Joanne Chen

CMSC320 Spring 2019 Final Project

Honey Production in the USA from 1998-2012

Bees. Some of us may have had painful memories when it comes to bees. Apart from their stingers, bees are vital to Earth; for instance, plants need bees to pollinate. Pollination gives way to habitats for animals and food production for all.

However, climate change, harmful pesticides, loss of habitat, etc. have significantly contributed to the rapid decline of the honeybee population in the past decades. Subsequently, the honey production has been negatively affected.

Data Science has helped researchers, environmentalists, and others observe the bee population decline and bring more attention to the importance of taking actions towards protecting the bees.

Here we explore a dataset regarding the honey production in the United States of America from 1998 to 2012, while also providing a walk through tutorial of the entire data science pipeline.

Sources: - https://www.earthday.org/campaigns/endangered-species/bees/ - http://sos-bees.org/ - https://www.planetbee.org/save-honeybees

Dataset URL: https://www.kaggle.com/jessicali9530/honey-production

Pre-Processing

First, we are calling numerous libraries that are necessary for our code because these libraries consist of many useful R functions that will be used in the code.

```r
# Calling Libraries
library(tidyverse)
library(rvest)
library(dplyr)
library(ggplot2)
library(broom)
library(magrittr)
library(tidyr)
library(stringr)
```

Data Loading and Parsing

After downloading a CSV file from Kaggle, use the read_csv() function with the file path as the parameter to read the data from a CSV file into a data frame in R. When loading a CSV file, you can parse it with column specifications using the cols() or cols_only() function. cols() includes all columns in the data, whereas cols_only() only include the columns specified.

For more information on the read_csv() function:
- https://readr.tidyverse.org/reference/read_delim.html for more information on the cols() and cols_only() function: - https://www.rdocumentation.org/packages/readr/versions/1.3.1/topics/cols Find datasets through links provided here: - http://www.hcbravo.org/IntroDataSci/resources/ - https://www.kaggle.com/datasets

```r
df <- read_csv('/Users/JoanneChen/Desktop/honeyproduction.csv')
```

```
## Parsed with column specification:
## cols(
##   state = col_character(),
##   numcol = col_double(),
##   yieldpercol = col_double(),
##   totalprod = col_double(),
##   stocks = col_double(),
##   priceperlb = col_double(),
##   prodvalue = col_double(),
##   year = col_double()
## )
```

```
# Parsed with column specification
cols(
  state = col_character(),
  numcol = col_double(),
  yieldpercol = col_double(),
  totalprod = col_double(),
  stocks = col_double(),
  priceperlb = col_double(),
  prodvalue = col_double(),
  year = col_integer()
)
```

```
## cols(
##   state = col_character(),
##   numcol = col_double(),
##   yieldpercol = col_double(),
##   totalprod = col_double(),
##   stocks = col_double(),
##   priceperlb = col_double(),
##   prodvalue = col_double(),
##   year = col_integer()
## )
```

```
df
```

```
## # A tibble: 626 x 8
##    state numcol yieldpercol totalprod   stocks priceperlb prodvalue  year
##    <chr>  <dbl>       <dbl>     <dbl>    <dbl>      <dbl>     <dbl> <dbl>
##  1 AL     16000          71   1136000   159000       0.72    818000  1998
##  2 AZ     55000          60   3300000  1485000       0.64   2112000  1998
##  3 AR     53000          65   3445000  1688000       0.59   2033000  1998
##  4 CA    450000          83  37350000 12326000       0.62  23157000  1998
##  5 CO     27000          72   1944000  1594000       0.7    1361000  1998
##  6 FL    230000          98  22540000  4508000       0.64  14426000  1998
##  7 GA     75000          56   4200000   307000       0.69   2898000  1998
##  8 HI      8000         118    944000    66000       0.77    727000  1998
##  9 ID    120000          50   6000000  2220000       0.65   3900000  1998
## 10 IL      9000          71    639000   204000       1.19    760000  1998
## # ... with 616 more rows
```

The set_colnames function allows for renaming column names.

```
# Change column names
df <- df %>%
  set_colnames(c("state", "num_colonies", "yield_per_colony_lb", "total_production", "stocks", "price_pe
```

```r
df
```

```
## # A tibble: 626 x 8
##    state num_colonies yield_per_colon~ total_production stocks price_per_lb
##    <chr>        <dbl>            <dbl>            <dbl>  <dbl>        <dbl>
##  1 AL           16000               71          1136000 1.59e5         0.72
##  2 AZ           55000               60          3300000 1.48e6         0.64
##  3 AR           53000               65          3445000 1.69e6         0.59
##  4 CA          450000               83         37350000 1.23e7         0.62
##  5 CO           27000               72          1944000 1.59e6         0.7
##  6 FL          230000               98         22540000 4.51e6         0.64
##  7 GA           75000               56          4200000 3.07e5         0.69
##  8 HI            8000              118           944000 6.60e4         0.77
##  9 ID          120000               50          6000000 2.22e6         0.65
## 10 IL            9000               71           639000 2.04e5         1.19
## # ... with 616 more rows, and 2 more variables: production_value <dbl>,
## #   year <dbl>
```

Exploratory Data Analysis

Operations, such as select, filter, slice, arrange, group_by, and summarise, are used to help perform almost any analysis on data frames.

Using the various operations, we can determine which state has had the largest and smallest total average honey production between 1998 and 2012, and on average, how many colonies it takes to create 1 pound of honey for each state.

Documentation of operations used: - group_by(): https://www.rdocumentation.org/packages/dplyr/versions/0.7.8/topics/group_by - summarise(): https://www.rdocumentation.org/packages/dplyr/versions/0.7.8/topics/summarise - filter(): https://www.rdocumentation.org/packages/dplyr/versions/0.7.8/topics/filter - mutate(): https://www.rdocumentation.org/packages/dplyr/versions/0.5.0/topics/mutate

```r
# North Dakota (ND) has the largest average honey production
df %>%
  group_by(state) %>%
  summarise(avg_production = mean(total_production)) %>%
  filter(avg_production == max(avg_production))
```

```
## # A tibble: 1 x 2
##   state avg_production
##   <chr>          <dbl>
## 1 ND         31672333.
```

```r
# Oklahoma (OK) has the smallest average honey production
df %>%
  group_by(state) %>%
  summarise(avg_production = mean(total_production)) %>%
  filter(avg_production == min(avg_production))
```

```
## # A tibble: 1 x 2
##   state avg_production
##   <chr>          <dbl>
## 1 OK           201167.
```

```r
# On average, the amount of colonies it takes to make 1 pound of honey for each state
df %>%
  group_by(state) %>%
  summarise(avg_production = mean(total_production),
            avg_yield = mean(yield_per_colony_lb)) %>%
  mutate(num_colonies_1lb = avg_production/avg_yield)
```

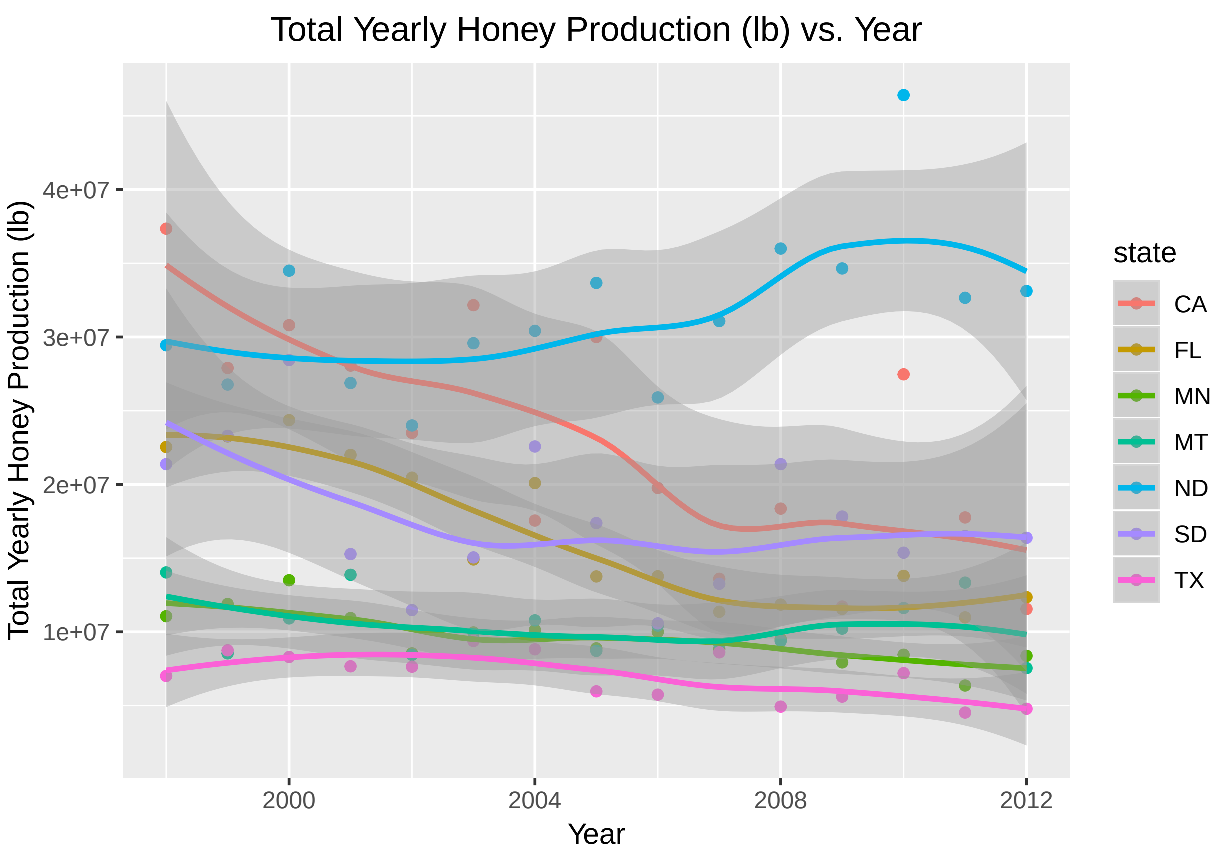
```
## # A tibble: 44 x 4
##    state avg_production avg_yield num_colonies_1lb
##    <chr>          <dbl>     <dbl>            <dbl>
##  1 AL           825467.      67.5           12223.
##  2 AR          2810400       73.9           38013.
##  3 AZ          2032267.      60.1           33834.
##  4 CA         23169000       55.8          415215.
##  5 CO          1750600       62.8           27876.
##  6 FL         16469867.      83.1          198273.
##  7 GA          3299933.      54.7           60365.
##  8 HI           843133.      98              8603.
##  9 IA          2080000       65.7           31643.
## 10 ID          4410667.      44            100242.
## # ... with 34 more rows
```

The 4 plots below are all scatterplots of Total Honey Production in Pounds vs Year for all states. For majority of the states, the plots illustrate a gradual decrease in total honey production between the years 1998 and 2012.

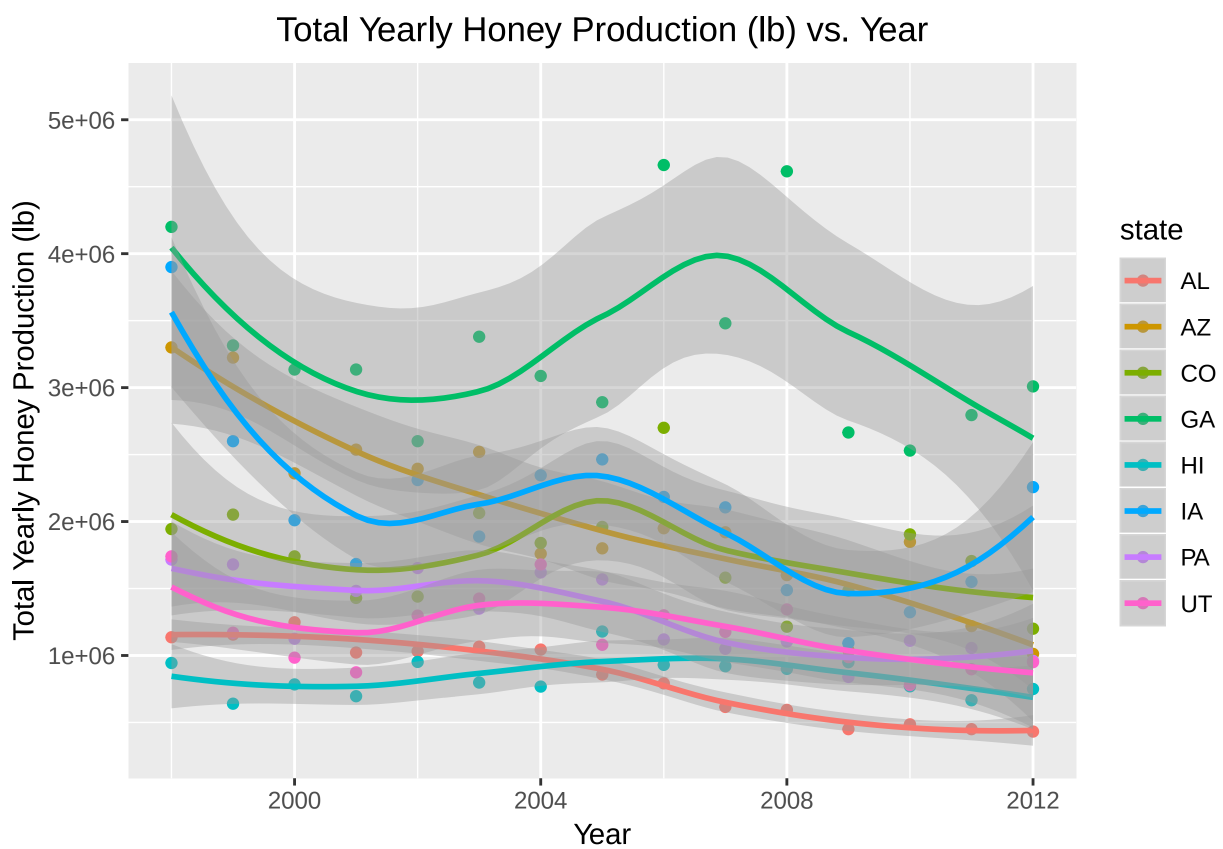
geom_point() is used to create the scatterplot and geom_smooth() is used to create the trend line. labs() and ggtitle() is used to customize the names of the axises and title of the plot. In addition, theme(plot.title = element_text(hjust = 0.5)) helps center the title of the plot.

Here is a ggplot2 reference sheet: https://ggplot2.tidyverse.org/reference/

```r
df %>%
  filter(state %in% c("CA", "FL", "ND", "MT", "MN", "SD", "TX")) %>%
  ggplot(aes(x = year, y = total_production, color = state)) + geom_point() +
  geom_smooth(method = loess) + labs(x = "Year", y = "Total Yearly Honey Production (lb)") +
  ggtitle("Total Yearly Honey Production (lb) vs. Year") +
  theme(plot.title = element_text(hjust = 0.5))
```



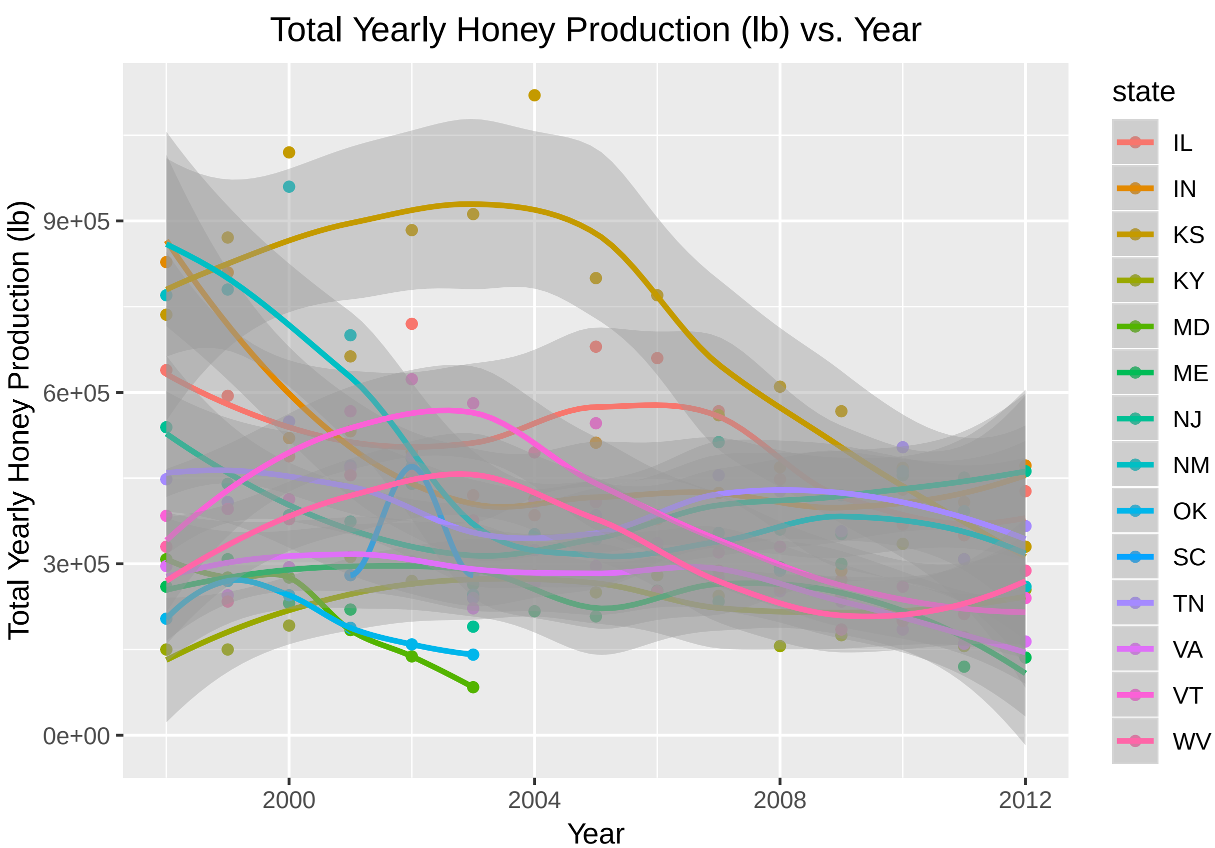

```r
df %>%
  filter(state %in% c("AZ", "GA", "IA", "UT", "PA", "CO", "AL", "HI")) %>%
  ggplot(aes(x = year, y = total_production, color = state)) + geom_point() +
  geom_smooth(method = loess) + labs(x = "Year", y = "Total Yearly Honey Production (lb)") +
  ggtitle("Total Yearly Honey Production (lb) vs. Year") +
  theme(plot.title = element_text(hjust = 0.5))
```



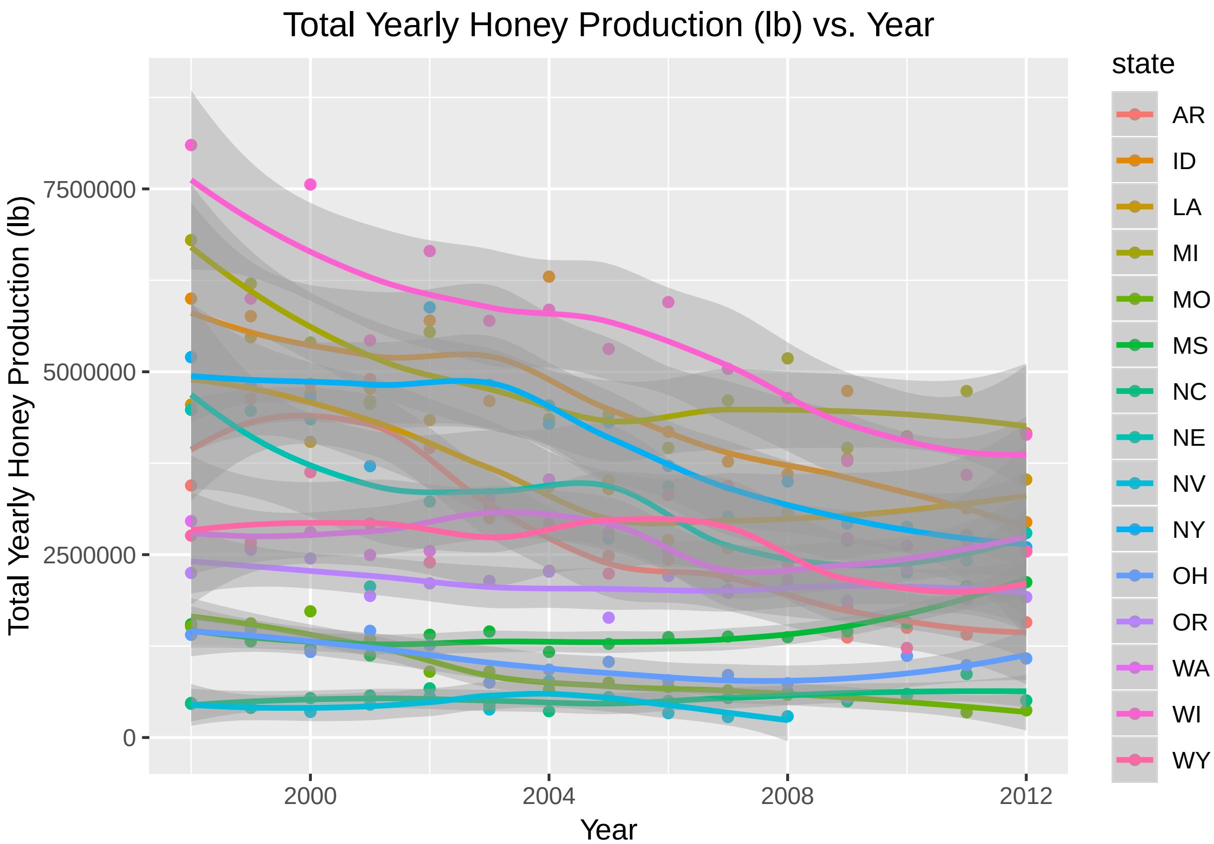

```r
df %>%
  filter(state %in% c("KS", "IL", "IN", "MD", "OK", "ME", "VA", "SC", "TN", "VT",
                      "WV", "KY", "NM", "NJ")) %>%
  ggplot(aes(x = year, y = total_production, color = state)) + geom_point() +
  geom_smooth(method = loess) + labs(x = "Year", y = "Total Yearly Honey Production (lb)") +
  ggtitle("Total Yearly Honey Production (lb) vs. Year") +
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : Chernobyl! trL>n 6

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : Chernobyl! trL>n 6

## Warning in sqrt(sum.squares/one.delta): NaNs produced

## Warning in stats::qt(level/2 + 0.5, pred$df): NaNs produced

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : Chernobyl! trL>n 6

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : Chernobyl! trL>n 6

## Warning in sqrt(sum.squares/one.delta): NaNs produced

## Warning in stats::qt(level/2 + 0.5, pred$df): NaNs produced

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 2001

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1.01

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1.0201

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : span too small.
## fewer data values than degrees of freedom.

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used
## at 2001

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 1.01

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal
## condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other
## near singularities as well. 1.0201
```



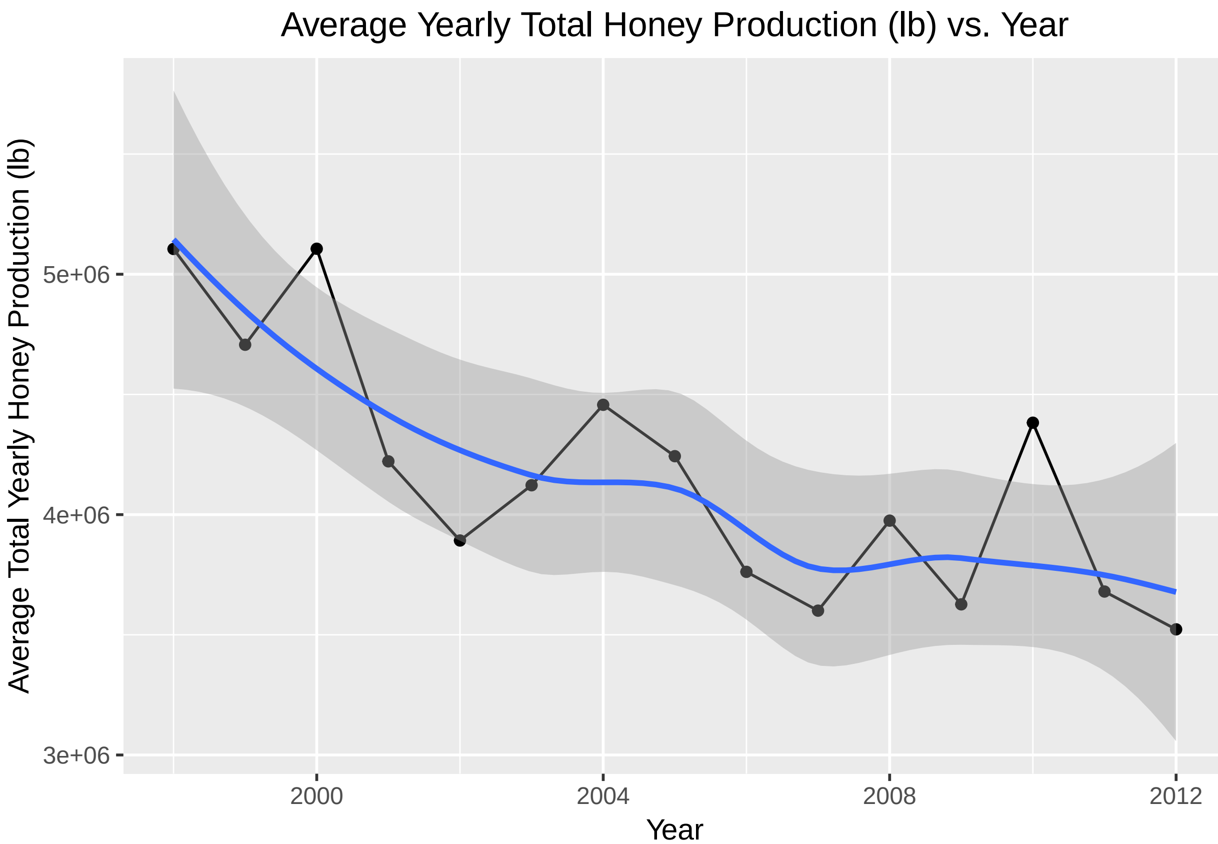

```
df %>%
  filter(state %in% c("LA", "ID", "AR", "OR", "WI", "WY", "WA", "NY", "NE", "MI",
                      "MS", "MO", "NC", "NV", "OH")) %>%
  ggplot(aes(x = year, y = total_production, color = state)) + geom_point() +
  geom_smooth(method = loess) + labs(x = "Year", y = "Total Yearly Honey Production (lb)") +
  ggtitle("Total Yearly Honey Production (lb) vs. Year") +
  theme(plot.title = element_text(hjust = 0.5))
```



Below is a scatterplot of the Average Total Honey Production in Pounds vs. Year. geom_line() connects each point in order of the variable on the x-axis, highlighting the changes between each year in average honey production (lb).

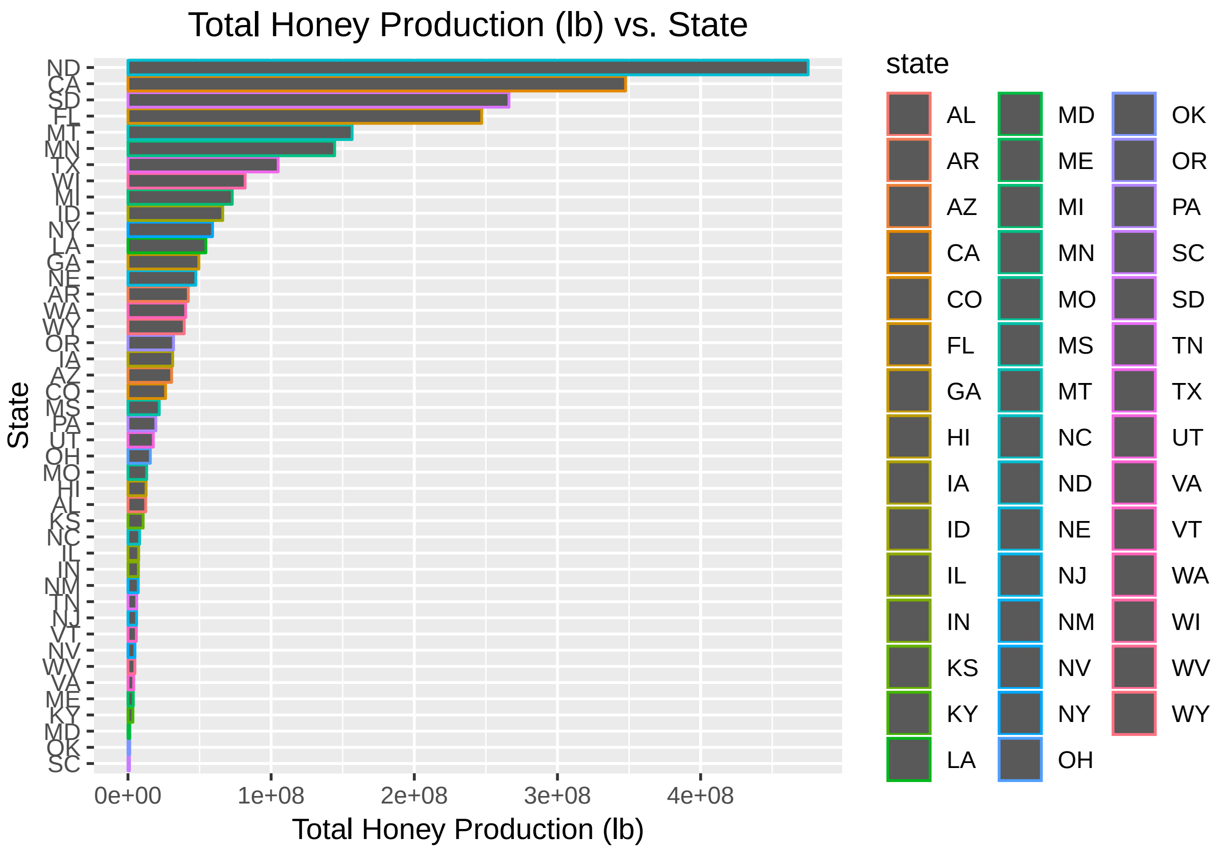
Even though the total average honey production increases a little during 1999-2000, 2002-2004, 2007-2008, an 2009-2010, as seen through the black line, overall, there is a general decrease in total average honey production from 1998 to 2012 as seen through the blue line.

```
df %>%
  group_by(year) %>%
  summarise(avg_total_production = mean(total_production)) %>%
  ggplot(aes(x = year, y = avg_total_production)) + geom_point() + geom_line() +
  geom_smooth(method = loess) +
  labs(x = "Year", y = "Average Total Yearly Honey Production (lb)") +
  ggtitle("Average Yearly Total Honey Production (lb) vs. Year") +
  theme(plot.title = element_text(hjust = 0.5))
```



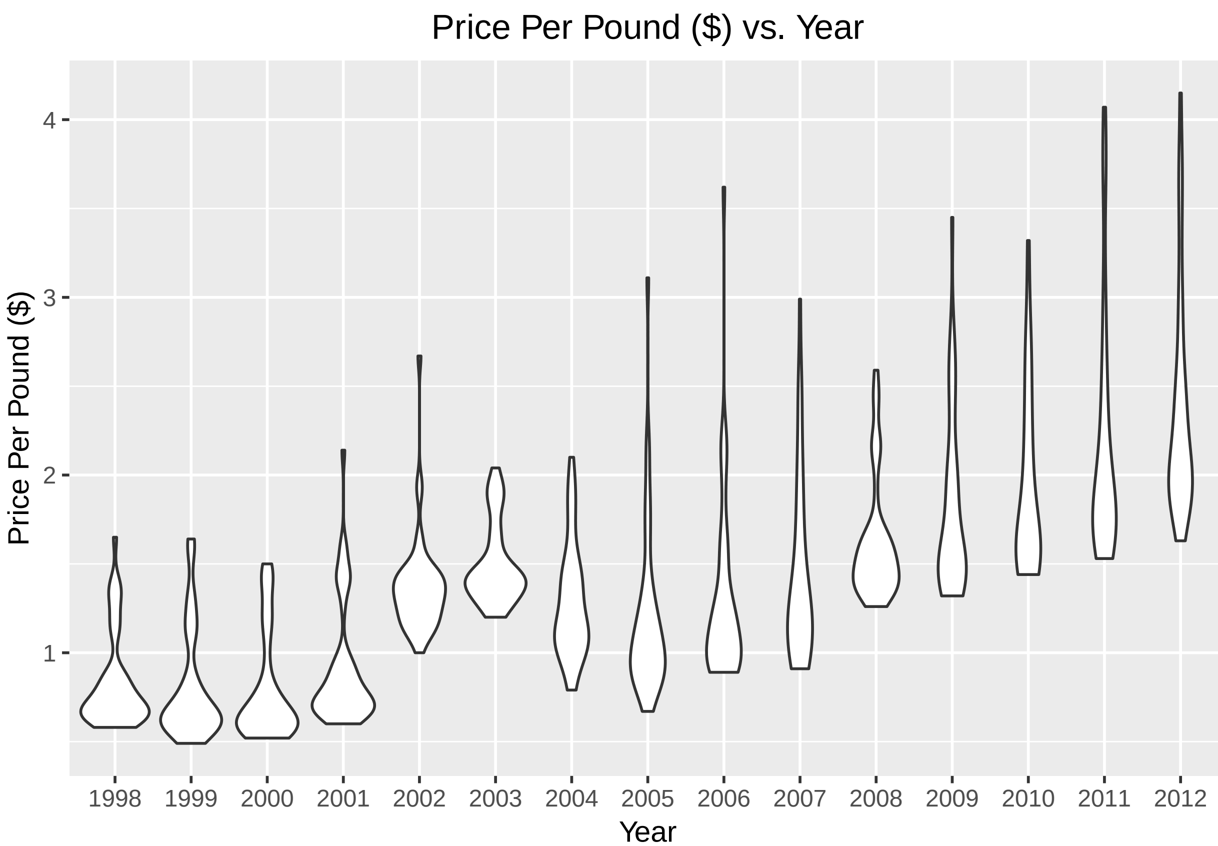
Using geom_bar(), below is a bar graph of State vs. Total Honey Production in Pounds. The Total Honey Production number is the sum of all the honey a state has produced between 1998 and 2012. North Dakota (ND) has produced the most amount of honey, while South Carolina (SC) has produced the least amount of honey over time.

```
df %>%
  group_by(state) %>%
  summarise(total_amount = sum(total_production)) %>%
  ggplot(mapping = aes(y = total_amount, x = reorder(state, total_amount),
                       color = state, reorder(total_amount))) +
  geom_bar(stat = "identity") + labs(y = "Total Honey Production (lb)", x = "State") +
  ggtitle("Total Honey Production (lb) vs. State") +
  theme(plot.title = element_text(hjust = 0.5)) + coord_flip()
```



Using geom_violin(), below is a violin plot of Price Per Pound ($) vs. Year. A violin plot is a type of plot that displays the distribution of the variable in the y-axis (price_per_lb) for each value of the variable in the x-axis (year). Over the years, the price of honey per pound has gradually increased from less than 1 dollar to prices over 4 dollars.

```
df %>%
  ggplot(aes(x = factor(year), y = price_per_lb)) + geom_violin() +
  labs(x = "Year", y = "Price Per Pound ($)") +
  ggtitle("Price Per Pound ($) vs. Year") + theme(plot.title = element_text(hjust = 0.5))
```

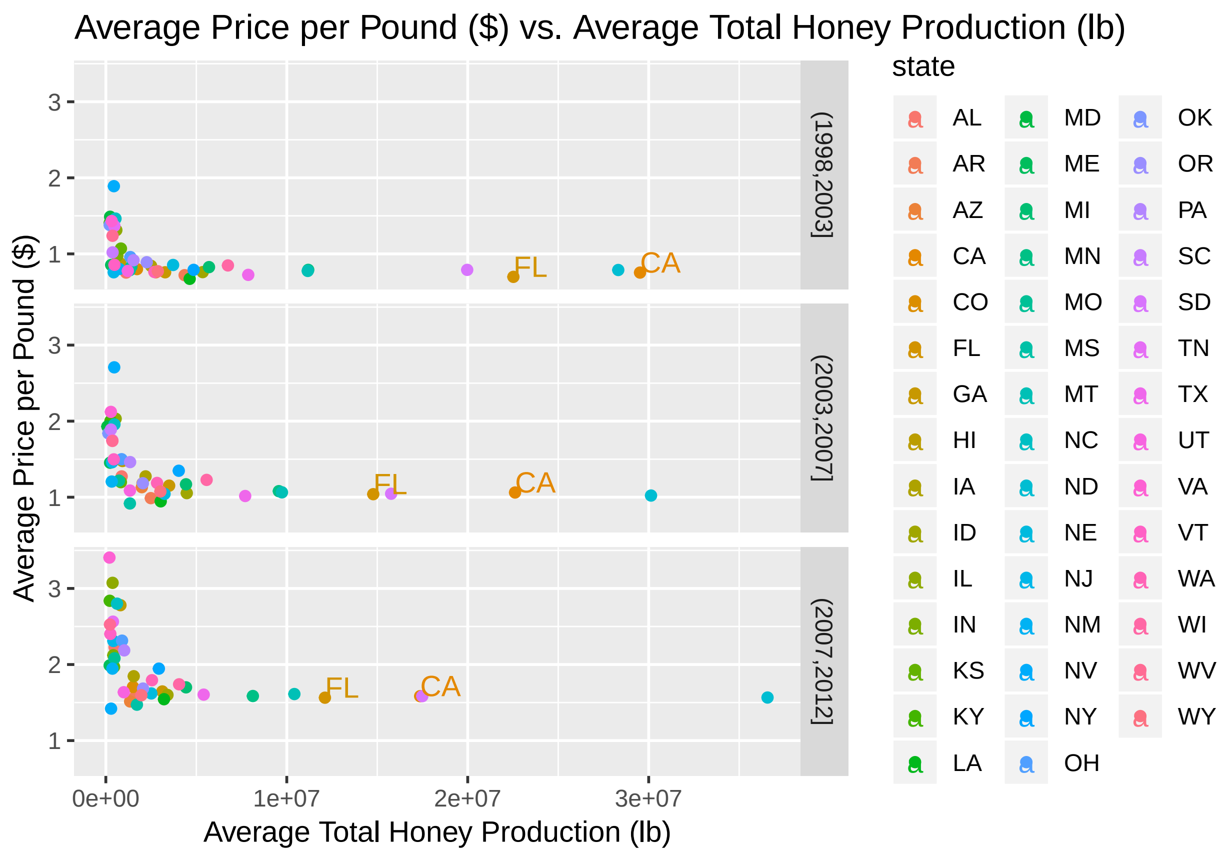


Below, using the cut() function, discretization is applied where year is split into 3 equal bins or intervals: (1998-2003], (2003-2007], (2007-2012]. These year intervals are then added into a new column, called year_interavls, in the data frame.

The scatterplot shows the Average Price Per Pound ($) vs. the Average Total Honey Production (lb) in each year interval, using the facet_grid() function. From the 1st interval to the 3rd interval, there is an increasing trend in the average price of honey, while there is a decreasing trend in the average total honey production. Florida (FL) and California (CA) are labeled to highlight this trend using the geom_text() function. The price increases approximately 50 cents per interval and the honey production decreases several million pounds per interval.

Function Documentation: - cut() ~ https://www.rdocumentation.org/packages/base/versions/3.6.0/topics/cut - facet_grid() ~ https://ggplot2.tidyverse.org/reference/facet_grid.html - geom_text() ~ https://www.rdocumentation.org/packages/ggplot2/versions/0.9.1/topics/geom_text

```r
# Discretize year into 3 intervals
df$year_intervals <- cut(df$year, breaks = 3)

df %>%
  group_by(state, year_intervals) %>%
  summarise(avg_production = mean(total_production), avg_price = mean(price_per_lb)) %>%
  ggplot(aes(x = avg_production, y = avg_price, color = state, label = state)) + geom_point() +
  geom_text(aes(label = ifelse(state == 'CA' | state == 'FL', state, '')),
            hjust = 0, vjust = 0) + facet_grid(rows = vars(year_intervals)) +
  labs(x = "Average Total Honey Production (lb)", y = "Average Price per Pound ($)") +
  ggtitle("Average Price per Pound ($) vs. Average Total Honey Production (lb)")
```



Here is a dataframe of what was plotted above.

```r
df %>%
  group_by(state, year_intervals) %>%
  summarise(avg_production = mean(total_production), avg_price = mean(price_per_lb)) %>%
  arrange(state)
```

```
## # A tibble: 129 x 4
## # Groups:   state [44]
##    state year_intervals avg_production avg_price
##    <chr> <fct>                   <dbl>     <dbl>
##  1 AL    (1998,2003]           1118800     0.754
##  2 AL    (2003,2007]            875200     1.27
##  3 AL    (2007,2012]            482400     2.23
##  4 AR    (1998,2003]           4353800     0.72
##  5 AR    (2003,2007]           2487200     0.988
##  6 AR    (2007,2012]           1590200     1.57
##  7 AZ    (1998,2003]           2763000     0.758
##  8 AZ    (2003,2007]           1990000     1.13
##  9 AZ    (2007,2012]           1343800     1.51
## 10 CA    (1998,2003]          29522000     0.754
## # ... with 119 more rows
```

Linear Regression

Linear regression is a very useful technique for data analysis. It allows for constructing confidence intervals, utilizing hypothesis testing for relationships between variables, and providing continuous outcomes of interest.

From http://r-statistics.co/Linear-Regression.html , "linear regression is used to predict the value of an outcome variable Y based on one or more predictor variables X. The aim is to establish a linear relationship (a mathematical formula) between the predictor variable(s) and the response variable, so that, we can use this formula to estimate the value of the response Y, when only the predictors (Xs) values are known."

For more information on Linear Regression and Linear Models: - https://www.statisticssolutions.com/what-is-linear-regression/ - http://www.stat.yale.edu/Courses/1997-98/101/linreg.htm - https://data.princeton.edu/r/linearmodels - This link helps explains how to fit a model, examine a fit, extract results, and much
more.

```
# Fit a linear regression model for Total Honey Production (lb) vs. Year
df_fit_honey <- lm(total_production~year, data = df)
df_fit_honey
```

```
##
## Call:
## lm(formula = total_production ~ year, data = df)
##
## Coefficients:
## (Intercept)         year
##   181765231       -88583
```

```
df_fit_honey_stats <- df_fit_honey %>%
  tidy()
df_fit_honey_stats
```

```
## # A tibble: 2 x 5
##   term           estimate  std.error statistic p.value
##   <chr>             <dbl>      <dbl>     <dbl>   <dbl>
## 1 (Intercept)  181765231. 127773490.      1.42   0.155
## 2 year            -88583.     63732.     -1.39   0.165
```

```
cat("On average, the total honey production decreased by", df_fit_honey_stats$estimate[2],
    "pounds per year from 1998 to 2012.")
```

```
## On average, the total honey production decreased by -88582.63 pounds per year from 1998 to 2012.
```

If there was a null hypothesis of no relationship between total honey production and year, I would reject that null hypothesis because there is a relationship between the two variables: Over time, the total honey production decreases by around -88582.63 per year.

Now, we augment the df_fit_honey, in which columns, such as predictions, residuals, etc. are added. Residuals is the difference between the observed value of the dependent variable and the predicted value. Fitted values are also known as predicted values. One way to check if your linear regression model is appropriate is to plot a graph of Residuals vs. Fitted Values. This graph will check for the linearity assumption. If the regression model is appropriate, the mean of residuals will be approximately 0.

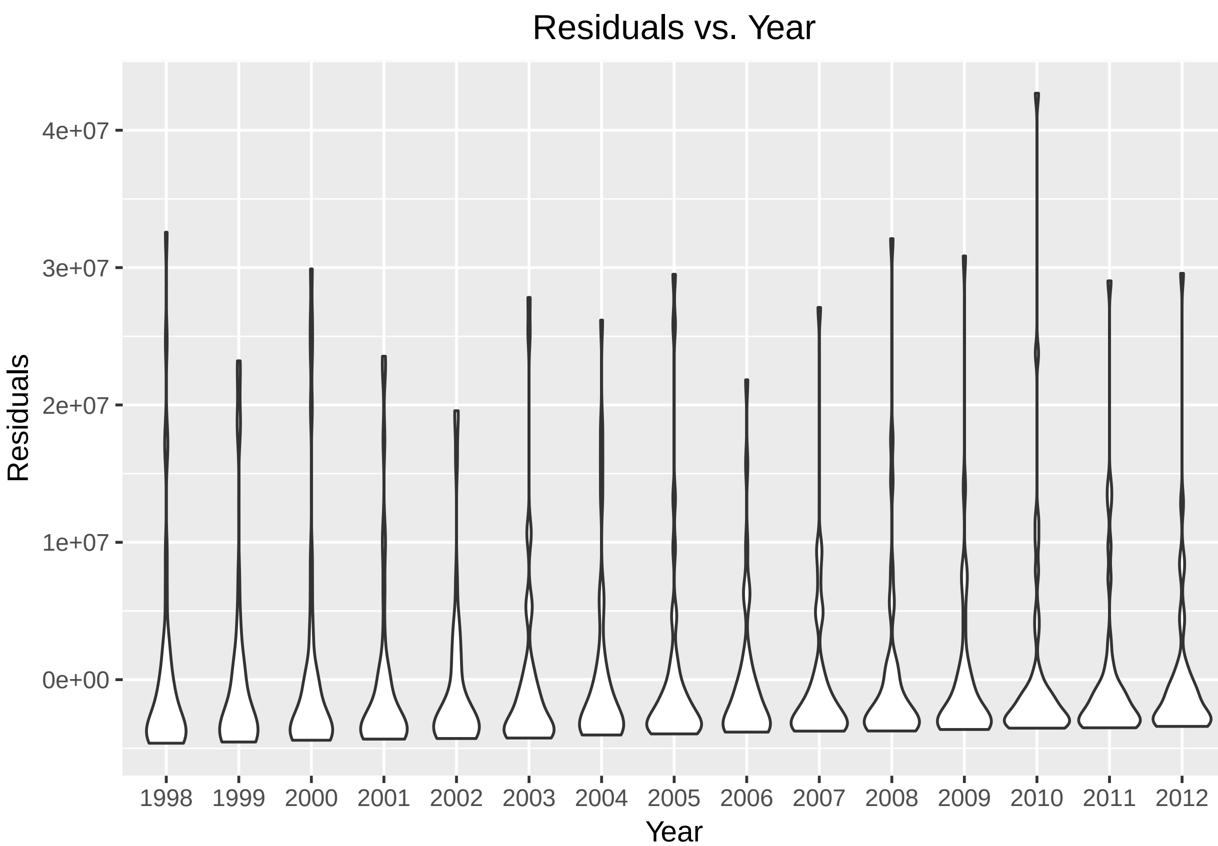
The augment() function Documentation: - https://www.rdocumentation.org/packages/broom/versions/0.4.3/topics/augment Residuals: - http://www.r-tutor.com/elementary-statistics/simple-linear-regression/residual-plot

```
aug_df_honey <- df_fit_honey %>%
  augment()
```

```r
aug_df_honey
```

```
## # A tibble: 626 x 9
##    total_production  year .fitted .se.fit  .resid    .hat .sigma .cooksd
##               <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
##  1          1136000  1998  4.78e6 516685. -3.64e6 0.00564 6.88e6 7.99e-4
##  2          3300000  1998  4.78e6 516685. -1.48e6 0.00564 6.88e6 1.32e-4
##  3          3445000  1998  4.78e6 516685. -1.33e6 0.00564 6.88e6 1.07e-4
##  4         37350000  1998  4.78e6 516685.  3.26e7 0.00564 6.76e6 6.40e-2
##  5          1944000  1998  4.78e6 516685. -2.83e6 0.00564 6.88e6 4.84e-4
##  6         22540000  1998  4.78e6 516685.  1.78e7 0.00564 6.85e6 1.90e-2
##  7          4200000  1998  4.78e6 516685. -5.77e5 0.00564 6.88e6 2.01e-5
##  8           944000  1998  4.78e6 516685. -3.83e6 0.00564 6.88e6 8.86e-4
##  9          6000000  1998  4.78e6 516685.  1.22e6 0.00564 6.88e6 9.02e-5
## 10           639000  1998  4.78e6 516685. -4.14e6 0.00564 6.88e6 1.03e-3
## # ... with 616 more rows, and 1 more variable: .std.resid <dbl>
```

```r
# A violin plot of model Residuals vs. Year
aug_df_honey %>%
  ggplot(aes(x = factor(year), y = .resid)) + geom_violin() +
  labs(x = "Year", y = "Residuals") + ggtitle("Residuals vs. Year") +
  theme(plot.title = element_text(hjust = 0.5))
```

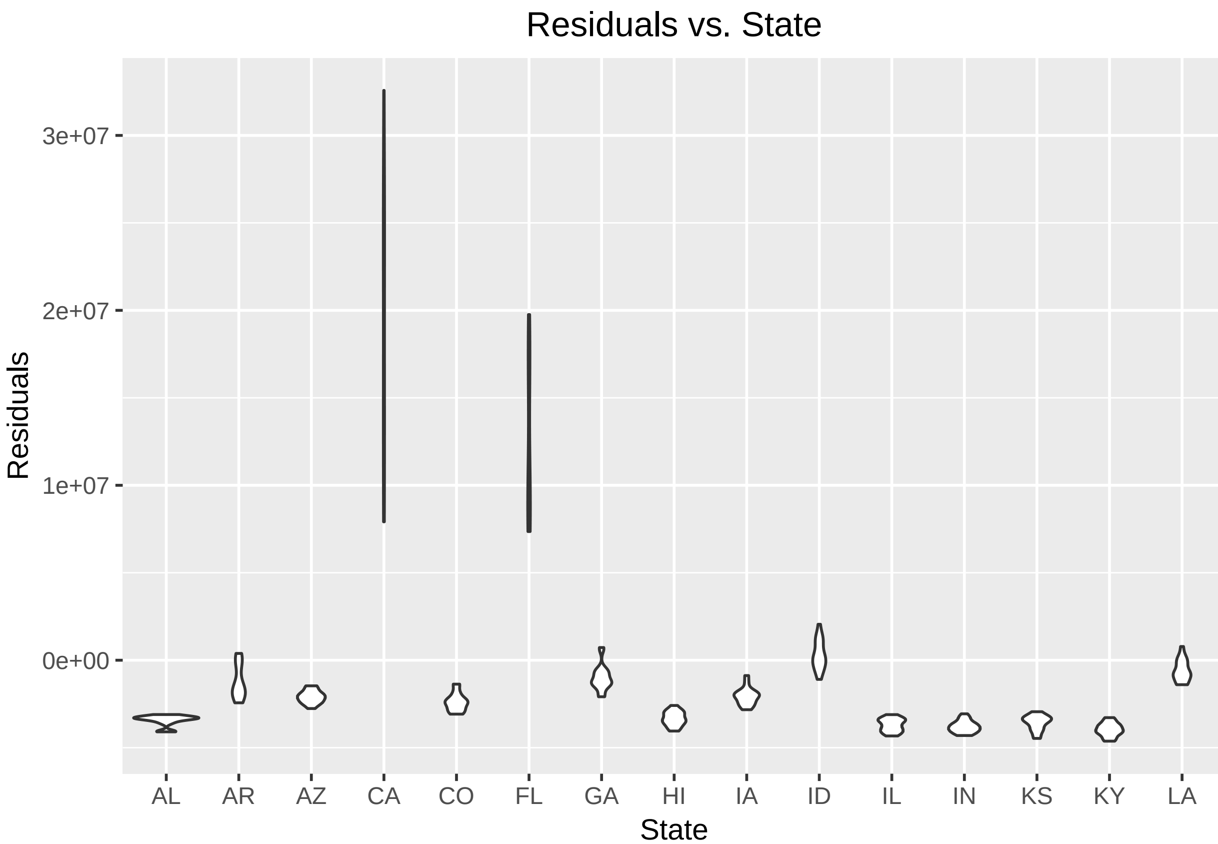


The 3 graphs below are violin plots of model Residuals vs. State. These 3 plots show that there is a dependence between model residuals and state because each violin is different. This suggests that when performing a regression analysis of total honey production across time, it is important to consider the state variable.

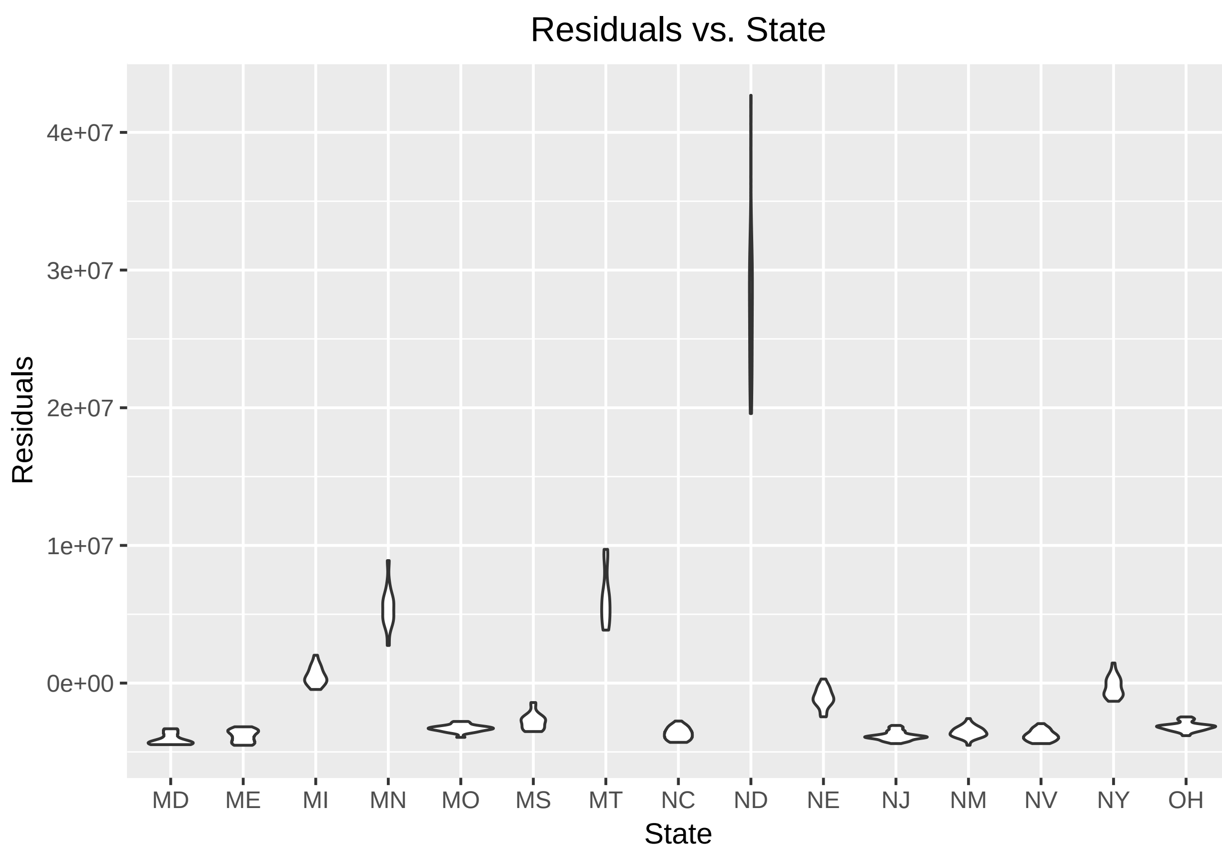
```r
aug_df_honey <- aug_df_honey %>%
  inner_join(df, by = "total_production")
aug_df_honey
```

```
## # A tibble: 930 x 17
##    total_production year.x .fitted .se.fit  .resid    .hat .sigma .cooksd
##               <dbl>  <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
##  1          1136000   1998  4.78e6 516685. -3.64e6 0.00564 6.88e6 7.99e-4
##  2          3300000   1998  4.78e6 516685. -1.48e6 0.00564 6.88e6 1.32e-4
##  3          3445000   1998  4.78e6 516685. -1.33e6 0.00564 6.88e6 1.07e-4
##  4         37350000   1998  4.78e6 516685.  3.26e7 0.00564 6.76e6 6.40e-2
##  5          1944000   1998  4.78e6 516685. -2.83e6 0.00564 6.88e6 4.84e-4
##  6         22540000   1998  4.78e6 516685.  1.78e7 0.00564 6.85e6 1.90e-2
##  7          4200000   1998  4.78e6 516685. -5.77e5 0.00564 6.88e6 2.01e-5
##  8           944000   1998  4.78e6 516685. -3.83e6 0.00564 6.88e6 8.86e-4
##  9          6000000   1998  4.78e6 516685.  1.22e6 0.00564 6.88e6 9.02e-5
## 10          6000000   1998  4.78e6 516685.  1.22e6 0.00564 6.88e6 9.02e-5
## # ... with 920 more rows, and 9 more variables: .std.resid <dbl>,
## #   state <chr>, num_colonies <dbl>, yield_per_colony_lb <dbl>,
## #   stocks <dbl>, price_per_lb <dbl>, production_value <dbl>,
## #   year.y <dbl>, year_intervals <fct>
```

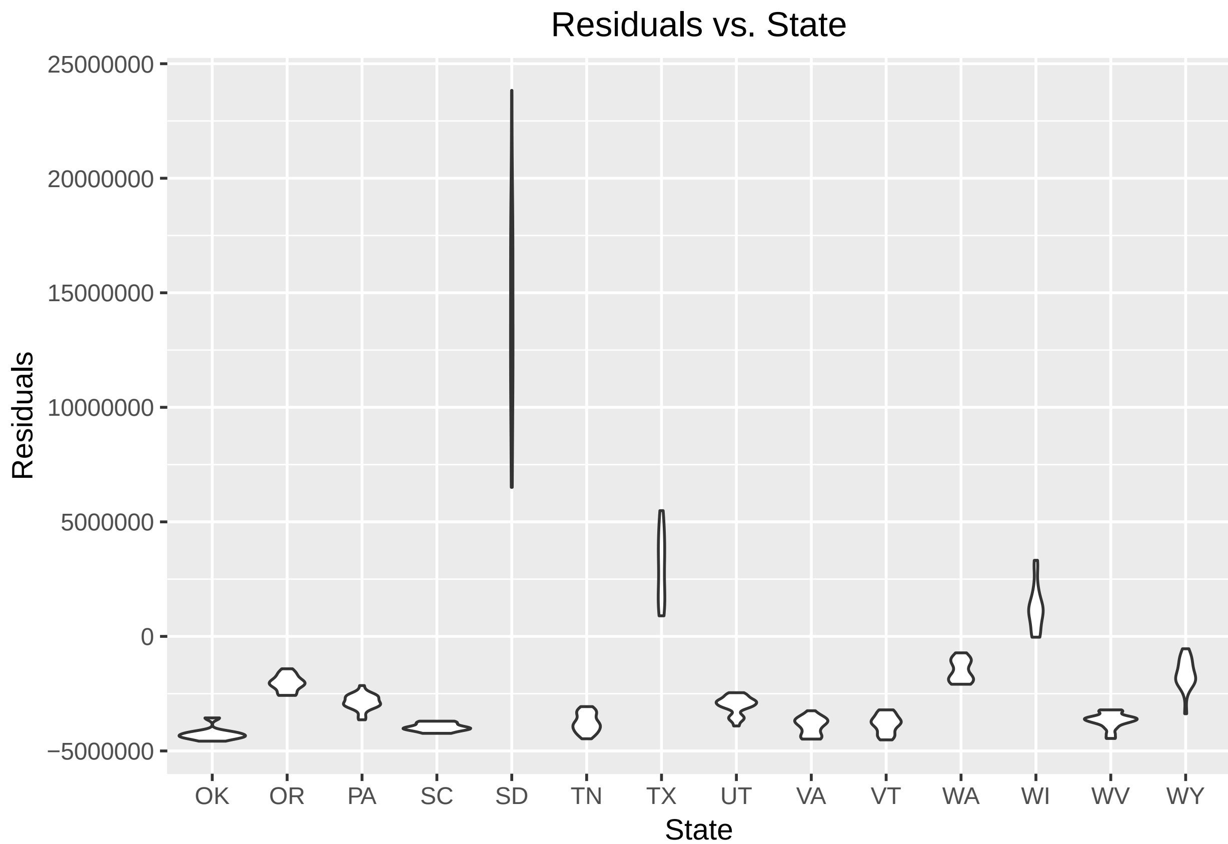
```r
aug_df_honey %>%
  filter(state %in% c("AL", "AR", "AZ", "CA", "CO", "FL", "GA","HI", "IA",
                      "ID", "IL", "IN", "KS", "KY", "LA")) %>%
  ggplot(aes(x = factor(state), y = .resid)) + geom_violin() +
  labs(x = "State", y = "Residuals") + ggtitle("Residuals vs. State") +
  theme(plot.title = element_text(hjust = 0.5))
```




```r
aug_df_honey %>%
  filter(state %in% c("MD", "ME", "MI", "MN", "MO", "MS", "MT", "NC",
                      "ND", "NE", "NJ", "NM", "NV", "NY", "OH")) %>%
  ggplot(aes(x = factor(state), y = .resid)) + geom_violin() +
  labs(x = "State", y = "Residuals") + ggtitle("Residuals vs. State") +
  theme(plot.title = element_text(hjust = 0.5))
```




```r
aug_df_honey %>%
  filter(state %in% c("OK", "OR", "PA", "SC", "SD", "TN", "TX", "UT",
                      "VA", "VT", "WA", "WI", "WV", "WY")) %>%
  ggplot(aes(x = factor(state), y = .resid)) + geom_violin() +
  labs(x = "State", y = "Residuals") + ggtitle("Residuals vs. State") +
  theme(plot.title = element_text(hjust = 0.5))
```



Here, we fit a linear regression model for total honey production including a term for an interaction between state and year.

```
state_honey_fit <- lm(total_production~year*state, data = df)
state_honey_fit
```

```
##
## Call:
## lm(formula = total_production ~ year * state, data = df)
##
## Coefficients:
##  (Intercept)         year     stateAR      stateAZ      stateCA
##    1.278e+08   -6.331e+04   3.775e+08    1.606e+08    2.572e+09
##      stateCO      stateFL     stateGA      stateHI      stateIA
##   -7.564e+07    1.885e+09  -6.330e+07   -1.269e+08    5.899e+07
##      stateID      stateIL     stateIN      stateKS      stateKY
##    2.854e+08   -9.486e+07  -8.319e+07   -4.658e+07   -1.319e+08
##      stateLA      stateMD     stateME      stateMI      stateMN
##    1.515e+08   -3.462e+07  -1.110e+08    1.490e+08    5.170e+08
##      stateMO      stateMS     stateMT      stateNC      stateND
##    5.800e+07   -2.093e+08   6.903e+07   -1.498e+08   -1.429e+09
##      stateNE      stateNJ     stateNM      stateNV      stateNY
##    1.464e+08   -1.288e+08  -5.643e+07   -1.094e+08    2.626e+08
##      stateOH      stateOK     stateOR      statePA      stateSC
##   -5.083e+07   -8.727e+07  -7.754e+07   -2.317e+07   -1.274e+08
##      stateSD      stateTN     stateTX      stateUT      stateVA
##    7.261e+08   -1.163e+08   3.615e+08   -6.593e+07   -1.068e+08
##      stateVT      stateWA     stateWI      stateWV      stateWY
```

```
##   -8.529e+07    -7.560e+07     4.027e+08    -1.030e+08     3.111e+07
## year:stateAR  year:stateAZ  year:stateCA  year:stateCO  year:stateFL
##   -1.873e+05    -7.949e+04    -1.272e+06     3.819e+04    -9.323e+05
## year:stateGA  year:stateHI  year:stateIA  year:stateID  year:stateIL
##    3.281e+04     6.329e+04    -2.880e+04    -1.406e+05     4.715e+04
## year:stateIN  year:stateKS  year:stateKY  year:stateLA  year:stateMD
##    4.132e+04     2.317e+04     6.547e+04    -7.415e+04     1.686e+04
## year:stateME  year:stateMI  year:stateMN  year:stateMO  year:stateMS
##    5.510e+04    -7.233e+04    -2.535e+05    -2.890e+04     1.047e+05
## year:stateMT  year:stateNC  year:stateND  year:stateNE  year:stateNJ
##   -2.964e+04     7.457e+04     7.282e+05    -7.188e+04     6.401e+04
## year:stateNM  year:stateNV  year:stateNY  year:stateOH  year:stateOK
##    2.797e+04     5.436e+04    -1.294e+05     2.546e+04     4.317e+04
## year:stateOR  year:statePA  year:stateSC  year:stateSD  year:stateTN
##    3.932e+04     1.179e+04     6.331e+04    -3.537e+05     5.777e+04
## year:stateTX  year:stateUT  year:stateVA  year:stateVT  year:stateWA
##   -1.772e+05     3.306e+04     5.298e+04     4.232e+04     3.863e+04
## year:stateWI  year:stateWV  year:stateWY
##   -1.985e+05     5.111e+04    -1.462e+04
```

```r
state_honey_fit_stats <- state_honey_fit %>%
  tidy()
state_honey_fit_stats
```

```
## # A tibble: 88 x 5
##    term           estimate  std.error statistic  p.value
##    <chr>             <dbl>      <dbl>     <dbl>    <dbl>
##  1 (Intercept)   127770610. 178592842.     0.715 4.75e- 1
##  2 year             -63314.     89074.    -0.711 4.78e- 1
##  3 stateAR       377478469. 252568420.     1.49  1.36e- 1
##  4 stateAZ       160589979. 252568420.     0.636 5.25e- 1
##  5 stateCA      2572073390. 252568420.    10.2   2.16e-22
##  6 stateCO       -75644385. 252568420.    -0.300 7.65e- 1
##  7 stateFL      1884977507. 252568420.     7.46  3.42e-13
##  8 stateGA       -63303855. 252568420.    -0.251 8.02e- 1
##  9 stateHI      -126884512. 252568420.    -0.502 6.16e- 1
## 10 stateIA        58991373. 252568420.     0.234 8.15e- 1
## # ... with 78 more rows
```

The state_honey_fit_stats allows us to find out how much total honey production (in pounds) increases or decreases each year on average for each state. For instance, below is how you would calculate how much total honey production changes per year for AL, HI, and CA.

```r
cat("AL:", state_honey_fit_stats$estimate[2], "\n")
```

```
## AL: -63314.29
```

```r
cat("HI:", (state_honey_fit_stats$estimate[2] + state_honey_fit_stats$estimate[9]), "\n")
```

```
## HI: -126947826
```

```r
cat("CA:", (state_honey_fit_stats$estimate[2] + state_honey_fit_stats$estimate[5]))
```

```
## CA: 2572010076
```

Below, we perform a F-test that compares the Linear Regression with and without the state variable using the anova() function. The anova() function Documentation: -https://www.rdocumentation.org/packages/car/versions/3.0-2/topics/Anova

```r
comp_honey <- anova(df_fit_honey, state_honey_fit)
comp_honey
```

```
## Analysis of Variance Table
##
## Model 1: total_production ~ year
## Model 2: total_production ~ year * state
##   Res.Df        RSS Df Sum of Sq      F    Pr(>F)
## 1    624 2.9526e+16
## 2    538 1.1952e+15 86 2.833e+16 148.29 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The interaction model (or Model #2) is significantly better than the year-only model because the p-value for Model #2 is ver low at 2.2e-16, meaning that adding state to the model significantly improved fit over Model #1

Here are some links for more information on how to interpret: - https://stats.stackexchange.com/questions/115304/interpreting-output-from-anova-when-using-lm-as-input - https://bookdown.org/ndphillips/YaRrr/comparing-regression-models-with-anova.html

Below, we'll do the same but with the price of honey per pound.

```r
# Fit a linear regression model for Price of Honey per Pound ($) vs. Year
df_fit_price <- lm(price_per_lb~year, data = df)
df_fit_price
```

```
##
## Call:
## lm(formula = price_per_lb ~ year, data = df)
##
## Coefficients:
## (Intercept)         year
##   -204.3926       0.1027
```

```r
df_fit_price_stats <- df_fit_price %>%
  tidy()
df_fit_price_stats
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept) -204.      8.55        -23.9 3.38e-90
## 2 year           0.103   0.00426      24.1 4.30e-91
```

```r
cat("On average, the price per pound of honey increased by", df_fit_price_stats$estimate[2],
    "cents per year from 1998 to 2012.")
```

```
## On average, the price per pound of honey increased by 0.1026514 cents per year from 1998 to 2012.
```

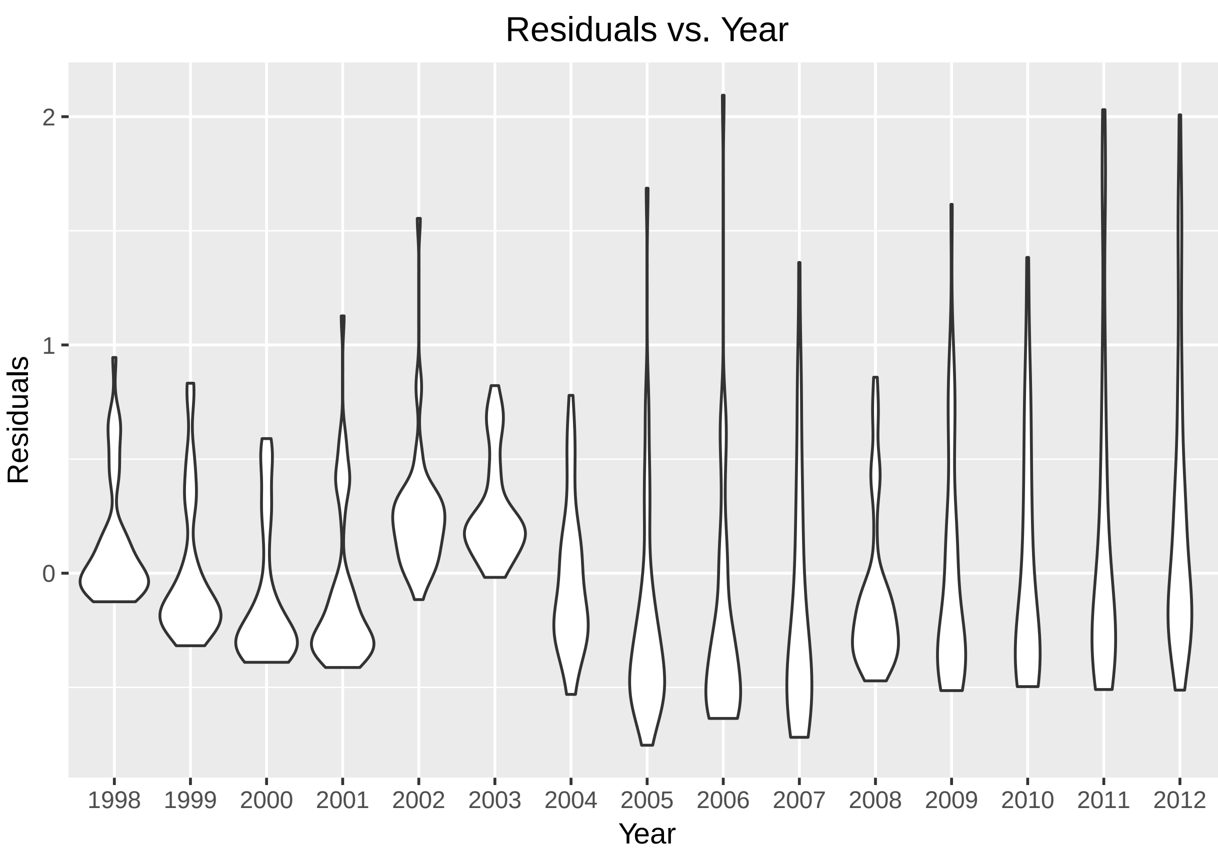
If there was a null hypothesis of no relationship between the price of honey per pound and year, I would reject that null hypothesis because there is a relationship between the two variables: Over time, the price of honey per pound increases by approximately $0.10.

```r
aug_df_price <- df_fit_price %>%
  augment()
aug_df_price
```

```
## # A tibble: 626 x 9
```

```
##    price_per_lb  year .fitted .se.fit   .resid    .hat .sigma .cooksd
##           <dbl> <dbl>   <dbl>   <dbl>    <dbl>   <dbl>  <dbl>   <dbl>
##  1         0.72  1998   0.705  0.0346  0.0151  0.00564  0.461 3.05e-6
##  2         0.64  1998   0.705  0.0346 -0.0649  0.00564  0.461 5.68e-5
##  3         0.59  1998   0.705  0.0346 -0.115   0.00564  0.460 1.78e-4
##  4         0.62  1998   0.705  0.0346 -0.0849  0.00564  0.461 9.72e-5
##  5         0.7   1998   0.705  0.0346 -0.00495 0.00564  0.461 3.30e-7
##  6         0.64  1998   0.705  0.0346 -0.0649  0.00564  0.461 5.68e-5
##  7         0.69  1998   0.705  0.0346 -0.0149  0.00564  0.461 3.01e-6
##  8         0.77  1998   0.705  0.0346  0.0651  0.00564  0.461 5.70e-5
##  9         0.65  1998   0.705  0.0346 -0.0549  0.00564  0.461 4.07e-5
## 10         1.19  1998   0.705  0.0346  0.485   0.00564  0.460 3.17e-3
## # ... with 616 more rows, and 1 more variable: .std.resid <dbl>
```

```
# A violin plot of model Residuals vs. Year
aug_df_price %>%
  ggplot(aes(x = factor(year), y = .resid)) + geom_violin() +
  labs(x = "Year", y = "Residuals") + ggtitle("Residuals vs. Year") +
  theme(plot.title = element_text(hjust = 0.5))
```



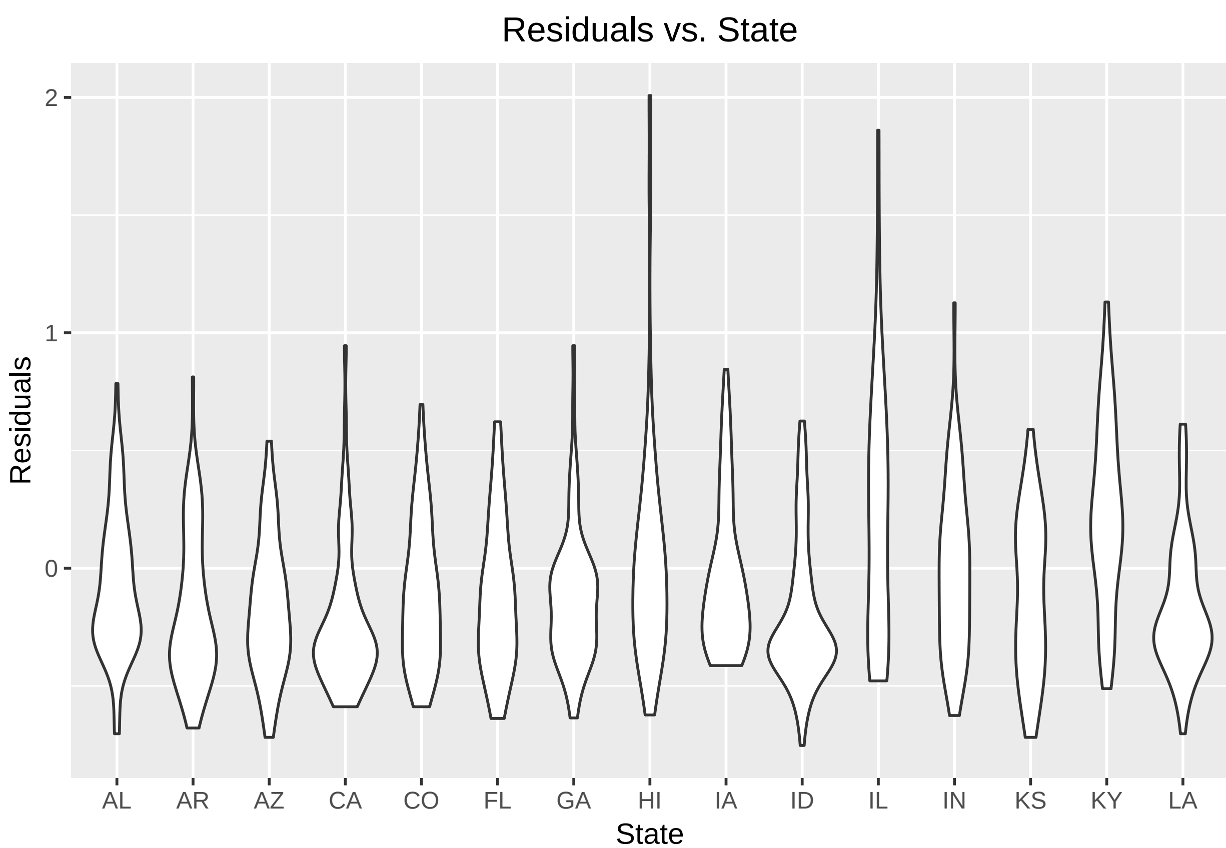
The 3 graphs below are violin plots of model Residuals vs. State. These 3 plots show that there is a dependence between model residuals and state because each violin is different. This suggests that when performing a regression analysis of price of honey per pound across time, it is important to consider the state variable.

```
aug_df_price <- aug_df_price %>%
  inner_join(df, by = "price_per_lb")
aug_df_price
```

```
## # A tibble: 2,858 x 17
```

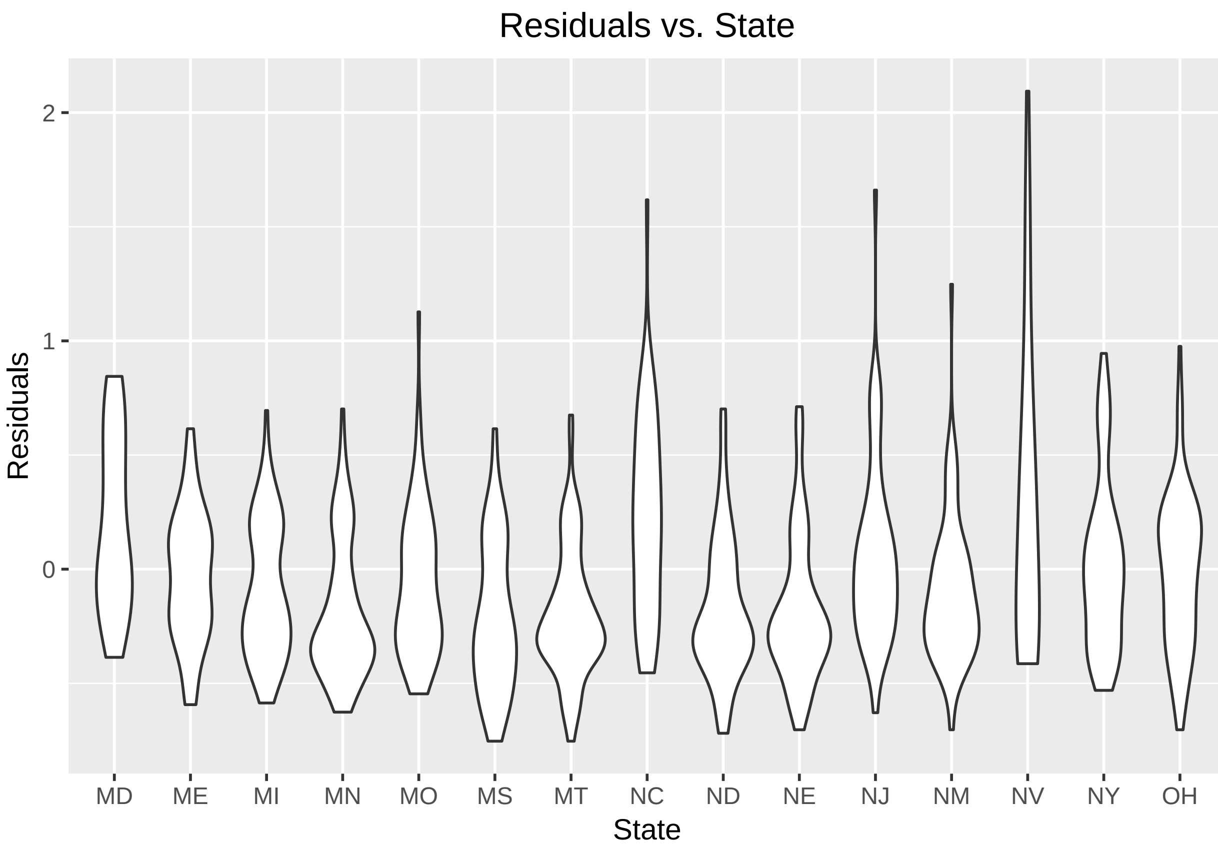
```
##    price_per_lb year.x .fitted .se.fit  .resid    .hat .sigma .cooksd
##           <dbl>  <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
##  1         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
##  2         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
##  3         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
##  4         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
##  5         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
##  6         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
##  7         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
##  8         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
##  9         0.72   1998   0.705  0.0346  0.0151 0.00564  0.461 3.05e-6
## 10         0.64   1998   0.705  0.0346 -0.0649 0.00564  0.461 5.68e-5
## # ... with 2,848 more rows, and 9 more variables: .std.resid <dbl>,
## #   state <chr>, num_colonies <dbl>, yield_per_colony_lb <dbl>,
## #   total_production <dbl>, stocks <dbl>, production_value <dbl>,
## #   year.y <dbl>, year_intervals <fct>
```

```
aug_df_price %>%
  filter(state %in% c("AL", "AR", "AZ", "CA", "CO", "FL", "GA","HI", "IA",
                      "ID", "IL", "IN", "KS", "KY", "LA")) %>%
  ggplot(aes(x = factor(state), y = .resid)) + geom_violin() +
  labs(x = "State", y = "Residuals") + ggtitle("Residuals vs. State") +
  theme(plot.title = element_text(hjust = 0.5))
```

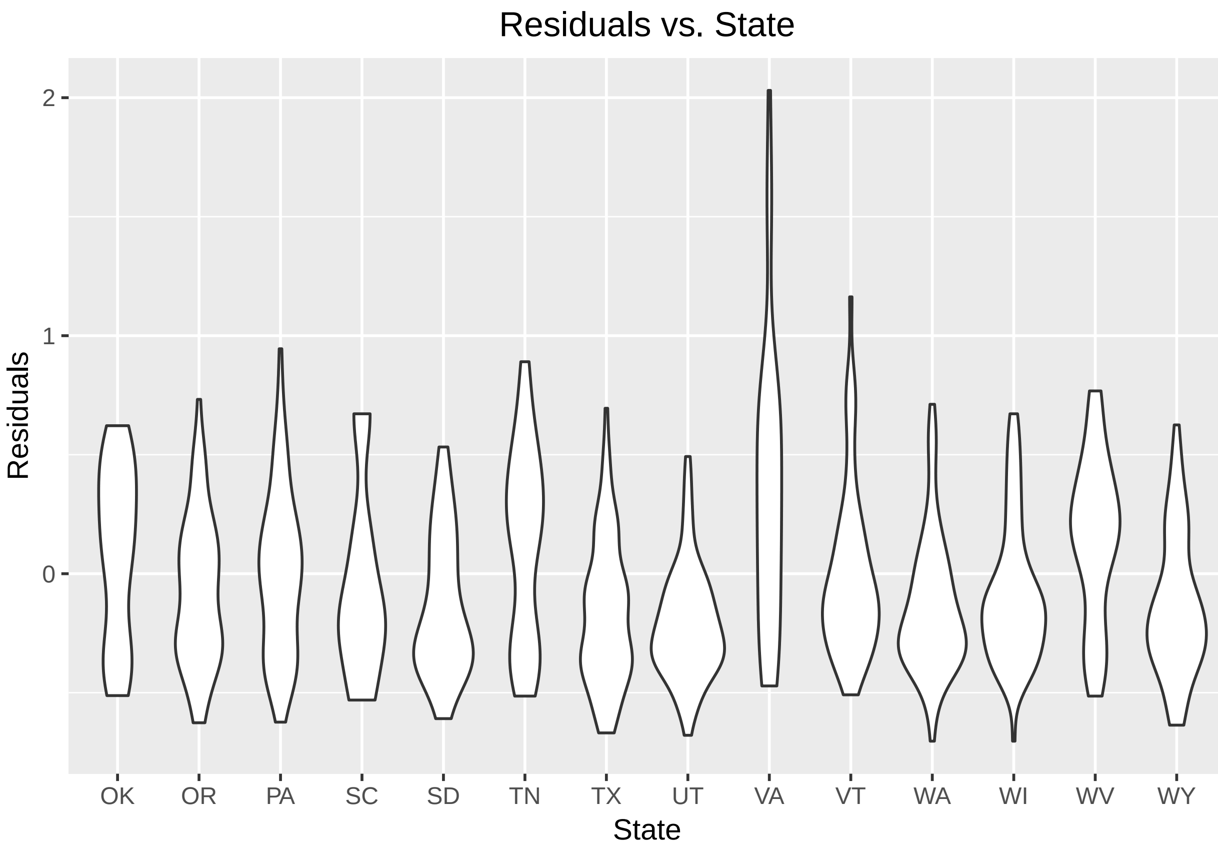



```
aug_df_price %>%
  filter(state %in% c("MD", "ME", "MI", "MN", "MO", "MS", "MT", "NC",
                      "ND", "NE", "NJ", "NM", "NV", "NY", "OH")) %>%
  ggplot(aes(x = factor(state), y = .resid)) + geom_violin() +
```

```
  labs(x = "State", y = "Residuals") + ggtitle("Residuals vs. State") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
aug_df_price %>%
  filter(state %in% c("OK", "OR", "PA", "SC", "SD", "TN", "TX", "UT",
                      "VA", "VT", "WA", "WI", "WV", "WY")) %>%
  ggplot(aes(x = factor(state), y = .resid)) + geom_violin() +
  labs(x = "State", y = "Residuals") + ggtitle("Residuals vs. State") +
  theme(plot.title = element_text(hjust = 0.5))
```



Here, we fit a linear regression model for price of honey per pound including a term for an interaction between state and year.

```
state_price_fit <- lm(price_per_lb~year+state, data=df)
state_price_fit
```

```
##
## Call:
## lm(formula = price_per_lb ~ year + state, data = df)
##
## Coefficients:
## (Intercept)         year      stateAR      stateAZ      stateCA
##  -213.82242      0.10735     -0.32667     -0.28400     -0.28533
##     stateCO      stateFL      stateGA      stateHI      stateIA
##    -0.18933     -0.31800     -0.23333      0.29133     -0.09800
##     stateID      stateIL      stateIN      stateKS      stateKY
##    -0.28067      0.72067      0.10733     -0.00600      0.66400
##     stateLA      stateMD      stateME      stateMI      stateMN
##    -0.36467      0.62608      0.01333     -0.18667     -0.26667
##     stateMO      stateMS      stateMT      stateNC      stateND
##    -0.04067     -0.36000     -0.26800      0.65533     -0.29333
##     stateNE      stateNJ      stateNM      stateNV      stateNY
##    -0.24600      0.09067     -0.10067      1.01513     -0.05733
##     stateOH      stateOK      stateOR      statePA      stateSC
##     0.17200      0.52108     -0.16600      0.10267      0.21339
##     stateSD      stateTN      stateTX      stateUT      stateVA
##    -0.27867      0.47533     -0.30400     -0.25200      0.90133
##     stateVT      stateWA      stateWI      stateWV      stateWY
```

```
##     0.16667     -0.17067     -0.14667      0.41600     -0.27067
```

```r
state_price_fit_stats <- state_price_fit %>%
  tidy()
state_price_fit_stats
```

```
## # A tibble: 45 x 5
##    term        estimate std.error statistic   p.value
##    <chr>          <dbl>     <dbl>     <dbl>     <dbl>
##  1 (Intercept) -214.      5.65      -37.9   5.61e-159
##  2 year           0.107   0.00282    38.1   3.48e-160
##  3 stateAR       -0.327   0.110      -2.98  2.99e-  3
##  4 stateAZ       -0.284   0.110      -2.59  9.78e-  3
##  5 stateCA       -0.285   0.110      -2.60  9.44e-  3
##  6 stateCO       -0.189   0.110      -1.73  8.45e-  2
##  7 stateFL       -0.318   0.110      -2.90  3.84e-  3
##  8 stateGA       -0.233   0.110      -2.13  3.36e-  2
##  9 stateHI        0.291   0.110       2.66  8.05e-  3
## 10 stateIA       -0.098   0.110      -0.894 3.71e-  1
## # ... with 35 more rows
```

The state_price_fit_stats allows us to find out how much price per pound of honey increases or decreases each year on average for each state. For instance, below is how you would calculate how much total honey production changes per year for AL, VA, and TN.

```r
cat("AL:", state_price_fit_stats$estimate[2], "\n")
```

```
## AL: 0.1073522
```

```r
cat("VA:", (state_price_fit_stats$estimate[2] + state_price_fit_stats$estimate[40]), "\n")
```

```
## VA: 1.008685
```

```r
cat("TN:", (state_price_fit_stats$estimate[2] + state_price_fit_stats$estimate[37]))
```

```
## TN: 0.5826855
```

Below, we perform a F-test that compares the Linear Regression with and without the state variable using the anova() function.

```r
comp_price <- anova(df_fit_price, state_price_fit)
comp_price
```

```
## Analysis of Variance Table
##
## Model 1: price_per_lb ~ year
## Model 2: price_per_lb ~ year + state
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1    624 132.126
## 2    581  52.308 43    79.819 20.618 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The interaction model (or Model #2) is significantly better than the year-only model because the p-value for Model #2 is ver low at 2.2e-16, meaning that adding state to the model significantly improved fit over Model #1

Conclusion

Through analyzing the data from the National Agricultural Statistics Service (NASS) of the U.S. Department of Agriculture (USDA), it is evident that in almost all 44 out of 50 states included in this dataset, the honey production has decreased from 1998-2012, subsequently causing the price of honey to increase.

According to the NASS and USDA, the honey Americans consume per year now mostly comes from foreign countries out of the US, instead of locally. The decline of honey production is largely due to the decline of the bee population.

Actions YOU can take to protect the bees and the honey!!! Educate yourself

Plant a pollinator garden
Stop using pesticides
Get involved in various projects: - https://www.greatsunflower.org/ - https://www.planetbee.org/zbw - https://honeybeenet.gsfc.nasa.gov/Sites.htm
Support beekeepers
Become a bee keeper :D