41.60   43.43   56.10   99.90    7934
```

```r
sum(is.na(strawberry$Value))
```

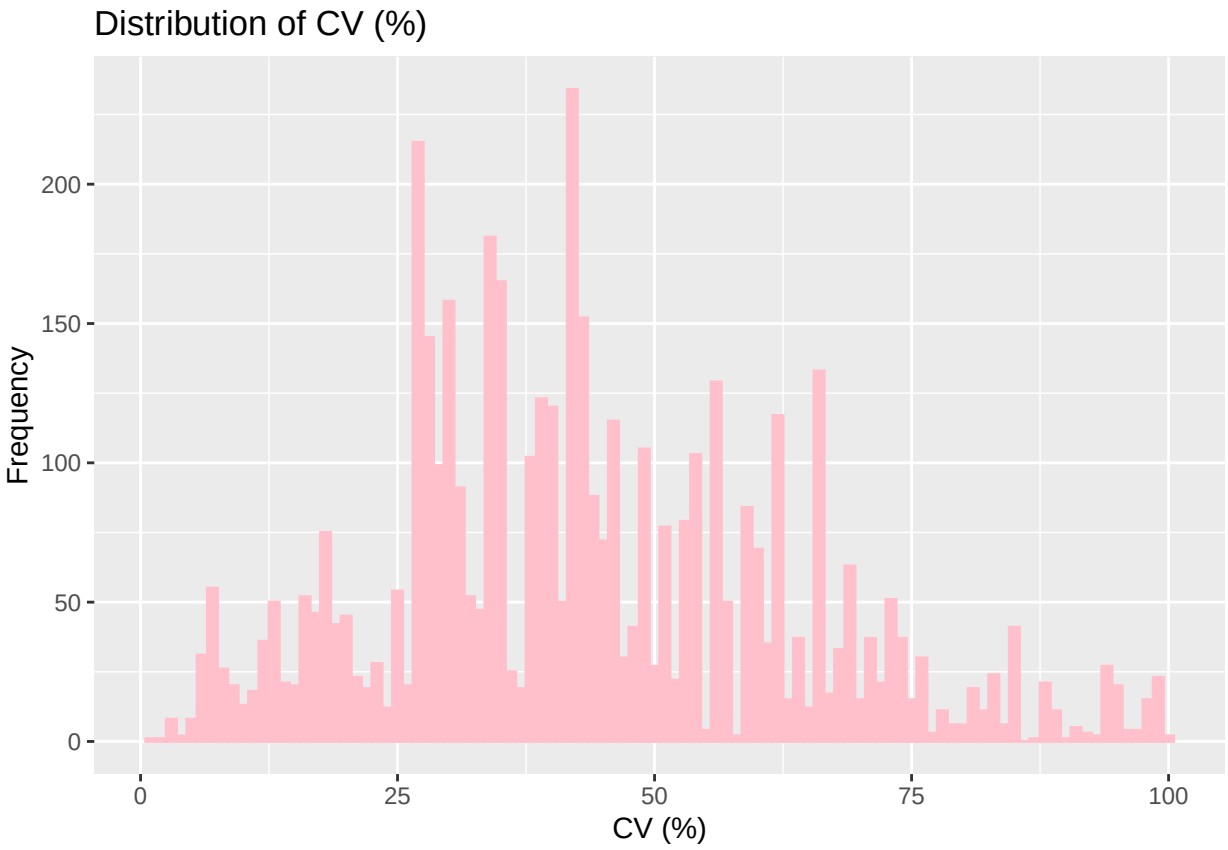
```
## [1] 4744
```

```r
sum(is.na(strawberry$`CV (%)`))
```

```
## [1] 7934
```

```r
ggplot(strawberry, aes(x = `CV (%)`)) +
  geom_histogram(binwidth = 1, col = "pink", fill = "pink") +
  labs(title = "Distribution of CV (%)", x = "CV (%)", y = "Frequency")
```

```
## Warning: Removed 7934 rows containing non-finite outside the scale range
## (`stat_bin()`).
```

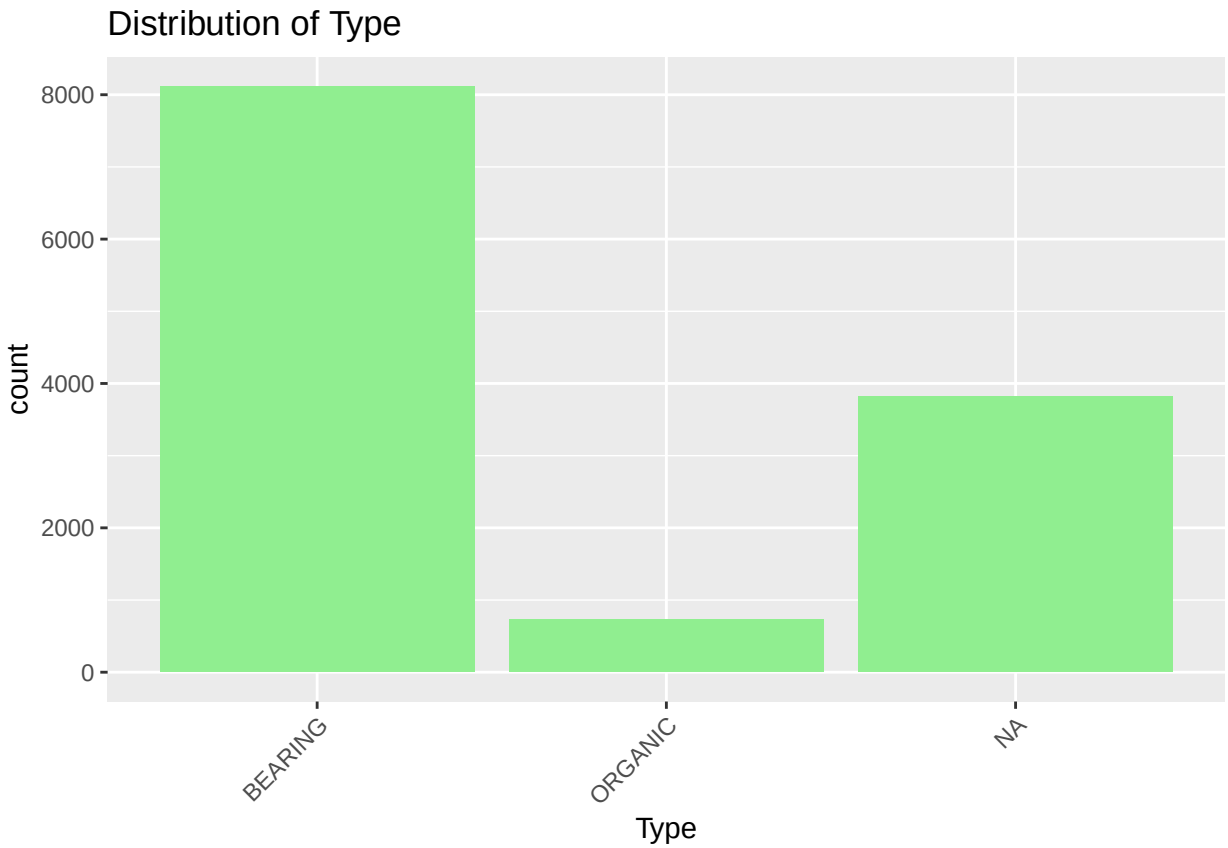


The Value column shows a strong right skew with most data concentrated at lower values and only a few larger ones. The CV (%) column displays a more spread distribution. The frequent occurrence of CV values between 20% and 30% may indicate that this range represents the typical variation in the dataset. However, the existence of high CV values suggests that certain categories or items show much higher variability.

Bar plot for 'Type' column

```r
ggplot(strawberry, aes(x=Type)) +
  geom_bar(fill="lightgreen") +
  theme(axis.text.x = element_text(angle=45, hjust=1)) +
```

```
  labs(title="Distribution of Type")
```



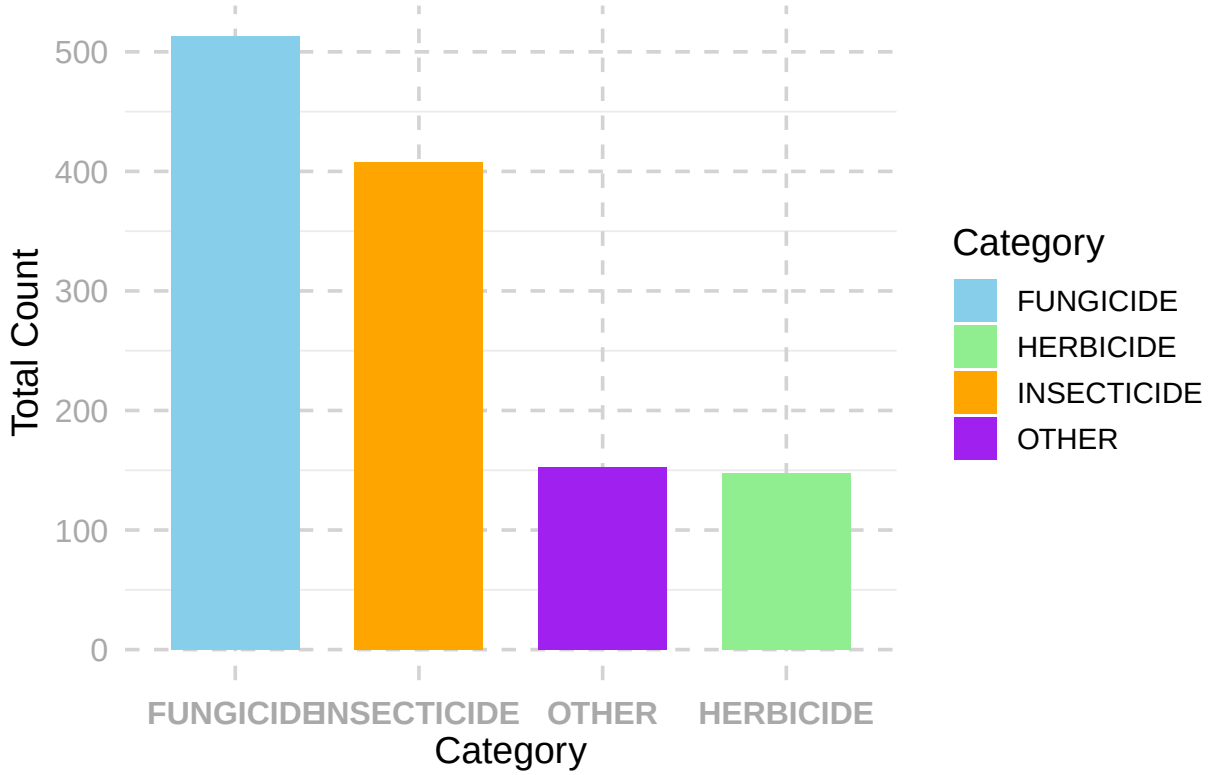
The BEARING type is the most common category in the Type column, while ORGANIC data points are minimal. The significant proportion of NA values suggests that a substantial amount of Type information is missing, which could have implications for further analyses or interpretations related to strawberry types.

```
library(tidyverse)
library(ggplot2)
strawberry = read.csv("cleaned_strawberries.csv")
view(strawberry)
```

Count the total number of occurrences of chemicals in each category. Create a bar chart for the total counts of each category.

```
filtered_data <- strawberry %>%
  filter(State == "FLORIDA" &
         Category %in% c("FUNGICIDE", "OTHER", "HERBICIDE", "INSECTICIDE"))

category_total_counts <- filtered_data %>%
  group_by(Category) %>%
  summarise(Total_Count = n()) %>%
  arrange(desc(Total_Count))

ggplot(category_total_counts, aes(x = reorder(Category, -Total_Count), y = Total_Count, fill = Category)
  geom_bar(stat = "identity", width = 0.7) +
  scale_fill_manual(values = c("FUNGICIDE" = "skyblue", "HERBICIDE" = "lightgreen",
                               "INSECTICIDE" = "orange", "OTHER" = "purple")) +  # Custom colors for ea
  theme_minimal(base_size = 14) +
```

```
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5, vjust = 0.5, face = "bold"),  # Style x-axis
        axis.text = element_text(size = 12, color = "darkgray"),  # Adjust font size and color
        plot.title = element_text(hjust = 0.5, face = "bold", color = "darkblue"),  # Center and style
        panel.grid.major = element_line(color = "lightgray", linetype = "dashed")) +  # Dashed grid lin
  labs(title = "Total Count of Chemicals by Category in Florida",
       x = "Category", y = "Total Count")
```

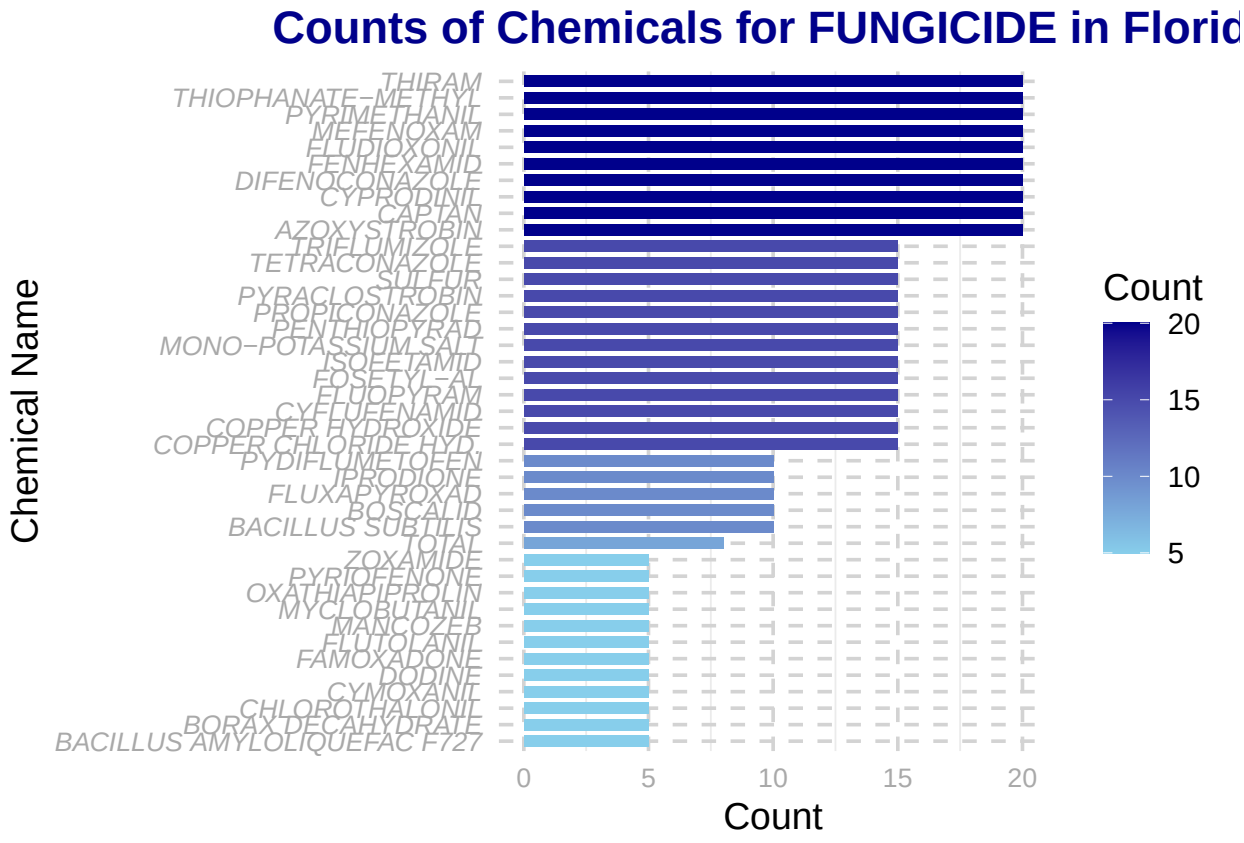


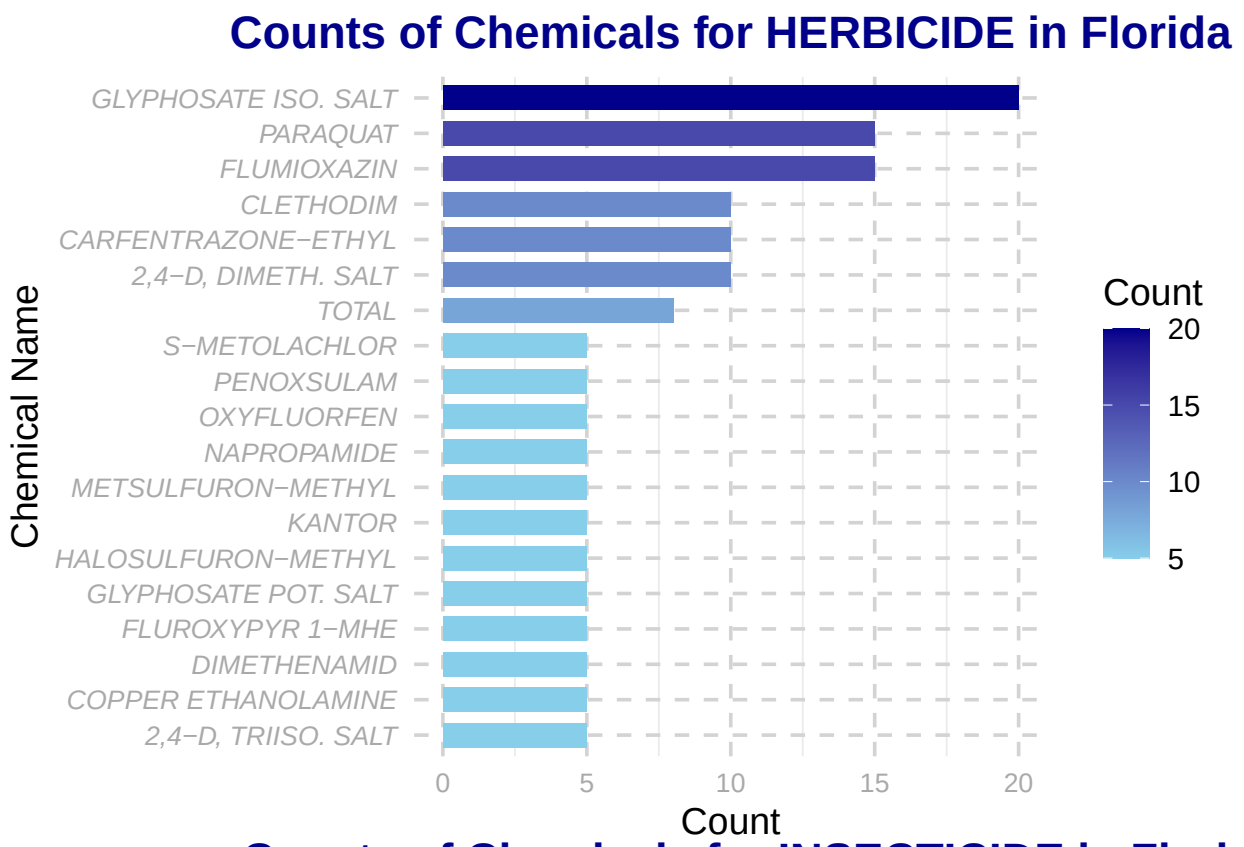
This chart effectively highlights the distribution of chemical categories in Florida, making it easy to compare the total counts visually.The bar chart displays the counts for each category in Florida, showing "FUNGICIDE" and "INSECTICIDE" as the most frequently occurring categories, with "OTHER" and "HERBICIDE" following.

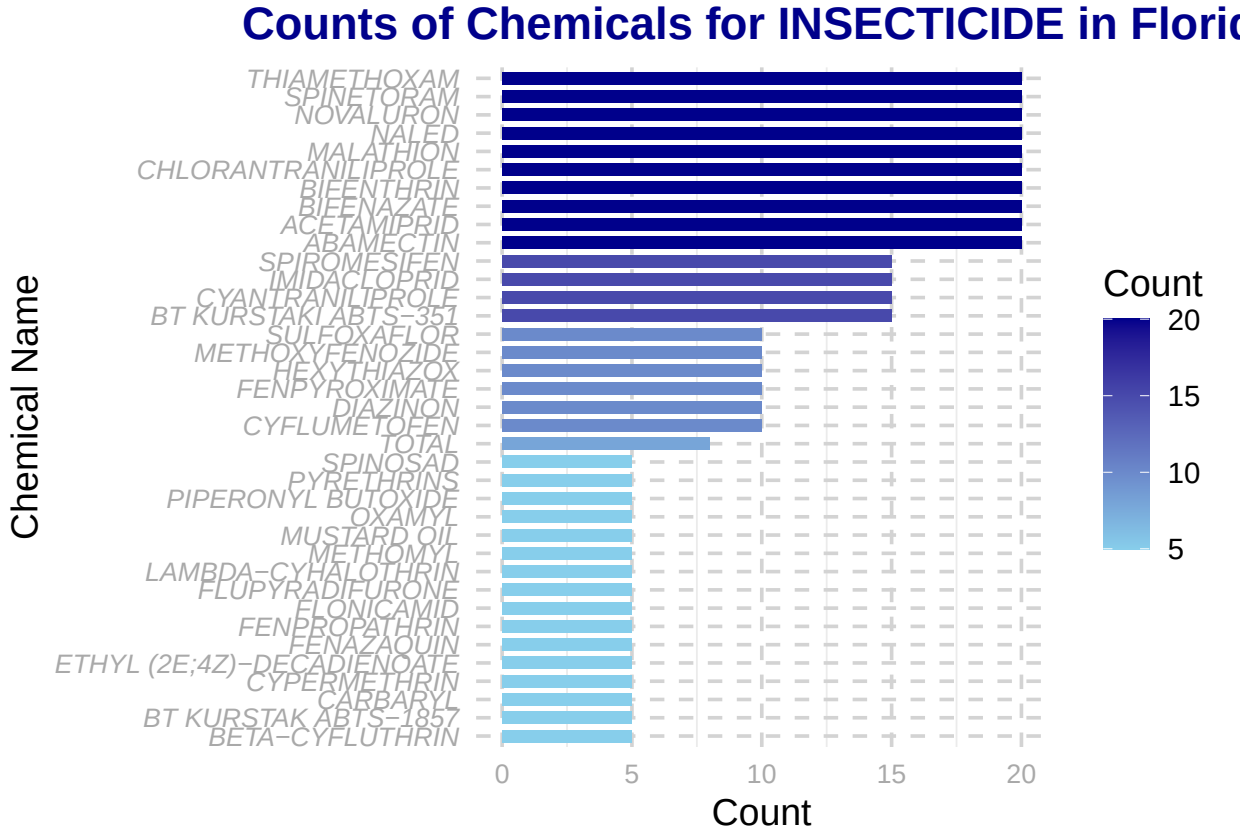
Filter data to include only the categories FUNGICIDE, OTHER, HERBICIDE, INSECTICIDE, State = New York, and Program = SURVEY. Count the number of occurrences of each chemical name within each category. Create a function to plot bar chart for each category. Check if there's data to plot. Generate and print plots for each category. Print only plots that were successfully created.

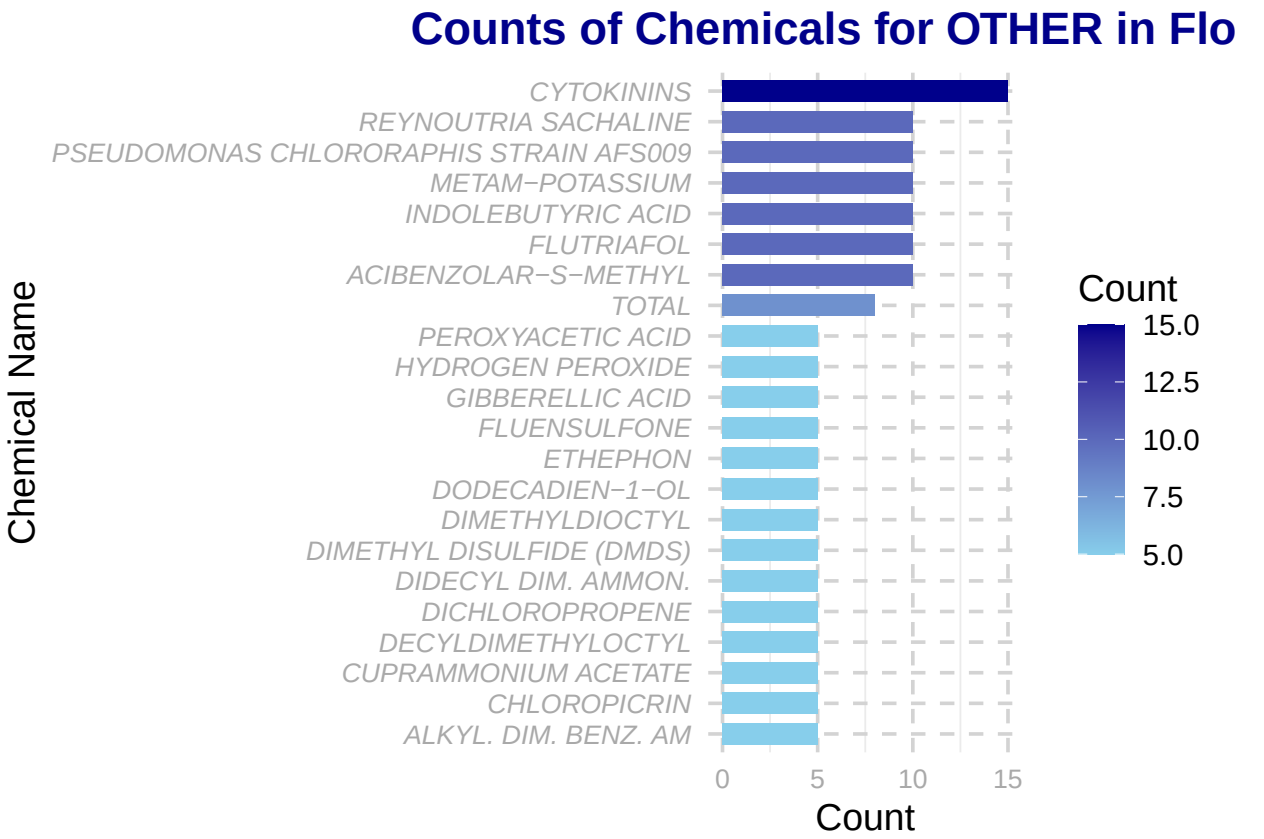
```
filtered_data <- strawberry %>%
  filter(State == "FLORIDA" & Program == "SURVEY" &
         Category %in% c("FUNGICIDE", "OTHER", "HERBICIDE", "INSECTICIDE"))

category_chemical_counts <- filtered_data %>%
  group_by(Category, Name) %>%
  summarise(Count = n()) %>%
  arrange(Category, desc(Count))

plot_category <- function(category_name) {
  subset_data <- category_chemical_counts %>%
```

```
    filter(Category == category_name)

  if(nrow(subset_data) == 0) {
    message(paste("No data available for category:", category_name))
    return(NULL)
  }

  ggplot(subset_data, aes(y = reorder(Name, Count), x = Count, fill = Count)) +
    geom_bar(stat = "identity", width = 0.7) +
    scale_fill_gradient(low = "skyblue", high = "darkblue") +  # Add a gradient fill
    theme_minimal(base_size = 14) +
    theme(axis.text.y = element_text(angle = 0, hjust = 1, vjust = 0.5, face = "italic"),  # Style y-ax
          axis.text = element_text(size = 10, color = "darkgray"),  # Adjust font size and color
          plot.title = element_text(hjust = 0.5, face = "bold", color = "darkblue"),  # Center and styl
          panel.grid.major = element_line(color = "lightgray", linetype = "dashed")) +  # Dashed grid l
    labs(title = paste("Counts of Chemicals for", category_name, "in Florida"),
         y = "Chemical Name", x = "Count")
}

categories <- c("FUNGICIDE", "HERBICIDE", "INSECTICIDE", "OTHER")
plots <- lapply(categories, plot_category)

for (plot in plots) {
  if (!is.null(plot)) {
    print(plot)
  }
}
```



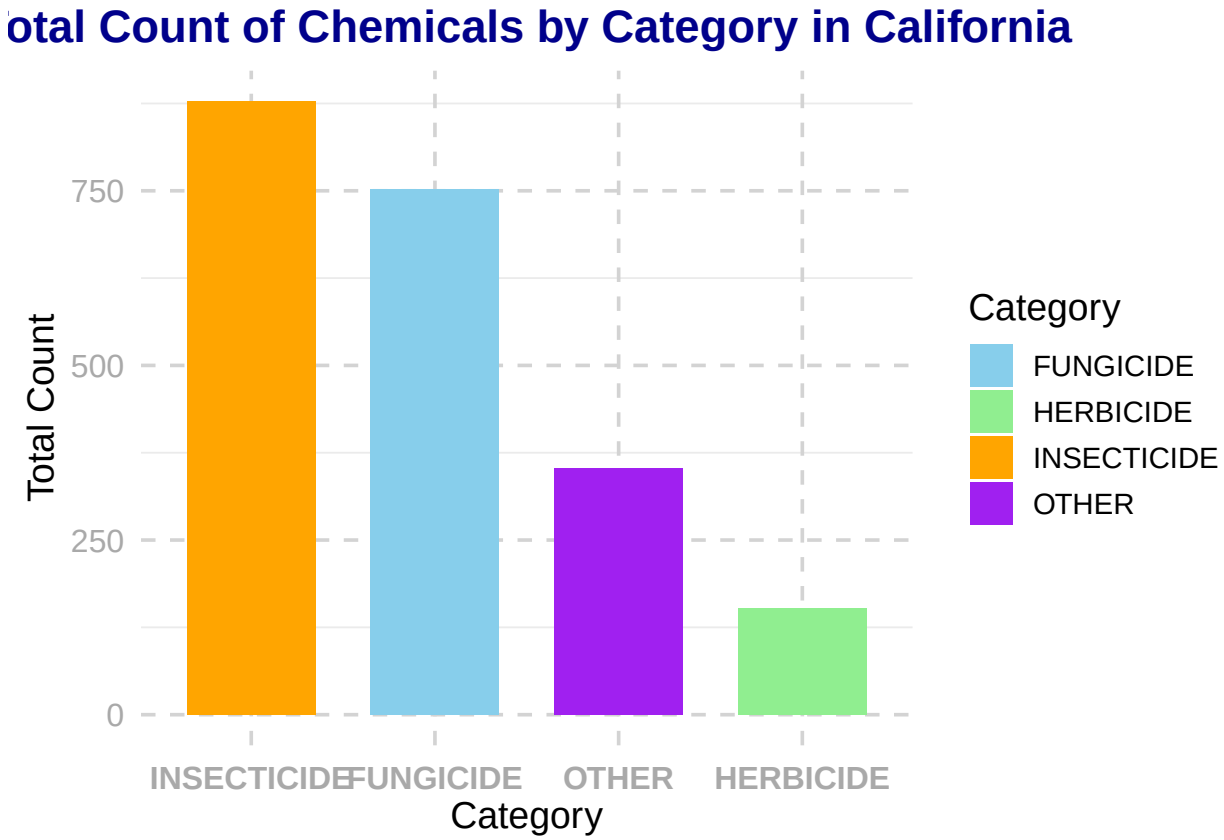

Counts of Chemicals for HERBICIDE in Florida
GLYPHOSATE ISO. SALT
PARAQUAT
FLUMIOXAZIN
CLETHODIM
CARFENTRAZONE−ETHYL
2,4−D, DIMETH. SALT
TOTAL
S−METOLACHLOR
PENOXSULAM
OXYFLUORFEN
NAPROPAMIDE
METSULFURON−METHYL
KANTOR
HALOSULFURON−METHYL
GLYPHOSATE POT. SALT
FLUROXYPYR 1−MHE
DIMETHENAMID
COPPER ETHANOLAMINE
2,4−D, TRIISO. SALT
Chemical Name
0
5
10
15
20
Count
Count
20
15
10
5


Counts of Chemicals for INSECTICIDE in Floric
THIAMETHOXAM
SPINETORAM
NOVALURON
NALED
MALATHION
CHLORANTRANILIPROLE
BIFENTHRIN
BIFENAZATE
ACETAMIPRID
ABAMECTIN
SPIROMESIFEN
IMIDACLOPRID
CYANTRANILIPROLE
BT KURSTAKI ABTS−351
SULFOXAFLOR
METHOXYFENOZIDE
HEXYTHIAZOX
FENPYROXIMATE
DIAZINON
CYFLUMETOFEN
TOTAL
SPINOSAD
PYRETHRINS
PIPERONYL BUTOXIDE
OXAMYL
MUSTARD OIL
METHOMYL
LAMBDA−CYHALOTHRIN
FLUPYRADIFURONE
ELONICAMID
FENPROPATHRIN
FENAZAQUIN
ETHYL (2E;4Z)−DECADIENOATE
CYPERMETHRIN
CARBARYL
BT KURSTAK ABTS−1857
BETA−CYFLUTHRIN
Chemical Name
0
5
10
15
20
Count
Count
20
15
10
5



The resulting plots display each category's chemical counts in Florida with a color gradient indicating the count levels, making it easy to compare the usage frequency of each chemical within each category. These visualizations are useful for assessing which chemicals are most common within each category in Florida's survey data, especially with the gradient highlighting relative frequencies.

```
filtered_data <- strawberry %>%
  filter(State == "CALIFORNIA" &
         Category %in% c("FUNGICIDE", "OTHER", "HERBICIDE", "INSECTICIDE"))

# Count the total number of occurrences of chemicals in each category
category_total_counts <- filtered_data %>%
  group_by(Category) %>%
  summarise(Total_Count = n()) %>%
  arrange(desc(Total_Count))

# Create a bar chart for the total counts of each category
ggplot(category_total_counts, aes(x = reorder(Category, -Total_Count), y = Total_Count, fill = Category
  geom_bar(stat = "identity", width = 0.7) +
  scale_fill_manual(values = c("FUNGICIDE" = "skyblue", "HERBICIDE" = "lightgreen",
                               "INSECTICIDE" = "orange", "OTHER" = "purple")) +  # Custom colors for ea
  theme_minimal(base_size = 14) +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5, vjust = 0.5, face = "bold"),  # Style x-axis
        axis.text = element_text(size = 12, color = "darkgray"),  # Adjust font size and color
        plot.title = element_text(hjust = 0.5, face = "bold", color = "darkblue"),  # Center and style
        panel.grid.major = element_line(color = "lightgray", linetype = "dashed")) +  # Dashed grid lin
  labs(title = "Total Count of Chemicals by Category in California",
       x = "Category", y = "Total Count")
```



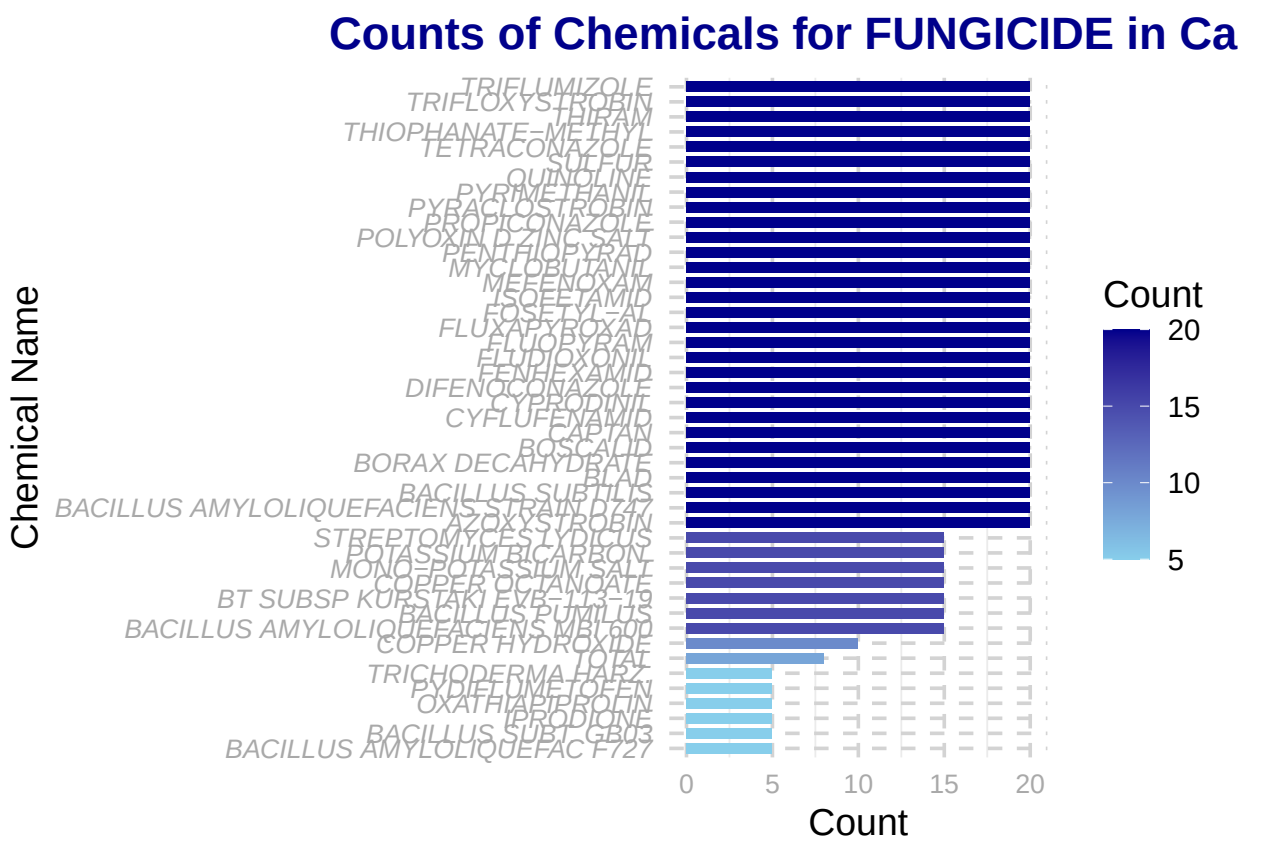
This chart effectively highlights the distribution of chemical categories in California, making it easy to compare the total counts visually.The bar chart displays the counts for each category in California, showing "INSECTICIDE" and "FUNGICIDE" as the most frequently occurring categories, with "OTHER" and "HERBICIDE" following.

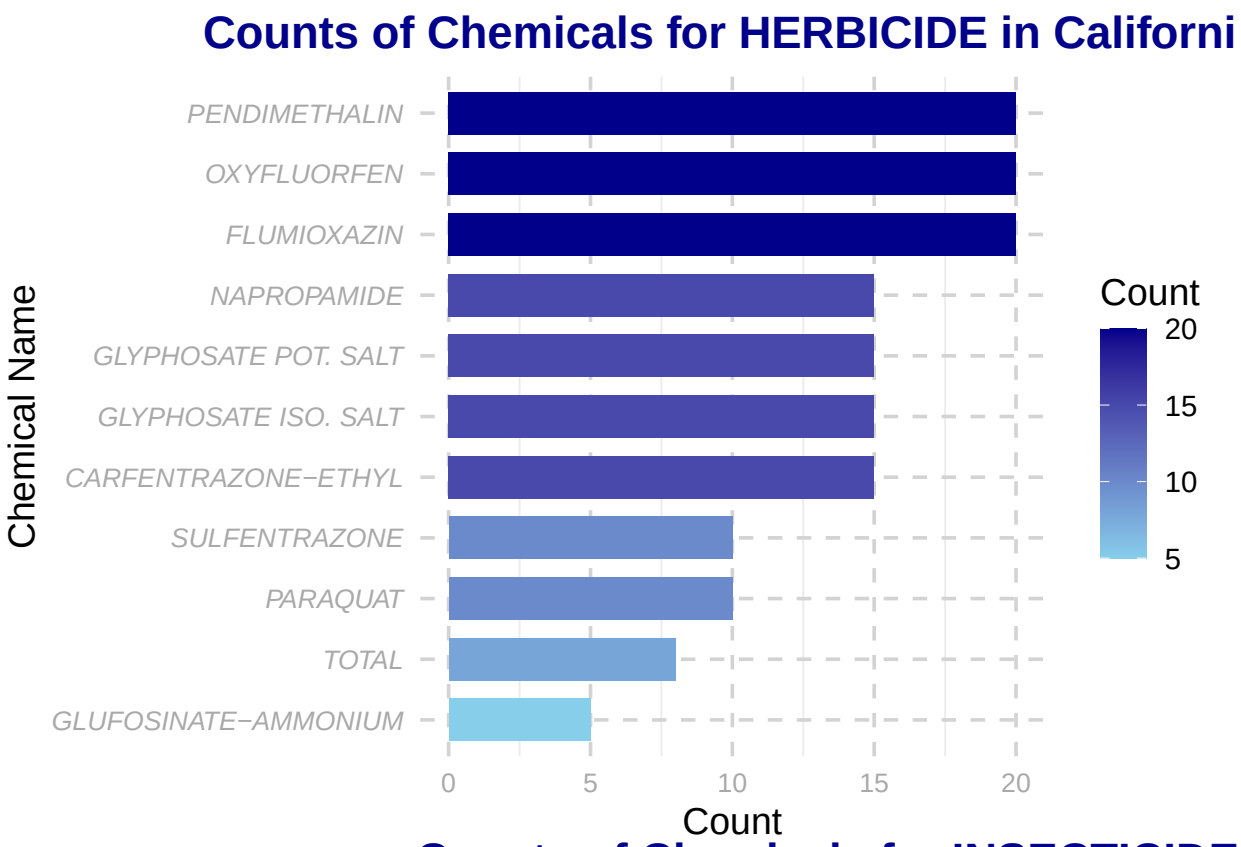
```
# Filter data to include only the categories FUNGICIDE, OTHER, HERBICIDE, INSECTICIDE,
# State = New York, and Program = SURVEY
filtered_data <- strawberry %>%
  filter(State == "CALIFORNIA" & Program == "SURVEY" &
         Category %in% c("FUNGICIDE", "OTHER", "HERBICIDE", "INSECTICIDE"))

# Count the number of occurrences of each chemical name within each category
category_chemical_counts <- filtered_data %>%
  group_by(Category, Name) %>%
  summarise(Count = n()) %>%
  arrange(Category, desc(Count))
```

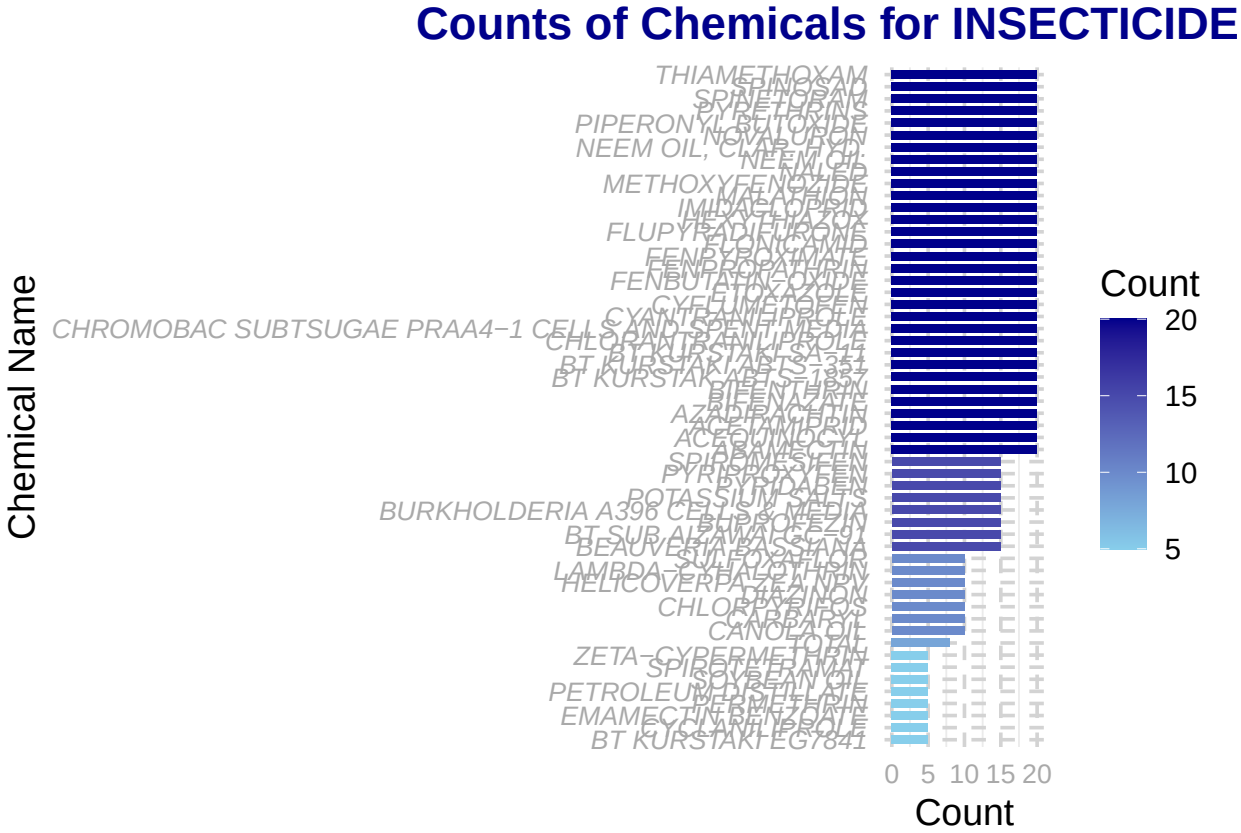
```
## `summarise()` has grouped output by 'Category'. You can override using the
## `.groups` argument.
```

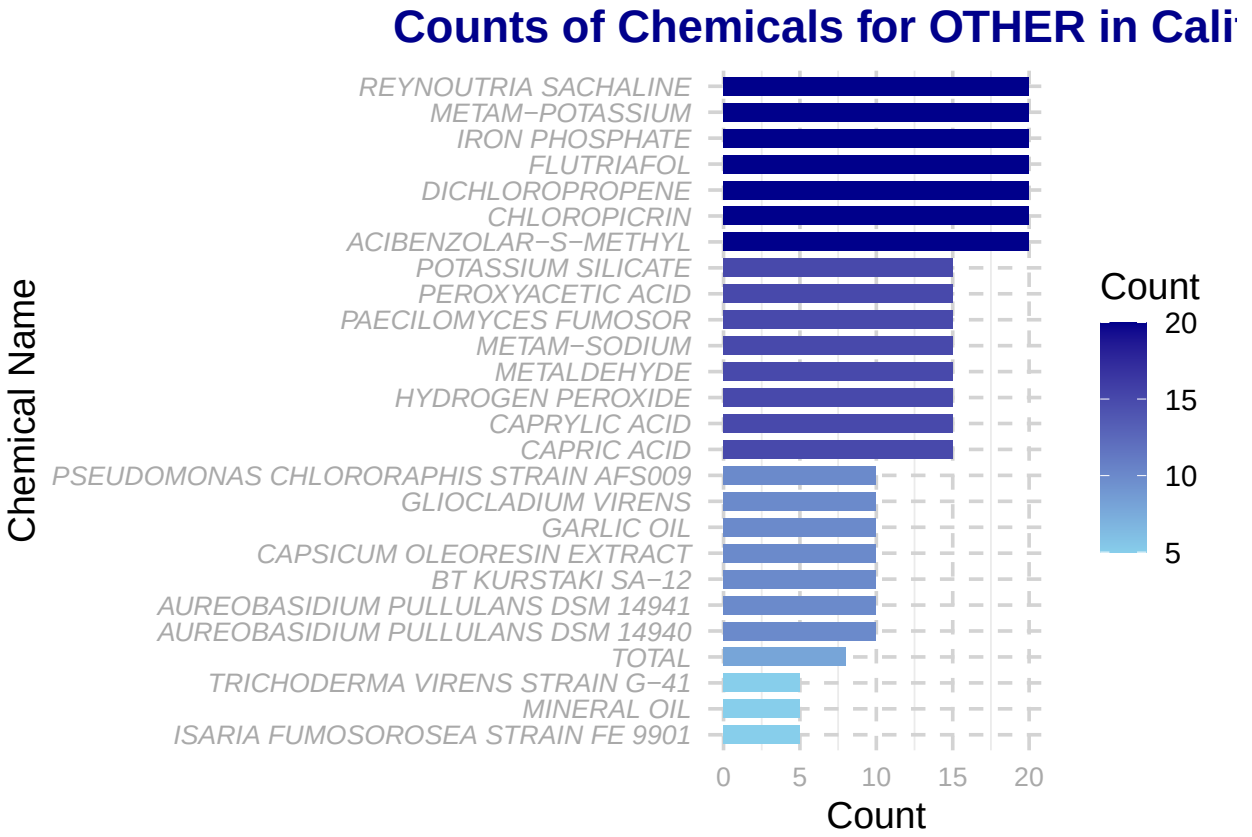
```
# Create a function to plot bar chart for each category
plot_category <- function(category_name) {
  subset_data <- category_chemical_counts %>%
    filter(Category == category_name)

  # Check if there's data to plot
  if(nrow(subset_data) == 0) {
    message(paste("No data available for category:", category_name))
    return(NULL)
```

```
  }

  ggplot(subset_data, aes(y = reorder(Name, Count), x = Count, fill = Count)) +
    geom_bar(stat = "identity", width = 0.7) +
    scale_fill_gradient(low = "skyblue", high = "darkblue") +  # Add a gradient fill
    theme_minimal(base_size = 14) +
    theme(axis.text.y = element_text(angle = 0, hjust = 1, vjust = 0.5, face = "italic"),  # Style y-ax
          axis.text = element_text(size = 10, color = "darkgray"),  # Adjust font size and color
          plot.title = element_text(hjust = 0.5, face = "bold", color = "darkblue"),  # Center and styl
          panel.grid.major = element_line(color = "lightgray", linetype = "dashed")) +  # Dashed grid l
    labs(title = paste("Counts of Chemicals for", category_name, "in California"),
         y = "Chemical Name", x = "Count")
}

# Generate and print plots for each category
categories <- c("FUNGICIDE", "HERBICIDE", "INSECTICIDE", "OTHER")
plots <- lapply(categories, plot_category)

# Print only plots that were successfully created
for (plot in plots) {
  if (!is.null(plot)) {
    print(plot)
  }
}
```




Counts of Chemicals for HERBICIDE in Californi
PENDIMETHALIN
OXYFLUORFEN
FLUMIOXAZIN
NAPROPAMIDE
GLYPHOSATE POT. SALT
GLYPHOSATE ISO. SALT
CARFENTRAZONE−ETHYL
SULFENTRAZONE
PARAQUAT
TOTAL
GLUFOSINATE−AMMONIUM
Chemical Name
0
5
10
15
20
Count
Count
20
15
10
5


Counts of Chemicals for INSECTICIDE
THIAMETHOXAM
SPINOSAD
SPIROTETRAMAT
PYRETHRINS
PIPERONYL BUTOXIDE
NOVALURON
NEEM OIL, CLARIFIED
NEEM OIL
METHOXYFENOZIDE
MALATHION
IMIDACLOPRID
FLUPYRADIFURONE
FLONICAMID
FENPYROXIMATE
FENPROPATHRIN
FENBUTATIN−OXIDE
ETOXAZOLE
CYFLUMETOFEN
CYANTRANILIPROLE
CHROMOBAC SUBTSUGAE PRAA4−1 CELLS AND SPENT MEDIA
CHLORANTRANILIPROLE
BT KURSTAKI ABTS−351
BT KURSTAKI ABTS−1857
BIFENTHRIN
AZADIRACHTIN
ACEQUINOCYL
ABAMECTIN
SPIROMESIFEN
PYRIPROXYFEN
POTASSIUM OLEATE
BURKHOLDERIA A396 CELLS AND SPENT MEDIA
BACILLUS FIRMUS I−1582
BT SUB AIZAWAI GC−91
BEAUVERIA BASSIANA
SULFOXAFLOR
LAMBDA−CYHALOTHRIN
HELICOVERPA ARMIGERA NPV
AZINPHOS−METHYL
CHLORPYRIFOS
CARBARYL
CANOLA OIL
TOTAL
ZETA−CYPERMETHRIN
SPIROMESIFEN
SOYBEAN OIL
PETROLEUM OIL
BIFENTHRIN
EMAMECTIN BENZOATE
CYCLANILIPROLE
BT KURSTAKI EG7841
Chemical Name
0
5
10
15
20
Count
Count
20
15
10
5



The resulting plots display each category's chemical counts in California with a color gradient indicating the count levels, making it easy to compare the usage frequency of each chemical within each category. These visualizations are useful for assessing which chemicals are most common within each category in California's survey data, especially with the gradient highlighting relative frequencies.

A process to retrieve and display Global Harmonized System (GHS) hazard information for various chemicals. GHS Search and Hazard Retrieval: The code uses functions GHS_searcher and hazards_retriever to look up GHS hazard information based on a chemical identifier (result). This hazard information is saved in hazards. Storing Results in a List:

The retrieved hazard data (hazards) for each chemical is stored in a list named results_list, with each chemical's name as the list key. Converting to a Data Frame:

The results_list is converted to a data frame (results_df) using enframe, with the column names set to "Chemical" and "Hazard_Statements." unnest is used to expand Hazard_Statements (which may contain multiple statements for each chemical) into separate rows. Displaying the Data Frame:

The results_df is displayed, showing each chemical and its associated GHS hazard statements. Each hazard code and statement (e.g., "H302: Harmful if swallowed") is presented per row for easy reference.

```
library(tidyverse)
library(PubChemR)

GHS_searcher <- function(result_json_object) {
  hierarchies <- result_json_object[["result"]][["Hierarchies"]][["Hierarchy"]]

  for (i in seq_along(hierarchies)) {
    if (hierarchies[[i]][["SourceName"]] == "GHS Classification (UNECE)") {
      return(i)
    }
```

```
  }
  # Return NULL if GHS Classification is not found
  return(NULL)
}

hazards_retriever <- function(index, result_json_object) {
  if (is.null(index)) {
    return(NA)  # Return NA if GHS data is not available
  }

  hierarchy <- result_json_object[["result"]][["Hierarchies"]][["Hierarchy"]][[index]]
  nodes <- hierarchy[["Node"]]
  hazard_statements <- c()
  i <- 1

  while (i <= length(nodes) && str_detect(nodes[[i]][["Information"]][["Name"]], "^H")) {
    hazard_statements <- c(hazard_statements, nodes[[i]][["Information"]][["Name"]])
    i <- i + 1
  }
  if (length(hazard_statements) == 0) {
    return(NA)
  }
  return(hazard_statements)
}

# List of chemicals to process
chemical_vec <- c("reynoutria sachaline", "flutriafol", "chloropicrin")

# Initialize an empty list to store results
results_list <- list()

for (chemical in chemical_vec) {
  result <- get_pug_rest(
    identifier = chemical,
    namespace = "name",
    domain = "compound",
    operation = "classification",
    output = "JSON"
  )

  ghs_index <- GHS_searcher(result)
  hazards <- hazards_retriever(ghs_index, result)

  # Store the results in a list
  results_list[[chemical]] <- hazards
}

# Convert the results list into a data frame
results_df <- results_list %>%
  enframe(name = "Chemical", value = "Hazard_Statements") %>%
  unnest(cols = c(Hazard_Statements))

# Display the data frame
```

```r
print(results_df)
```

```
## # A tibble: 25 x 2
##    Chemical             Hazard_Statements
##    <chr>                <chr>
##  1 reynoutria sachaline <NA>
##  2 flutriafol           H302: Harmful if swallowed [Warning Acute toxicity, ora~
##  3 flutriafol           H300: Health Hazards
##  4 flutriafol           Hazard Statement Codes
##  5 flutriafol           H312: Harmful in contact with skin [Warning Acute toxic~
##  6 flutriafol           H332: Harmful if inhaled [Warning Acute toxicity, inhal~
##  7 flutriafol           H411: Toxic to aquatic life with long lasting effects [~
##  8 flutriafol           H400: Environmental Hazards
##  9 flutriafol           H412: Harmful to aquatic life with long lasting effects~
## 10 chloropicrin         H301: Toxic if swallowed [Danger Acute toxicity, oral]
## # i 15 more rows
```

The output table lists chemicals and their hazard statements. For example: "flutriafol" has multiple hazards like "H302: Harmful if swallowed" and "H410: Toxic to aquatic life." "chloropicrin" includes hazards such as "H330: Fatal if inhaled" and "H410: Very toxic to aquatic life." This table provides a comprehensive overview of hazard classifications for each chemical, facilitating risk assessment and safety measures.