Honey I Shrunk the Bees!

Chris G Martin

May 15, 2016

Part 1 - Introduction

Bees: the most **effective** pollinator on the planet. In the United States, we seem to have a love/hate relationship with these facinating creatures; on the one hand we are well aware and appreciative of their role in nature and yet the number of bees is quickly dwindling without *us* giving *them* any significant help. Regardless of their global collapse in number, one of the many questions we will explore here is: **do we rely on them?**

By looking at a variety of data, we hope to find links between the production of honey and other agricultural statistics.

Why Honey

Collecting honey is an ancient tradition dating back over 8,000 years ago[1]. It's a product made by bees while foraging for nectar from flowers. While honey bees are the most common type of bee that produces the honey that humans extract for food, there are a number of other bees and wasps that create honey, and of course all bees and other insects pollinate. For the purposes of this project, we are going to ignore the details and make the assumption that honey is directly associated only to bees.

Most if not all beekeepers harvest honey from their bees as a source of income, but they also sell or rent hives to farms. As the number of bees across the world has steadily dropped, renting or selling hives has become a lucrative business. The sold or leased bees are taken to a farm and help to naturally pollinate farms and the lands around them, helping the farmer to grow crop, cultivate land, fertilize land, and develop the land more effectively. When the bees return to the hive after foraging, they produce honey from the nectar they collect, are returned the beekeeper or farmer, who in turn can harvest the honey and sell it for profit.

Honey production is valuable to the United States economy. Honey is used in making desserts, breads, barbecues, mustards, jellies and ointment treatments. Honey as amazing health benefits, too. It's loaded with antibacterial and antifungal properties and has been known to work well as a dextromethorphan, which is a common ingredient in cough medications to soothe cough. Additionally, honey can treat dandruff, is used in energy drinks, and can treat wounds and burns. It is a common belief that honey consumption can also help fight local allergies.

This study sets out to understand better understand some of the impacts on honey production in the United States by exploring various data fields across all 50 states, and futher inbetween, and attempts to predict honey production by creating links between these fields.

Part 2 - Data

Outlined in this section are the data processes: [1] Importing, [2] Data Cleaning, [3] Data Tidying, and [4] Data Exploration. Please see Appendix One: Data Merging in MySQL (http://rpubs.com/chrisgmartin/HISB_AppendixOne) for more details on the data importing and manipulation techinques; the MySQL queries can also be found there. All of the data has been saved in the GitHub Data folder (https://github.com/chrisgmartin/HoneyIShrunkTheBees/tree/master/Data).

Package Setup

Before we jump in, it's important to load any packages we might use. One option could be to load them as needed, however we may run into several problems with that as the *dplyr* package must be loaded after the *plyr* package (included with the *choroplethr* package) because doing otherwise will not cause issues when trying to implement the **group_by** and **summarise** features of the *dplyr* package. Also, the *pacman* package is used to load multiple packages because it tends to simplifies the code and organize it better visually (see example below).

```
#install.packages('pacman')
#library(pacman)
pacman::p_load(choroplethr, choroplethrMaps, RODBC, stringr, dplyr, tidyr, inference, plotly)
#library(choroplethr)
#library(choroplethrMaps)
#library(RODBC)
#library(stringr)
#library(dplyr)
#library(tidyr)
```

Another data set we will need comes from the City University of New York's Masters of Data Analytics Program (CUNY MSDA Program) course IS 606 - Statistics and Probability for Data Analytics. These custom files assist in the linear regression as it includes two very handy custom functions: **ls** and **inference** (we will show this in the Linear Regression).

```
load(url("https://github.com/chrisgmartin/HoneyIShrunkTheBees/raw/master/Additional/nc.RData"))
```

Data Import

The most important part of this analysis is obviously the data. Fortunately there is a plethora of open source data available.

After following some detailed Stack Overflow (https://stackoverflow.com/questions/10292326/how-to-connect-r-with-mysql-or-how-to-install-rmysql-package) tips, I was able to connect to a local MySQL server that I created called 'HISP' (Honey, I Shrunk the Production – a previous title for this project). Please see the HIBS database sources in Appendix 1 and outlined below.

```
con <- odbcConnect("hisp")
sqlQuery(con, "use hisp")
```

```
## [1] "No Data"
```

National Agriculture Statistics Service

The first set of data came from the National Agriculture Statistics Service website (https://quickstats.nass.usda.gov/) (NASS), which houses a large amount of data related to United States Agriculture. The website stores census and survey data on a number of subjects (the main categories are animals, products, crops, demographics, economics, and environment) across a number of geographic levels (national, state, county, and zip code) for a number of years. The data can be access by the website UI allowing the user to select each subject, geographic level, and year which can then be exported into a .CSV file, or the entire database can be downloaded as a text file after extracting the database .GZ. For our data it was simple enough to query the database using the website rather than have to deal with the large and cumbersome dataset. The queries were then saved into seperate .CSV files, imported into MySQL, and prepared for R (https://github.com/chrisgmartin/HoneyIShrunkTheBees/blob/master/HoneyIShrunkTheBees.sql):

```
#Load the data
honey_county <- as.data.frame(sqlQuery(con, "select * from honey_county", stringsAsFactors=FALSE))
honey_state <- as.data.frame(sqlQuery(con, "select * from honey_state", stringsAsFactors=FALSE))
#If MySQL isn't set up for you, the tables can be imported via GitHub:
#honey_county <- read.csv('https://raw.githubusercontent.com/chrisgmartin/HoneyIShrunkTheBees/master/Data/honey_county.csv', heade
r = TRUE)
#honey_state <- read.csv('https://raw.githubusercontent.com/chrisgmartin/HoneyIShrunkTheBees/master/Data/honey_state.csv', header
= TRUE)

#limiting the output to 10 columns for space considerations
kable(head(honey_county[,1:10], 3))
```

Program	Year	Period	WeekEnding	GeoLevel	State	StateANSI	AGDistrict	AGDistrictCode	County
CENSUS	2012	YEAR	NA	COUNTY	AL	1	BLACK BELT	40	AUTAUGA
CENSUS	2012	YEAR	NA	COUNTY	AL	1	BLACK BELT	40	AUTAUGA
CENSUS	2012	YEAR	NA	COUNTY	AL	1	BLACK BELT	40	AUTAUGA

```
kable(head(honey_state[,1:10], 3))
```

Program	Year	Period	WeekEnding	GeoLevel	State	StateANSI	AGDistrict	AGDistrictCode	County
CENSUS	2014	YEAR	NA	STATE	AL	1	NA	NA	NA
CENSUS	2014	YEAR	NA	STATE	AL	1	NA	NA	NA
CENSUS	2014	YEAR	NA	STATE	AL	1	NA	NA	NA

American National Standards Institute (ANSI) Codes

The NASS data sets include a column for ANSI codes at both the state and county level. These ANSI codes can be extremely useful in analyzing data, especially when it comes to plotting it on a map. ANSI codes (American National Standards Institute (https://www.census.gov/geo/reference/ansi.html)) are standardized codes used to ensure uniform identification of geography between various agencies. Since September 2008, ANSI has also replaced the use of Federial Information Processessing Standard (https://en.wikipedia.org/wiki/FIPS_county_code) (FIPS) codes and National Institute of Standards and Technology codes (NIST).

One item of note is that the states in our NASS data sets are full text state names (for example, California instead of CA, or New York instead of NY). There is a seperate SQL script hosted at statetable.com (http://www.statetable.com) that was used to convert the full length text in the NASS data into its abbreviated form to simplify our useage. The script is hosted on my GitHub page (https://github.com/chrisgmartin/HoneyIShrunkTheBees/blob/master/Data/state_table.sql) for reference.

It might also come in handy to save the the FIPS County codes reference table (https://www.census.gov/geo/reference/codes/cou.html), which has been saved and stored locally using MySQL, in case we need to reference it:

```
#Load FIPS
FIPS <- (sqlQuery(con, "select * from FIPS", stringsAsFactors=FALSE))

#For those without a MySQL link established:
#FIPS <- read.csv('https://raw.githubusercontent.com/chrisgmartin/HoneyIShrunkTheBees/master/Data/FIPS.csv', header = TRUE)
```

Closing the MySQL Connection

For safe measures, we should always close out our MySQL connection to maintain data integrity.

```
odbcClose(con)
```

Data Cleaning

With the data loaded into R we'll need to clean it the best we can. First of all, there are several columns that are unnecessary for our purposes. Essentially, the columns we need (or might need) are: Program (column 1), Year (2), GeoLevel (5), State (6), StateANSI (7), County (10), CountyANSI (11), Commodity (16), DataItem (17), and Value (20). The first 10 items are descriptive, while the final column contains our variable. At the state level, we'll also need some addidtional columns Domain (18) and DomainCategory (19) that we'll group later to reduce.

Let's pull in only the columns we need:

```
honey_county2 <- honey_county[c(1,2,5,6,7,10,11,16,17,20)]
kable(head(honey_county2, 3))
```

Program	Year	GeoLevel	State	StateANSI	County	CountyANSI	Commodity	DataItem	Value
CENSUS	2012	COUNTY	AL	1	AUTAUGA	1	CHEMICAL TOTALS	CHEMICAL TOTALS - EXPENSE, MEASURED IN $	1250000
CENSUS	2012	COUNTY	AL	1	AUTAUGA	1	CHEMICAL TOTALS	CHEMICAL TOTALS - OPERATIONS WITH EXPENSE	192
CENSUS	2012	COUNTY	AL	1	AUTAUGA	1	FERTILIZER TOTALS	FERTILIZER TOTALS, INCL LIME & SOIL CONDITIONERS - EXPENSE, MEASURED IN $	2237000

```
honey_state2 <- honey_state[c(1,2,5,6,7,10,11,16,17,18,19,20)]
kable(head(honey_state2, 3))
```

Program	Year	GeoLevel	State	StateANSI	County	CountyANSI	Commodity	DataItem	Domain	DomainCategory	Value
CENSUS	2014	STATE	AL	1	NA	NA	HORTICULTURE TOTALS	HORTICULTURE TOTALS - OPERATIONS WITH SALES	SALES OF HORTICULTURE	SALES OF HORTICULTURE: (1,000,000 TO 2,499,999 $)	3.332205
CENSUS	2014	STATE	AL	1	NA	NA	HORTICULTURE TOTALS	HORTICULTURE TOTALS - OPERATIONS WITH SALES	SALES OF HORTICULTURE	SALES OF HORTICULTURE: (10,000 TO 19,999 $)	3.496508
CENSUS	2014	STATE	AL	1	NA	NA	HORTICULTURE TOTALS	HORTICULTURE TOTALS - OPERATIONS WITH SALES	SALES OF HORTICULTURE	SALES OF HORTICULTURE: (100,000 TO 249,999 $)	3.784190

Futher exploration of the data found several characters in the *Value* column, which will have to be fix but first understand why the characters are there: the NASS website includes a glossary (https://quickstats.nass.usda.gov/src/glossary.pdf) which describes that the **(D)** and **(Z)** characters found in the *Values* column mean "Withheald to avoid discolsing data for individual operations" (D) and "Less than half of the rounding unit" (Z). It seems that (Z) is

essentially a *(NULL)* value and can be simply replaced with a 0 but (D) is tricky since it could be 0, it could be extremely large, or it could be somewhere in between. There are a number of things we could do to fix (D) values: replace them with 0 (positively skewing our results), use a total average (likey skewing our results in an unforseen way), use a weighted average based on a certain metric such as *Year* (likely skewing our data), use an average based on another metric such as *State* (also skewing our data), or make an educated guess assumption (which will most likely skew our data). From my gut instinct I feel that using an average based on a metric such as *State* would be the best way to do this as it is less prone to skewing our data for any extreme data points (outliers) in a given year. Thankfully, MySQL again made doing that very simple (see the first appendix)–though, I've included the code below for some of the R work as a reference:

```
#Replace (Z)'s and (D)'s with 0 value
#honey_county2$Value[honey_county2$Value == " (Z)"] <- as.integer(0)
#honey_state2$Value[honey_state2$Value == " (Z)"] <- as.integer(0)
#honey_county2$Value[honey_county2$Value == " (D)"] <- as.integer(0)
#honey_state2$Value[honey_state2$Value == " (D)"] <- as.integer(0)

#Remove comma's from digits
#honey_county2$Value <- str_replace_all(honey_county2$Value, "[:punct:]", "")
#honey_state2$Value <- str_replace_all(honey_state2$Value, "[:punct:]", "")

#check to see if it worked correctly
#honey_county2$Value[honey_county2$Value == " (Z)"]
#honey_county2$Value[honey_county2$Value == " (D)"]
#honey_state2$Value[honey_county2$Value == " (Z)"]
#honey_state2$Value[honey_county2$Value == " (D)"]
```

Important note: R currently cannot properly store integer values greater than 2,147,483,647 (I was getting an error which pointed me to StackOverflow (http://stackoverflow.com/questions/14589354/struggling-with-integers-maximum-integer-size)). As a result, I have fixed the "*CROP TOTALS - SALES, MEASURED IN $*" rows in the county file (which contained several values greater than 2,147,483,647) to be the log form, and fixed all values in the entire state file to be log form. The original idea was to change all items above this limit to equal the limit but it would absolutely lead to errors in our data, and in a real-life situation this would never be a suggestion to recommened, ever as it would introduce a number of complications and throw off results. **In real-life, we are always affected by the assumptions we make, so limiting the potential for data bias can go a long way.** The solution to set all values in the State dataset as its log value also has the benefit of 'normalizing' the data.

```
#Change Values into Int
honey_county2$Value <- as.integer(honey_county2$Value)
honey_state2$Value <- as.integer(honey_state2$Value)

#Remove NA's to zero
#NA's are from null values from the MySQL Query
#Where (D) values did not have an average to calculate
honey_county2$Value[is.na(honey_county2$Value)] <- as.integer(0)
honey_state2$Value[is.na(honey_state2$Value)] <- as.integer(0)
```

But we're not done yet! ANSI codes need to be in a specific order for mapping. For counties, a 3 digit ANSI code length with leading zeros is required. (NULL) values are present, but unlike before where we simply used 0's for (NULL) values, removing these rows would lead to a loss of essential information from a very trivial column. Rather, we leave them as NULL because the CountryANSI column is only used in plotting values on a map; the values still exist and remain in the data, but since the map won't be able to decide where to place them they remain invisible. For the counties dataset, mapping using the ChoroplethR package requires combined 2 digit state ANSI codes AND 3 digit county codes. The State requirement is simiply the uncapitalized state name which was fixed in our MySQL code.

```
#honey_county2$StateANSI2 <- sprintf("%02d", honey_county2$StateANSI)
#honey_state2$StateANSI2 <- sprintf("%02d", honey_state2$StateANSI)

#convert all CountyANSI codes to three digits
honey_county2$CountyANSI <- sprintf("%03d", honey_county2$CountyANSI)

#merging the StateANSI code and the CountyANSI codes
honey_county2$CountyANSI <- as.numeric(paste0(honey_county2$StateANSI, honey_county2$CountyANSI))
```

```
## Warning: NAs introduced by coercion
```

Data Tidying

With the data loaded and more-or-less cleaned, it should be tidy'd a bit for viewing purposes. Here we can arrange the data by Data Item and State name:

```
honey_county2 <- honey_county2 %>%
  arrange(., DataItem) %>%
  arrange(., State)

honey_state2 <- honey_state2 %>%
  arrange(., DataItem) %>%
  arrange(., State)
```

Another arrangement we will need is to re-format our table. Our table currently exists with each data item, each year, each per state, and each county having it's own row and value but we actually need each data item to have it's own column so that each state (for example Alaska) will have one row per year and columns with Values running to the right of it. Our header output is limited to 10 columns, as it can be very long to display fully in our browser.

```
#install.packages('tidyr')

#transposes the county table
honey_county3 <- honey_county2[, c(2,4,5,6,7,9,10)] %>%
  spread(., DataItem, Value)
kable(head(honey_county3[,1:10], 1))
```

Year	State	StateANSI	County	CountyANSI	CHEMICAL TOTALS - EXPENSE, MEASURED IN $	CHEMICAL TOTALS - OPERATIONS WITH EXPENSE	CROP TOTALS - OPERATIONS WITH SALES	CROP TOTALS - SALES, MEASURED IN $	CROP TOTALS, ORGANIC - OPERATIONS WITH SALES
2012	AK	2	KENAI PENINSULA	NA	17000	36	101	14	NA

```
#grouping needs to be applied for the state data
#at this level because there are the additional columns
#for Domain and DomainCategory
honey_state2_tmp <- honey_state2 %>%
  group_by(Year, State, DataItem) %>%
  summarise(Value = sum(Value))

honey_state3 <- honey_state2_tmp %>%
  spread(., DataItem, Value)
kable(head(honey_state3[,1:10], 1))
```

Year	State	CHEMICAL TOTALS - EXPENSE, MEASURED IN $	CHEMICAL TOTALS - EXPENSE, MEASURED IN PCT OF OPERATING EXPENSES	CHEMICAL TOTALS - OPERATIONS WITH EXPENSE	CHEMICAL TOTALS, (FOR HORTICULTURE) - EXPENSE, MEASURED IN $	CHEMICAL TOTALS, (FOR HORTICULTURE) - OPERATIONS WITH EXPENSE	CHEMICAL TOTALS, (FOR ORGANIC) - EXPENSE, MEASURED IN $	CHEMICAL TOTALS, (FOR ORGANIC) - OPERATIONS WITH EXPENSE	CROP TOTALS - OPERATIONS WITH SALES
1987	AL	NA	NA	NA	NA	NA	NA	NA	NA

```
#export tables for use in presentation
#write.csv(honey_state2, file = 'D:/GitHub/HoneyIShrunkTheBees/Appendix/honey_state2.csv')
#write.csv(honey_county2, file = 'D:/GitHub/HoneyIShrunkTheBees/Appendix/honey_county2.csv')
```

Now going back to our Data Cleaning section, what do we do with pesky (NULL) values? We cleaned them up before, back when each data item had its own row, but now that the data items are rolled-up and transposed into their State/Year combinations we have a large number of rows where our given metric will be analysed. Likely to be the most practical DataItem for our topic, **HONEY - PRODUCTION, MEASURED IN LB** for counties and **HONEY - PRODUCTION, MEASURED IN LB / COLONY** for state level data has a number of (NULL) values. Let's remove those data items.

```
#example with NA columns
head(honey_county3$`HONEY - PRODUCTION, MEASURED IN LB`)
```

```
## [1]   982 26666    NA    NA  4975   590
```

```
honey_county3 <- honey_county3[!(is.na(honey_county3$`HONEY - PRODUCTION, MEASURED IN LB`)),]
head(honey_county3$`HONEY - PRODUCTION, MEASURED IN LB`)
```

```
## [1]   982 26666  4975   590  3306  9072
```

```
honey_state3 <- honey_state3[!(is.na(honey_state3$`HONEY - PRODUCTION, MEASURED IN LB / COLONY`)),]



#to export the table
write.csv(honey_state3, file = 'D:/GitHub/HoneyIShrunkTheBees/Data/honey_state3.csv')
write.csv(honey_county3, file = 'D:/GitHub/HoneyIShrunkTheBees/Data/honey_county3.csv')
```

Data Exploration

Since we have the data set we can explore what's in it before we decide where and how to clean it. Let's start with the column for data descriptions: Data Item

```
length(unique(honey_county2$DataItem))
```

```
## [1] 21
```

```
nrow(honey_county2)
```

```
## [1] 129054
```

```
length(unique(honey_state2$DataItem))
```

```
## [1] 108
```

```
nrow(honey_state2)
```

```
## [1] 95506
```

There are a lot more data items (87 more to be exact) in the state set than in the county set, yet the county set has more variables/row (33,548).

```
unique(honey_county$DataItem)
```

```
##  [1] "CHEMICAL TOTALS - EXPENSE, MEASURED IN $"
##  [2] "CHEMICAL TOTALS - OPERATIONS WITH EXPENSE"
##  [3] "FERTILIZER TOTALS, INCL LIME & SOIL CONDITIONERS - EXPENSE, MEASURED IN $"
##  [4] "FERTILIZER TOTALS, INCL LIME & SOIL CONDITIONERS - OPERATIONS WITH EXPENSE"
##  [5] "CROP TOTALS - OPERATIONS WITH SALES"
##  [6] "CROP TOTALS - SALES, MEASURED IN $"
##  [7] "CROP TOTALS, ORGANIC - OPERATIONS WITH SALES"
##  [8] "CROP TOTALS, ORGANIC - SALES, MEASURED IN $"
##  [9] "HONEY - OPERATIONS WITH PRODUCTION"
## [10] "HONEY - OPERATIONS WITH SALES"
## [11] "HONEY - PRODUCTION, MEASURED IN LB"
## [12] "HONEY - SALES, MEASURED IN $"
## [13] "HONEY, BEE COLONIES - OPERATIONS WITH SALES"
## [14] "HONEY, BEE COLONIES - SALES, MEASURED IN COLONIES"
## [15] "HORTICULTURE TOTALS, (EXCL CUT TREES & VEGETABLE SEEDS & TRANSPLANTS) - OPERATIONS WITH SALES"
## [16] "HORTICULTURE TOTALS, (EXCL CUT TREES & VEGETABLE SEEDS & TRANSPLANTS) - SALES, MEASURED IN $"
## [17] "HORTICULTURE TOTALS, (EXCL CUT TREES) - OPERATIONS WITH AREA IN PRODUCTION"
## [18] "HORTICULTURE TOTALS, (EXCL CUT TREES), IN THE OPEN - ACRES IN PRODUCTION"
## [19] "HORTICULTURE TOTALS, (EXCL CUT TREES), IN THE OPEN, IRRIGATED - ACRES IN PRODUCTION"
## [20] "HORTICULTURE TOTALS, (EXCL CUT TREES), IN THE OPEN, IRRIGATED - OPERATIONS WITH AREA IN PRODUCTION"
## [21] "HORTICULTURE TOTALS, (EXCL CUT TREES), UNDER PROTECTION - SQ FT IN PRODUCTION"
```

Data Check: Exploring Bee Colonies

At this stage, it would be extremely helpful to check that our data is, in fact, useful. The reason we do this here is that we need to know whether we can move on from the *Data* stage and into the *Exploration* phase which will lead us into our *Inference* and *Conclusion* stages. Once this box is checked, we can move on.

Let's now take a look at how the number of honey bee colonies are disbursed in the states (remember, all values in the States data set are log values):

```
#calculate or plot number of bee colonies over the years
kable(honey_county2[which(honey_county2$DataItem == 'HONEY, BEE COLONIES - OPERATIONS WITH SALES'), c(2,9,10)] %>%
  group_by(Year) %>%
  summarise(value = sum(Value)))
```

Year	value
2002	937
2007	1046

```
kable(honey_state2[which(honey_state2$DataItem == 'HONEY, BEE COLONIES - OPERATIONS WITH SALES'), c(2,9,12)] %>%
  group_by(Year) %>%
  summarise(value = sum(exp(Value))))
```

Year	value
1997	670.3869
2002	518.7462
2007	610.5204

Interestingly, the number of bee colonies (those that are actually operations with sales, not wild bees) didn't change a whole lot between 2002 and 2007 at the county level; it's more difficult to tell at the state level though as they have dropped significantly from 1997 and 2002 but grew between 2002 and 2007. Perhaps we are seeing growth in the number of bee colonies contrary to popular belief, or perhaps we are being misled by our data. We will explore a similar aspect our linear regression analysis.

Data Check: Looking into Honey Production

In the Honey Production data, there are a number of variables that we can choose to analyse such as the number of Operations with Sales, number of Operations with Production, Sales in $'s, and Production in lb's. The most important for this project I believe is Production in lb's since the number of operations can vary (due to business consolidation, expansion, etc.) and sales is highly influenced by economic factors not just how the bee's "performed" throughout the year.

```
kable(honey_county2[which(honey_county2$DataItem == 'HONEY - PRODUCTION, MEASURED IN LB'), c(2,9,10)] %>%
  group_by(Year) %>%
  summarise(value = sum(Value)))
```

Year	value
2002	168336363
2007	182863385
2012	170770048

```
kable(head(honey_state2[which(honey_state2$DataItem == 'HONEY - PRODUCTION, MEASURED IN LB'), c(2,9,12)] %>%
  group_by(Year) %>%
  summarise(value = sum(exp(Value)))))
```

Year	value
1987	150459561
1988	139057173
1989	97846617
1990	110702799
1991	126242150
1992	132334888

```
kable(tail(honey_state2[which(honey_state2$DataItem == 'HONEY - PRODUCTION, MEASURED IN LB'), c(2,9,12)] %>%
  group_by(Year) %>%
  summarise(value = sum(exp(Value)))))
```

Year	value
2010	110126406
2011	93430288
2012	182914053
2013	96463390
2014	123170176
2015	93294512

Again, the production of honey seems to follow the size of the bee colonies with its ups and downs through the years. Also very facinating, we can see how the honey production per colony has changed at the state level:

```
kable(head(honey_state2[which(honey_state2$DataItem == 'HONEY - PRODUCTION, MEASURED IN LB / COLONY'), c(2,9,12)] %>%
  group_by(Year) %>%
  summarise(value = sum(exp(Value)))))
```

Year	value

1987	1859.100
1988	1859.100
1989	1494.727
1990	1764.682
1991	1697.639
1992	1835.690

```
kable(tail(honey_state2[which(honey_state2$DataItem == 'HONEY - PRODUCTION, MEASURED IN LB / COLONY'), c(2,9,12)] %>%
  group_by(Year) %>%
  summarise(value = sum(exp(Value)))))
```

Year	value
2010	1582.785
2011	1444.734
2012	1479.247
2013	1286.598
2014	1513.759
2015	1341.196

Extremely interesting. We're seeing a major drop in production per colony throughout the years.

Initial Observations and Comments

On the resources I listed and used thus far (mainly NASS), they are incredibly excellent and economically valueable resources for researchers and businesses alike. The data seems to be well maintained, fairly simple to extract, and reliably updated. I'd like to extend my gratitude to those organizations for their work as I'm certain it will help us all now and in the future.

Of the variables in the dataset there are some I expect to have high positive correlation to Honey Production, while some I expect to have a strong negative correlation with Honey Production. Of those, horticulture data seems like it would have a strong impact as bees rely on pollen to survive. Chemical and Fertilizer use, however, I expect to have a strong negative correlation. My initial opinion is that these chemicals and fertilizers make it difficult for bees to sustain themselves in the "nature" we force them into. Crops, I believe, can be a "control" group in that I expect them to be fairly neutral to our Honey Produciton as they may or may not be directly related to our pollinator end product (more bees / more crops? or less bees / less crops?).

Another observation is the limited data in some categories. For example, at the state level for Honey Production in lb's, we have annual data from 1987 to 2015; however, in the county level we only have three years of data: 2002, 2007, and 2012. This limits our ability to really dive into that. It also hinders our ability to see a meaningful trend. There are many more useful data points found online which require some additional work to combine and analyse in tandem with other variables such as Crop Sales. For the sake of convience, the NASS dataset was a great start to my research.

By the end of this analysis: I hope that my personal expectations will not bias my end results, in fact I hope I am pleasantly surprised by my results in the end.

Part 3 - Exploratory data analysis

Here we will try to explore some of our data and see if we can find any interesting details that would be helpful to know.

Map

Perfect for our needs, the ChoroplethR package (https://cran.r-project.org/web/packages/choroplethr/index.html) will help us visualize our data on a map of the United States (it can do others countries too, but we only have data on the United States). The package can let us visualize at both the state and county level, and has a some extra reference tables which could be useful such as the **state regions** table matching our (now cleaned up version) data.

```
data(state.regions)
kable(head(state.regions))
```

region	abb	fips.numeric	fips.character
alaska	AK	2	02
alabama	AL	1	01
arkansas	AR	5	05
arizona	AZ	4	04

california	CA	6	06
colorado	CO	8	08

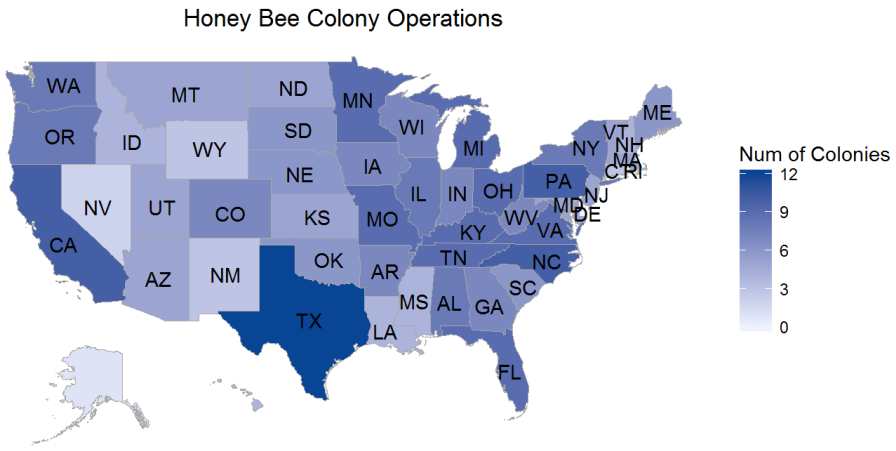
Let's take a look at how the number of honey bee colony operations with sales are distributed across the United States:

```
HS_BeeCol <- honey_state2[which(honey_state2$DataItem == 'HONEY, BEE COLONIES - OPERATIONS WITH SALES'), c(4,12)]

HS_HonPlot <- HS_BeeCol %>%
  group_by(State) %>%
  summarise(value = sum(Value))

HS_HonPlot <- data.frame(region= state.regions$region, value= HS_HonPlot[match(state.regions$abb, HS_HonPlot$State), 2])


state_choropleth(HS_HonPlot, title = "Honey Bee Colony Operations", legend = "Num of Colonies", num_colors = 1)
```



Texas seems to have the highest number of bee colony operations, with California, Pennsylvania, and North Carolina following behind.

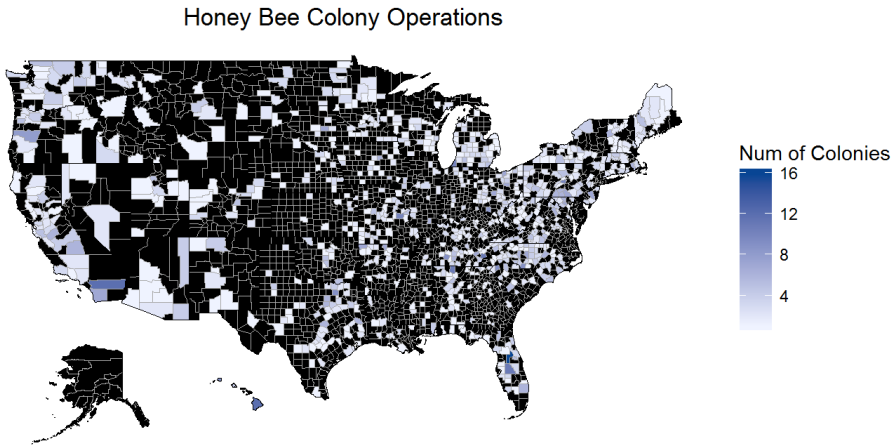
As far as counties go, the map is a bit more segregated than expected with some counties having larger quantities of bee colonies than others (black counties have no colonies). For example, most of the California bee colony operations are in Southern California, around San Diego. Nevada and the central United States seem to lack any honey bee operations, and Central Florida has a massive quantity.

```
HC_BeeCol <- honey_county2[which(honey_county2$DataItem == 'HONEY, BEE COLONIES - OPERATIONS WITH SALES'), c(7,10)]

colnames(HC_BeeCol) <- c('region', 'value')

HC_HonPlot <- HC_BeeCol %>%
  group_by(region) %>%
  summarize(value = sum(value))

county_choropleth(HC_HonPlot, title = "Honey Bee Colony Operations", legend = "Num of Colonies", num_colors = 1)
```



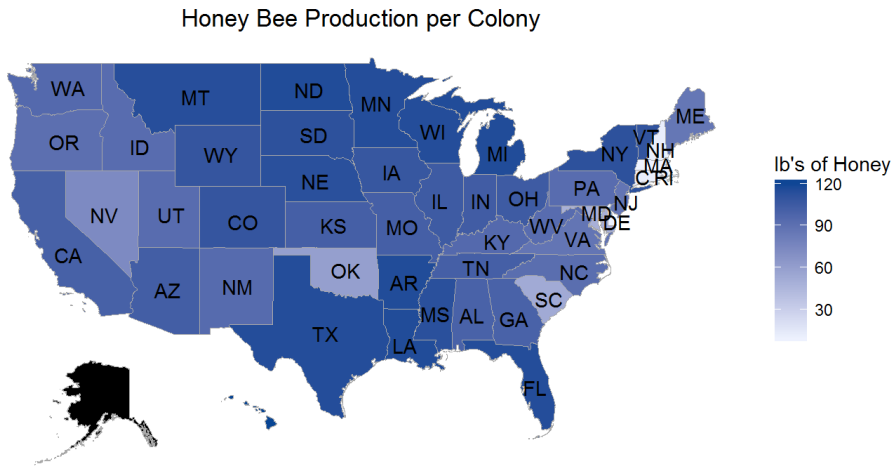
We can also look to see how much Honey Bee Production per Colony varies between states:

```
HS_HoneyProdPer <- honey_state2[which(honey_state2$DataItem == 'HONEY - PRODUCTION, MEASURED IN LB / COLONY'), c(4,12)]

HS_HonProdPlot <- HS_HoneyProdPer %>%
  group_by(State) %>%
  summarize(value = sum(Value))

HS_HonProdPlot <- data.frame(region= state.regions$region, value= HS_HonProdPlot[match(state.regions$abb, HS_HonProdPlot$State),
2])

state_choropleth(HS_HonProdPlot, title = "Honey Bee Production per Colony", legend = "lb's of Honey", num_colors = 1)
```



It seems like there is a fairly even distribution in this, with the exception of New Hampshire, Massachusetts, and Connecticut (whose bees apparently are less productive than others). They are in start contrast with states like Arkansas, FLorida, and Texas (whose bees apparently believe in the Texas motto: “Bigger Is Better”).

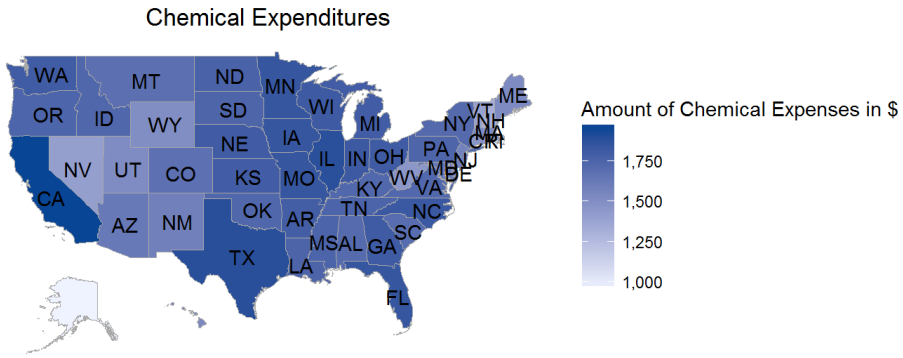
Let’s also take a look at chemical expenditures:

```
HS_ChemExp <- honey_state2[which(honey_state2$DataItem == 'CHEMICAL TOTALS - EXPENSE, MEASURED IN $'), c(4,12)]

HS_ChemPlot <- HS_ChemExp %>%
  group_by(State) %>%
  summarize(value = sum(Value))

HS_ChemPlot <- data.frame(region= state.regions$region, value= HS_ChemPlot[match(state.regions$abb, HS_ChemPlot$State), 2])

state_choropleth(HS_ChemPlot, title = "Chemical Expenditures", legend = "Amount of Chemical Expenses in $", num_colors = 1)
```



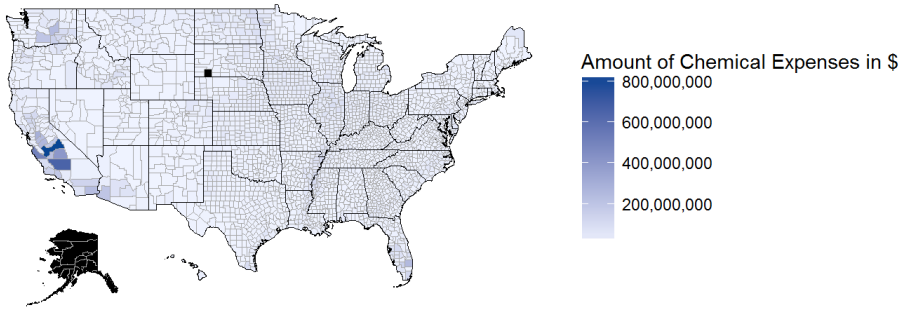
In California, expenditures on chemicals are surprisingly high. Details on what these chemicals are, unfortunately, are not here.

```
HC_ChemExp <- honey_county2[which(honey_county2$DataItem == 'CHEMICAL TOTALS - EXPENSE, MEASURED IN $'), c(7,10)]

colnames(HC_ChemExp) <- c('region', 'value')

HC_ChemPlot <- HC_ChemExp %>%
  group_by(region) %>%
  summarize(value = sum(value))

county_choropleth(HC_ChemPlot, title = "Chemical Expenditures", legend = "Amount of Chemical Expenses in $", num_colors = 1)
```



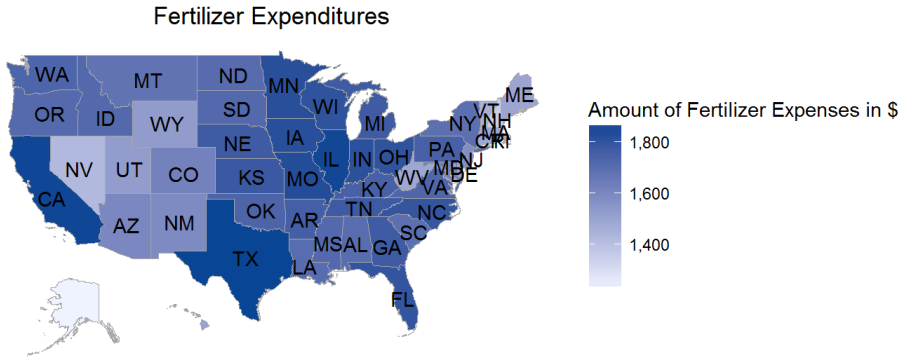
At the county level, almost all of California's expenditures are made right in the farming belt of Central California which is no surprise. What is surprising though is the concentration of expenditures compared to the rest of the United States. Some further research could be done as expenditures may actually be the location where the expense was booked from and not where the chemicals were distributed or used.

```
HS_FertExp <- honey_state2[which(honey_state2$DataItem == 'FERTILIZER TOTALS, INCL LIME & SOIL CONDITIONERS - EXPENSE, MEASURED IN
$'), c(4,12)]

HS_FertPlot <- HS_FertExp %>%
  group_by(State) %>%
  summarize(value = sum(Value))

HS_FertPlot <- data.frame(region= state.regions$region, value= HS_FertPlot[match(state.regions$abb, HS_FertPlot$State), 2])

state_choropleth(HS_FertPlot, title = "Fertilizer Expenditures", legend = "Amount of Fertilizer Expenses in $", num_colors = 1)
```



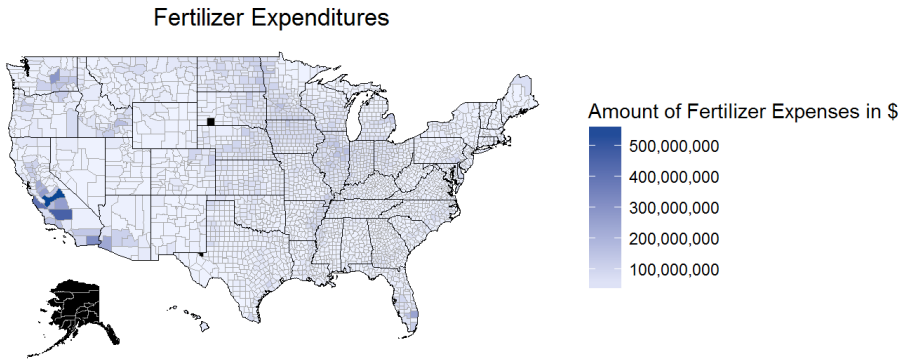
Fertilizer expenditures (including Lime and Soil Conditioners) looks similar to the chemicals map, but Texas is comparitively expending more on fertilizers than they were on chemicals. At the county level, same story as Chemicals with such a large expense taking place in the Central California region.

```
HC_FertExp <- honey_county2[which(honey_county2$DataItem == 'FERTILIZER TOTALS, INCL LIME & SOIL CONDITIONERS - EXPENSE, MEASURED
IN $'), c(7,10)]

colnames(HC_FertExp) <- c('region', 'value')

HC_FertPlot <- HC_FertExp %>%
  group_by(region) %>%
  summarize(value = sum(value))

county_choropleth(HC_FertPlot, title = "Fertilizer Expenditures", legend = "Amount of Fertilizer Expenses in $", num_colors = 1)
```



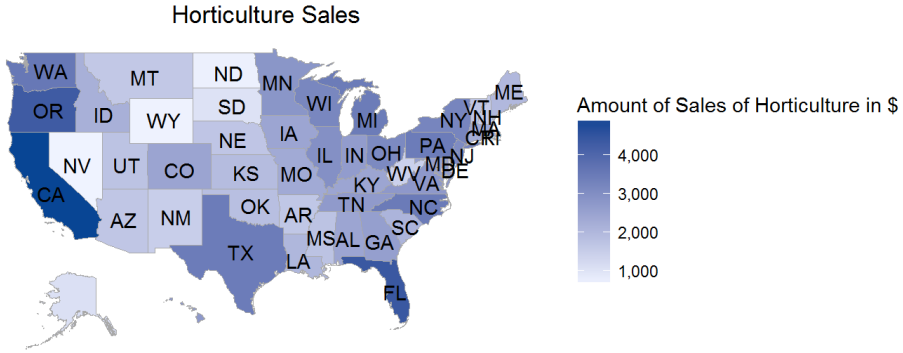
Horticulture sales are highly concentrated on the West Coast and Florida:

```
HS_HortSales <- honey_state2[which(honey_state2$DataItem == 'HORTICULTURE TOTALS - SALES, MEASURED IN $'), c(4,12)]

HS_HortPlot <- HS_HortSales %>%
  group_by(State) %>%
  summarize(value = sum(Value))

HS_HortPlot <- data.frame(region= state.regions$region, value= HS_HortPlot[match(state.regions$abb, HS_HortPlot$State), 2])

state_choropleth(HS_HortPlot, title = "Horticulture Sales", legend = "Amount of Sales of Horticulture in $", num_colors = 1)
```



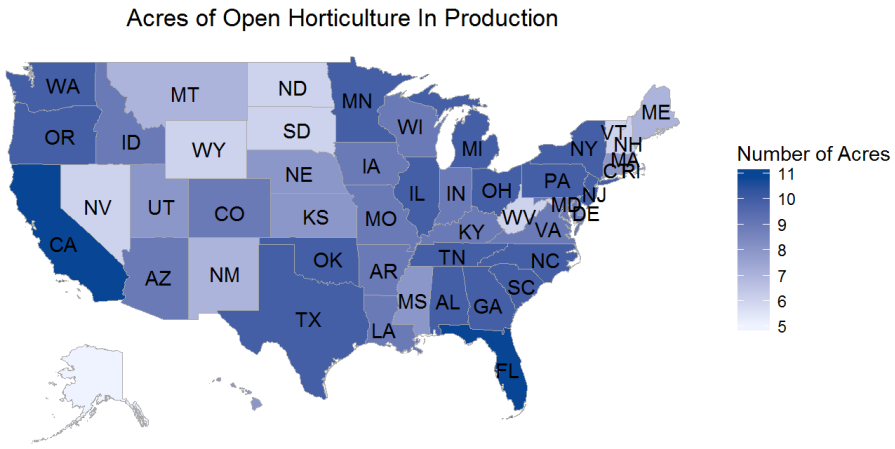
But for our purposes what could be important with horticulture is the number of acres in production, rather than sales, and it seems there is a more equal distribution of horticulture acres in production across the United States, with Calfornia and Florida having the lions share. I would venture to guess there is some cross over between horticulture production and chemical/fertilizer use, but that's another topic from another day.

```
HS_HortProd <- honey_state2[which(honey_state2$DataItem == 'HORTICULTURE TOTALS, (EXCL CUT TREES), IN THE OPEN - ACRES IN PRODUCTI
ON'), c(4,12)]

HS_HortPlot2 <- HS_HortProd %>%
  group_by(State) %>%
  summarize(value = sum(Value))

HS_HortPlot2 <- data.frame(region= state.regions$region, value= HS_HortPlot2[match(state.regions$abb, HS_HortPlot2$State), 2])

state_choropleth(HS_HortPlot2, title = "Acres of Open Horticulture In Production", legend = "Number of Acres", num_colors = 1)
```



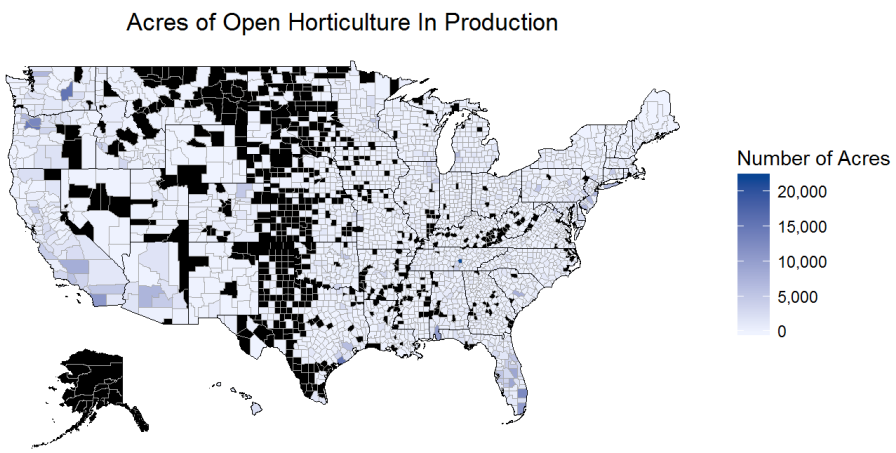
At the county level, it appears there is more acreage of horticulture open in Southern California (vs Central California's farming belt), with some in the Northwest US and a bright blue dot in the center of Tennesee. There's also a peculiar lack of acreage straight down the middle of the United States.

```
HC_HortProd <- honey_county2[which(honey_county2$DataItem == 'HORTICULTURE TOTALS, (EXCL CUT TREES), IN THE OPEN - ACRES IN PRODUC
TION'), c(7,10)]

colnames(HC_HortProd) <- c('region', 'value')

HC_HortPlot <- HC_HortProd %>%
  group_by(region) %>%
  summarize(value = sum(value))

county_choropleth(HC_HortPlot, title = "Acres of Open Horticulture In Production", legend = "Number of Acres", num_colors = 1)
```



Data Summary

Just to recap, so far we've managed to pull in our data from the NASS website, clean it (remove some character values, set others to a state average amount, remove punctuation, set values to integers, and set our state level numbers to their log form), organize it to our needs (sort by state), tidy'd it so that each data item has its own column, did some futher cleaning (removed columns without values for **HONEY - PRODUCTION, MEASURED IN LB** for counties) and explored some interesting visualizations for the data. We saw that the number of honey bees has been dynamic in that the number of colonies decreased and increased, next we will see how the amount of honey per colony has changed over the years. As we're not performing a full time-series analysis on this data, we should simply keep this trend in mind for our conculsions. We also see that some states have more expenditures in chemicals and fertilizers than others, and we've seen that there are plenty of acres of open horticulture in production for our wonderful bees to pollinate.

So let's move on. The fun's over, it's been great, but let's get back to business. How does all of this data come together? Is it possible to create a linear model to predict future honey production?

Part 4 - Inference

By using statistical measures we can attempt to use our data to study the relationship between variables and make casual links between them to infer information in the future. Essentially, we hope to use our data to predict the future. The tools we will use to do this include [1] summary statistics, [2] hypothesis testing and confidence intervals, [3] simulations based on inference, [4] linear regression, and [5] multiple linear regression.

Summary Statistics

Let's check out some summary statistics on our selected data: "HONEY - PRODUCTION, MEASURED IN LB" or "HONEY - PRODUCTION, MEASURED IN LB / COLONY".

```
summary(honey_county3$'HONEY - PRODUCTION, MEASURED IN LB')
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6    2932    9046   73540   50340 2989000
```

```
summary(honey_state3$'HONEY - PRODUCTION, MEASURED IN LB')
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    9.00   13.00   14.00   15.62   15.00   34.00
```

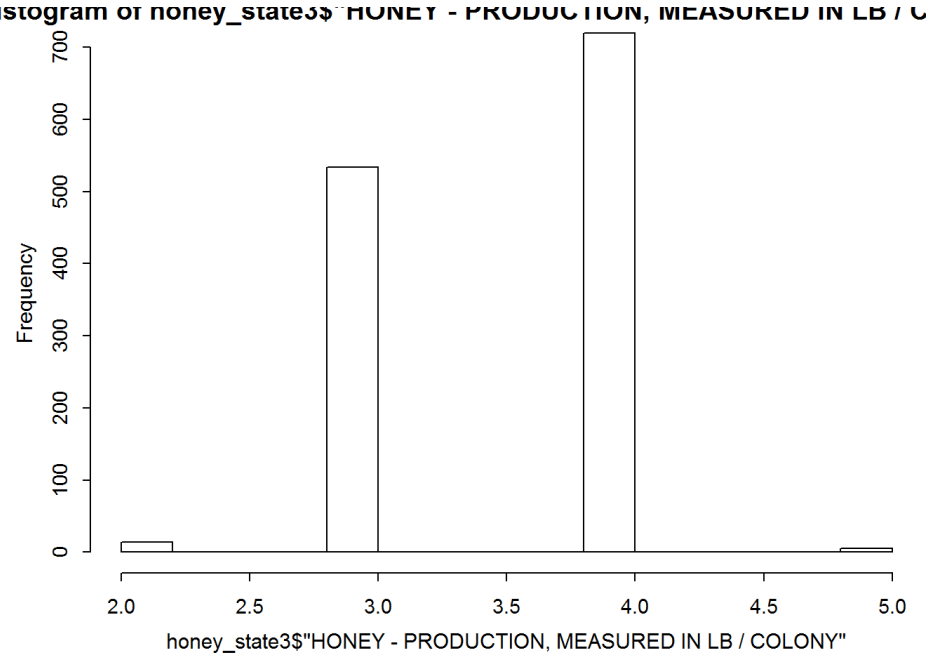
Our county level data has definite outliers and skew, given the spread between the 3rd quartile and the maximum integer, as well as the spread in the median and mean. Same with our state level data (remember, it's already in log form). Let's see how honey production per colony appears:

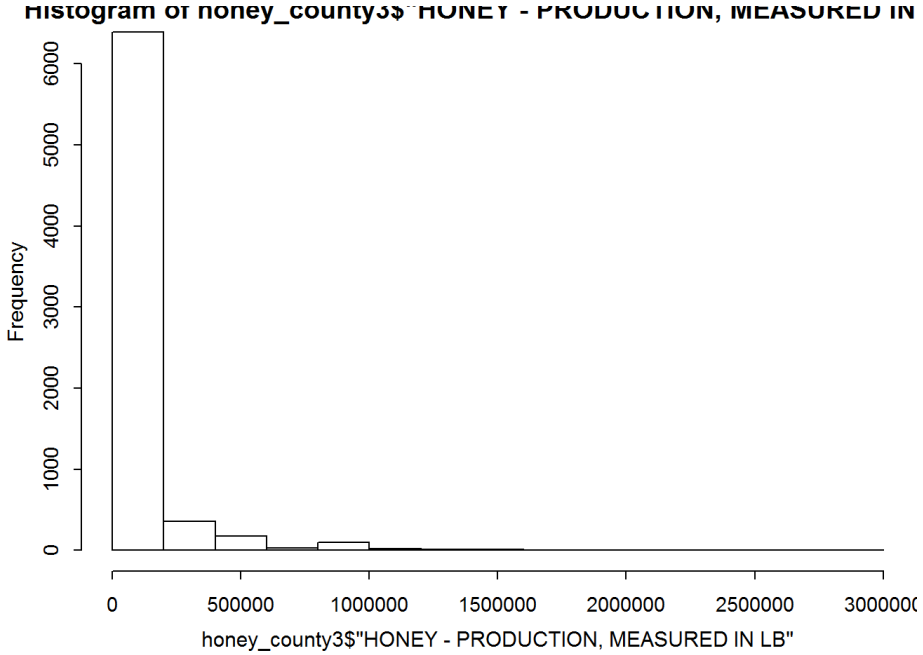
```
summary(honey_state3$'HONEY - PRODUCTION, MEASURED IN LB / COLONY')
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   3.000   4.000   3.562   4.000   5.000
```

Much more normal in appearance, but that would be assumed since all values are in log form. A histogram confirms both results:

```
hist(honey_state3$'HONEY - PRODUCTION, MEASURED IN LB / COLONY')
hist(honey_county3$'HONEY - PRODUCTION, MEASURED IN LB')
```





We can also recall from our previous exploration of the data that there are many more observation variables in our county set than in our state set:

```
length(honey_county3$'HONEY - PRODUCTION, MEASURED IN LB')
```

```
## [1] 7098
```

```
length(honey_state3$'HONEY - PRODUCTION, MEASURED IN LB / COLONY')
```

```
## [1] 1273
```

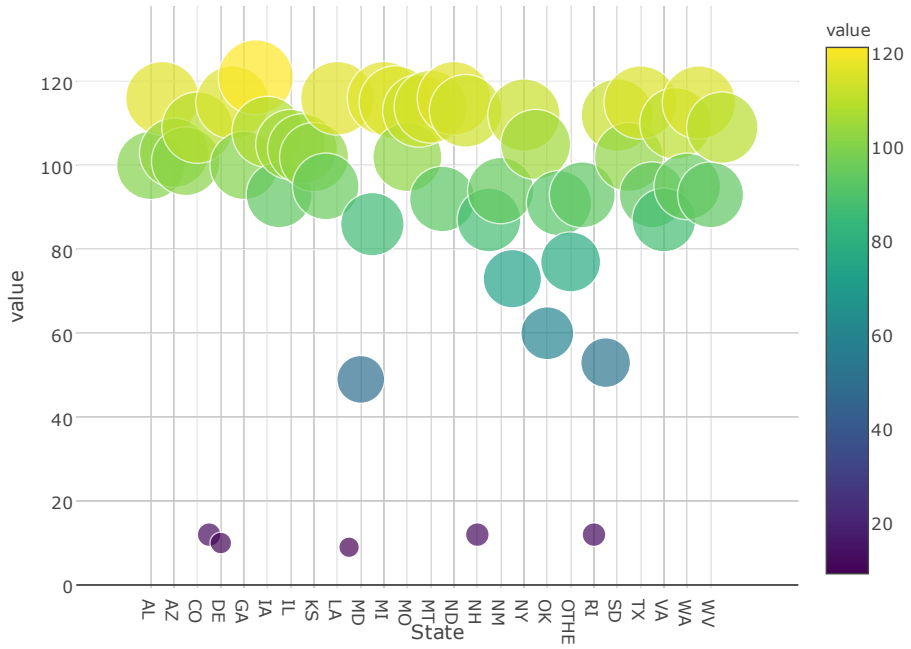
Another intersting visual we can look at in line with this information is to see how honey production varies per state and year using the **plotly** package. We'll first need to know which column 'HONEY - PRODUCTION, MEASURED IN LB / COLONY' is stored in:

```
which(colnames(honey_state3) =='HONEY - PRODUCTION, MEASURED IN LB / COLONY')
```

```
## [1] 39
```

Which is column 39:

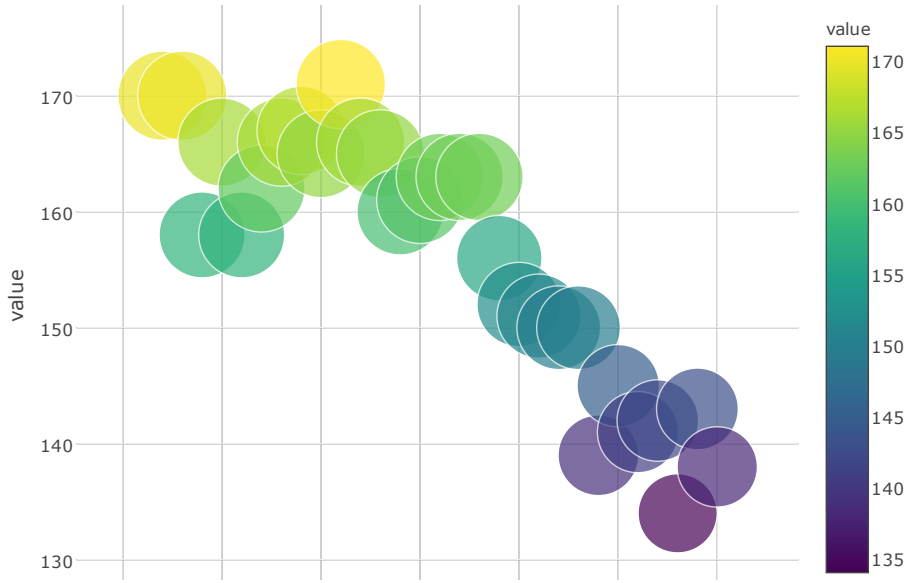
```
StateProd <- honey_state3[,c(2,39)] %>%
  group_by(State)
colnames(StateProd) <- c("State","value")
StateProd <- StateProd %>%
  group_by(State) %>%
  summarise(value = sum(value))
plot_ly(StateProd, x = State, y = value, text = paste("State: ", State), mode = "markers", color = value, size = value)
```



There seems to be a big discrepency between states and how much honey is produced in each. Interesting.

How has honey production varied throughout the years since 1987?

```
StateProd2 <- honey_state3[,c(1,39)] %>%
  group_by(Year)
colnames(StateProd2) <- c("Year","value")
StateProd2 <- StateProd2 %>%
  group_by(Year) %>%
  summarise(value = sum(value))
plot_ly(StateProd2, x = Year, y = value, text = paste("Year: ", Year), mode = "markers", color = value, size = value)
```



I think this graph is unmistakable.

Moving forward and looking back, with the length of our data we can create sample sizes useful for inference. Let's use a random sample size of 60 (randomly selected size).

Determining the Status Quo: Expected Honey Production per Colony

One question we can try to answer at this stage is: What should the "typical" amount of honey produced per colony be? Since we only have data on this category in the state set, we will focus our attention there. Also, we know that our full data population has 1273 variables and, using a random sample of 60, we can set up some of the ground work:

```
population <- honey_state3$'HONEY - PRODUCTION, MEASURED IN LB / COLONY'
samp_mean <- rep(NA, 50)
samp_sd <- rep(NA, 50)
n <- 60
```

These boundaries are simply pulling our population (1273 variables of lb's of Honey Produced per Colony), and determining that we will have 50 test cases with 60 observations per each case. We can then use these to loop through and build 50 random samples of 60 observations, at the same time calculating the mean and standard deviation of the sample groups and storing them into their own vectors for **samp_mean** and **samp_sd** respectively:

```
for(i in 1:50){
  samp <- sample(population, n)
  samp_mean[i] <- mean(samp)
  samp_sd[i] <- sd(samp)
}
```

With our sample groups built and means / standard deviations calculated we can see what a 95% confidence interval would be for the amount of honey produced per colony:

```
lower_vector <- samp_mean - 1.96 * samp_sd / sqrt(n)
upper_vector <- samp_mean + 1.96 * samp_sd / sqrt(n)

c(lower_vector[1], upper_vector[1])
```

```
## [1] 3.276266 3.590401
```

```
mean(population)
```

```
## [1] 3.562451
```

As we can see, our population average falls right inside the inveral which is expected. Let's also see what a 90% CI would look like:

```
lower_vector <- samp_mean - 1.645 * samp_sd / sqrt(n)
upper_vector <- samp_mean + 1.645 * samp_sd / sqrt(n)

c(lower_vector[1], upper_vector[1])
```

```
## [1] 3.301509 3.565158
```

```
mean(population)
```

```
## [1] 3.562451
```

Inference Testing: Expected Honey per Colony

Let's try to test our honey production against itself:

- Null Hypothesis: The amount of honey produced per colony of bees significantly shrunk from 1987 to 2015.
- Alternative Hypothesis: The amount of honey produced per colony of bees did not significantly shrink from 1987 to 2015

Let's use the same parameters as before with a slight change in that we will use a sample size of 25 instead of 50 (due to less variables, keeping our number of sample populations at 50) and calculate the both the 1987 average and 2015 average. The 2015 average will be used to evaulate our hypothesis.

```
population2 <- honey_state3[,c(1,39)]
population2 <- unlist(population2[population2$Year == 1987,2])

population3 <- honey_state3[,c(1,39)]
population3 <- unlist(population3[population3$Year == 2015,2])

samp_mean <- rep(NA, 50)
samp_sd <- rep(NA, 50)
n <- 25

for(i in 1:50){
  samp <- sample(population2, n)
  samp_mean[i] <- mean(samp)
  samp_sd[i] <- sd(samp)
}

lower_vector <- samp_mean - 1.96 * samp_sd / sqrt(n)
upper_vector <- samp_mean + 1.96 * samp_sd / sqrt(n)

c(lower_vector[1], upper_vector[1])
```

```
## [1] 3.290309 3.749691
```

```
mean(population2)
```

```
## [1] 3.469388
```

```
mean(population3)
```

```
## [1] 3.365854
```

It appears that it is a very close call. Running the simulation over and over yields different results (naturally, as it is a *random* sample) with some results showing a fit within the confidence interval, and some results showing just outside of the interval. In either case I believe it is too close to call, and we therefore fail to reject the null hypothesis. In other words, there could be a significant drop in honey production throughout the years.

All of this can more easily be summarized using the package previously loaded from the IS 606 Course. Solving for Confidence Intervals:

```
inference(y = population, est = "mean", type = "ci", null = 0, alternative = "twosided", method = "theoretical")
```

```
## Single mean
## Summary statistics:
```

```
## mean = 3.5625 ;  sd = 0.5255 ;  n = 1273
## Standard error = 0.0147
## 95 % Confidence interval = ( 3.5336 , 3.5913 )
```

```
inference(y = population2, est = "mean", type = "ci", null = 0, alternative = "twosided", method = "theoretical")
```

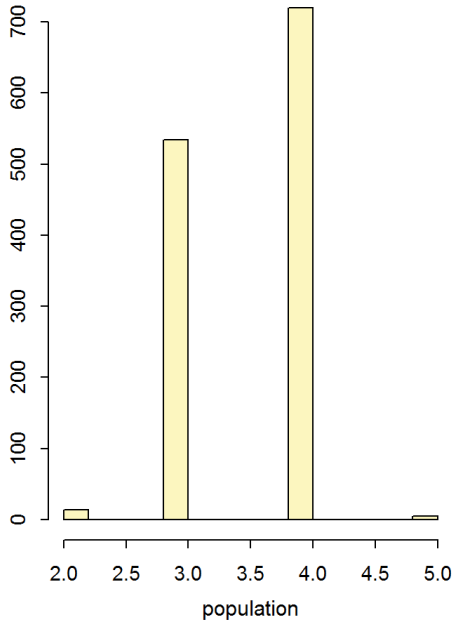
```
## Single mean
## Summary statistics:
```

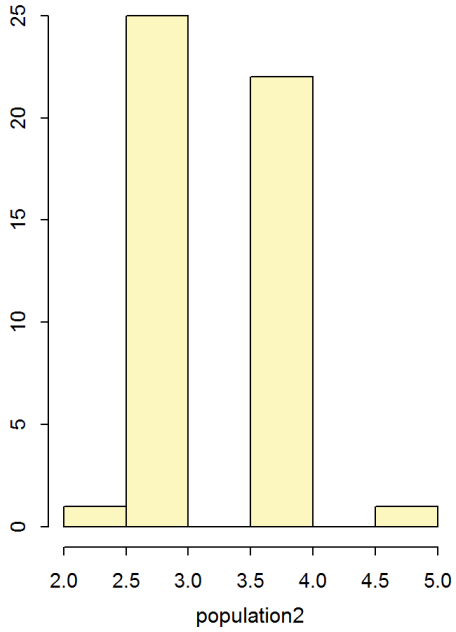
```
## mean = 3.4694 ;  sd = 0.581 ;  n = 49
## Standard error = 0.083
## 95 % Confidence interval = ( 3.3067 , 3.6321 )
```

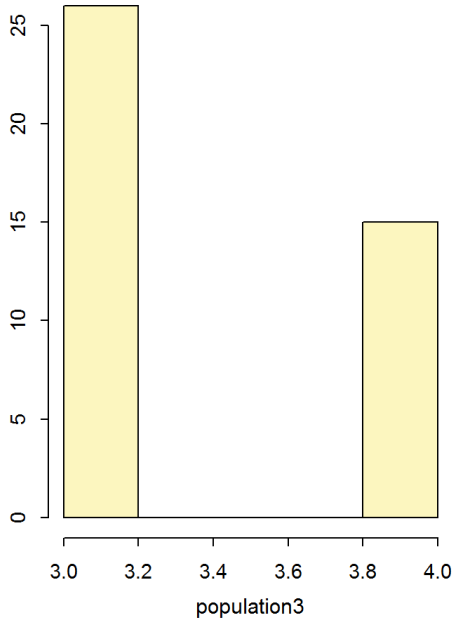
```
inference(y = population3, est = "mean", type = "ci", null = 0, alternative = "twosided", method = "theoretical")
```

```
## Single mean
## Summary statistics:
```

```
## mean = 3.3659 ;  sd = 0.4877 ;  n = 41
## Standard error = 0.0762
## 95 % Confidence interval = ( 3.2166 , 3.5151 )
```


0
100
200
300
400
500
600
700
2.0
2.5
3.0
3.5
4.0
4.5
5.0
population


0
5
10
15
20
25
2.0
2.5
3.0
3.5
4.0
4.5
5.0
population2



To solve for hypothesis tests, I'll combine the two smaller populations (for years 1987 and 2015) and see how they compare:

```
population4 <- honey_state3[,c(1,39)]
population4 <- population4[population4$Year == 1987,1:2]
population5 <- honey_state3[,c(1,39)]
population5 <- population5[population5$Year == 2015,1:2]
population6 <- bind_rows(population4, population5)
colnames(population6) <- c("Year","Value")
```

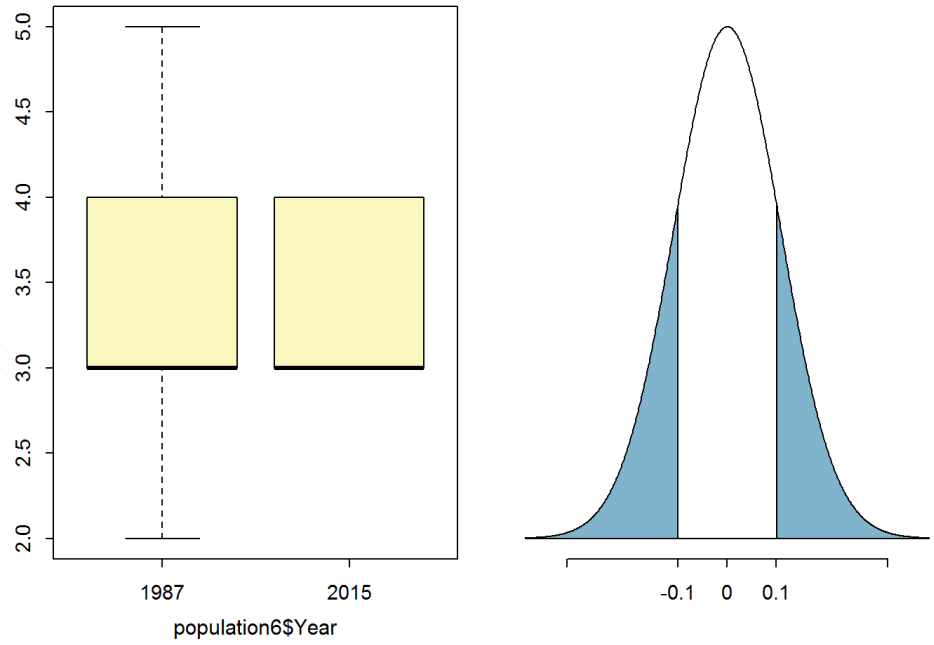
To run the hypothesis tests:

```
inference(y = population6$Value, x = population6$Year, est = "mean", type = "ht", null = 0, alternative = "twosided", method = "th
eoretical")
```

```
## Warning: Explanatory variable was numerical, it has been converted to
## categorical. In order to avoid this warning, first convert your explanatory
## variable to a categorical variable using the as.factor() function.
```

```
## Response variable: numerical, Explanatory variable: categorical
## Difference between two means
## Summary statistics:
## n_1987 = 49, mean_1987 = 3.4694, sd_1987 = 0.581
## n_2015 = 41, mean_2015 = 3.3659, sd_2015 = 0.4877
```

```
## Observed difference between means (1987-2015) = 0.1035
##
## H0: mu_1987 - mu_2015 = 0
## HA: mu_1987 - mu_2015 != 0
## Standard error = 0.113
## Test statistic: Z =  0.919
## p-value =  0.358
```



As the results of this inference test for the hypothesis goes, we again fail to reject the null since the p-value is fairly large (>0.05). Failing to reject the null in this case means that we cannot, at this point, determine that there was not a significant change in the amount of honey produced per each colony of bees from 1987 to 2015. (Please keep in mind the double negative, they can get confusing–I'm simply saying that there may in fact have been a significant change).

Linear Regression: Drawing a Line to the Hive

With linear regression, I'm interested in seeing the linear relationship between honey produced per colony and another statistic. But how to pick one… We could use a random number:

```
rando <- sample(ncol(honey_state3), 1)
rando
```

```
## [1] 106
```

But with so many different Data Items in our data set we're likely to get one that's fairly riduculous (I'm looking at you **HORTICULTURE TOTALS, WHOLESALE, LANDSCAPE REDISTRIBUTION YARDS - SALES, MEASURED IN $**!) Let's use one could be interesting to see: **HONEY, BEE COLONIES - INVENTORY, MEASURED IN COLONIES**.

First we'll take a look at the summary statistics on this new field, then we can see if the relationship appears linear. I'm going to create a new data frame and remove all null values as well (changing them to zero will skew our data):

```
which(colnames(honey_state3) =='HONEY, BEE COLONIES - INVENTORY, MEASURED IN COLONIES')
```

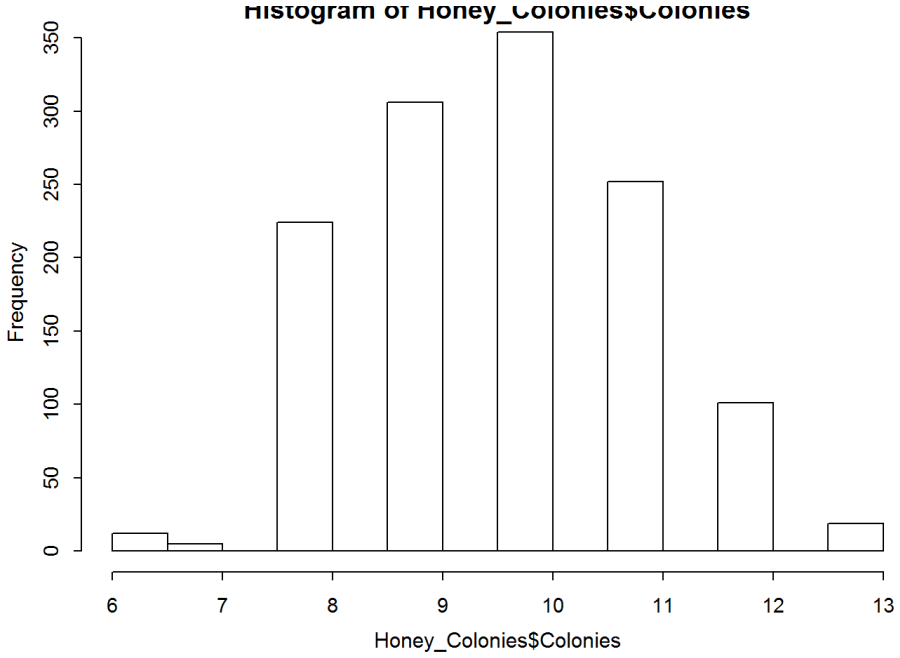
```
## [1] 41
```

```
Honey_Colonies <- honey_state3[,c(1,2,41,39)]
colnames(Honey_Colonies) <- c("Year", "State", "Colonies","Honey")
Honey_Colonies <- Honey_Colonies[complete.cases(Honey_Colonies[,3:4]),]

summary(Honey_Colonies$Colonies)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6.00    9.00   10.00    9.76   11.00   13.00
```

```
hist(Honey_Colonies$Colonies)
```



It looks like the distribution is much tighter in this set of data.

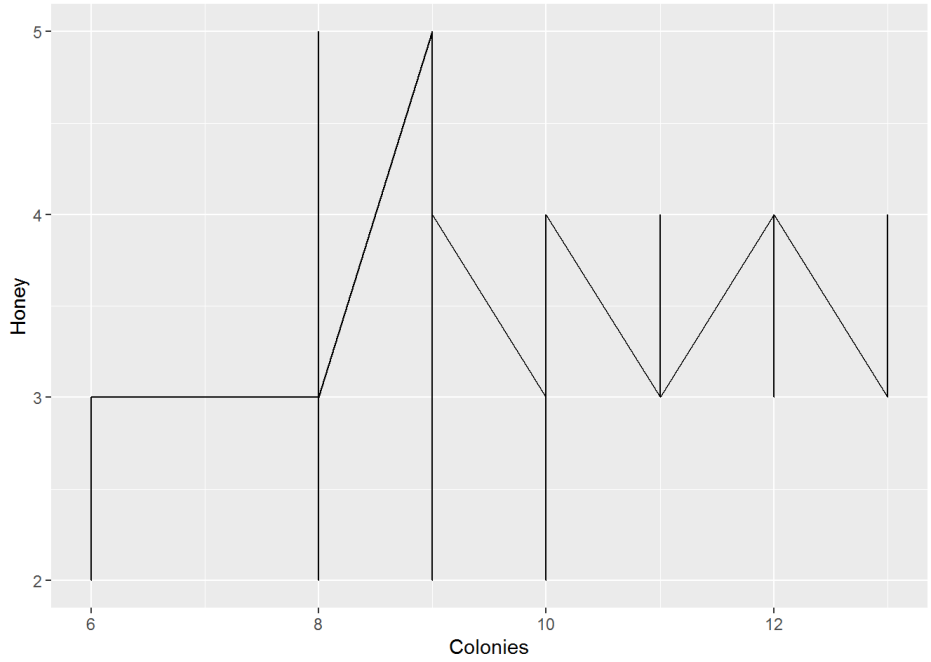
```
#calculate correlation
cor(Honey_Colonies$Honey, Honey_Colonies$Colonies)
```

```
## [1] 0.2986684
```

There is a correlation of 0.2987, indicating a positive correlation but not an entirely strong correlation.

We can also check it on a plot:

```
ggplot(Honey_Colonies, aes(x=Colonies, y=Honey)) + geom_line()
```



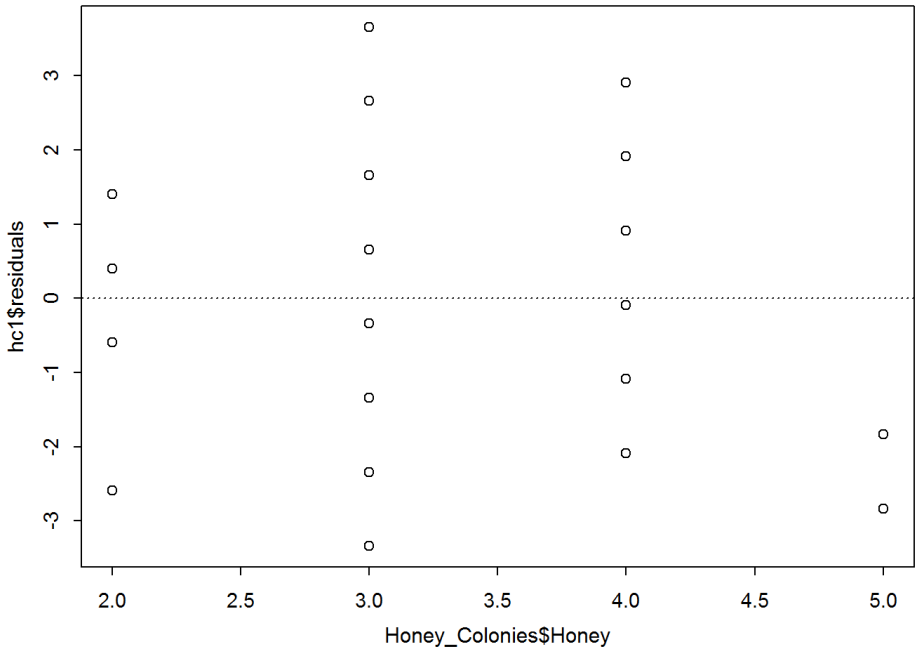
But we don't learn a whole lot. There seems to be a higher spread at log 8-9 colonies and a majority of the grouping at log 3-4 honey production per colony.

More interestingly (and more usefully) if we plot the residuals with a we can see a much clearer trend of the fit between these two variables:

```
hc1 <- lm(Colonies ~ Honey, data=Honey_Colonies)
summary(hc1)
```

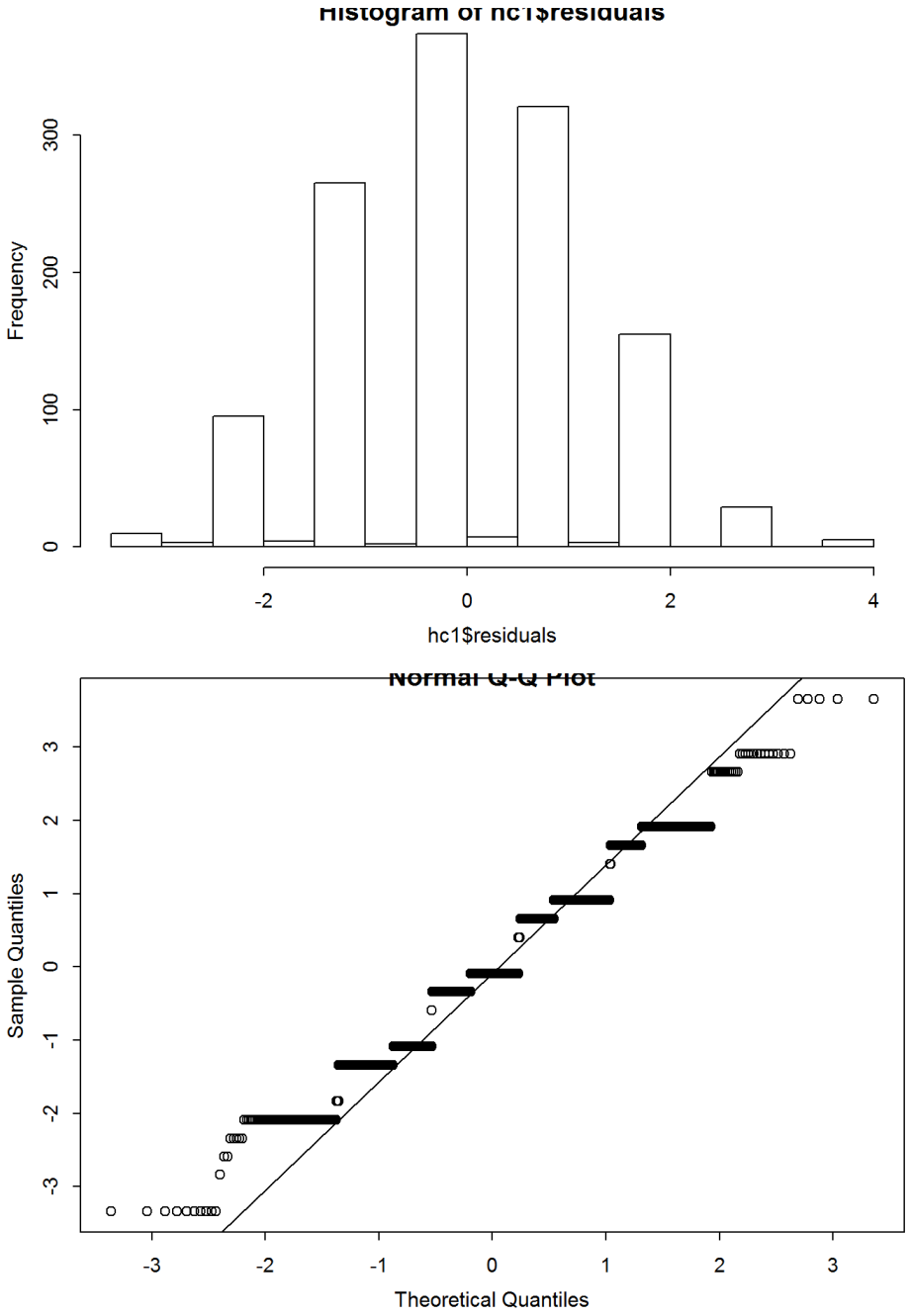
```
##
## Call:
## lm(formula = Colonies ~ Honey, data = Honey_Colonies)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3400 -1.0861 -0.0861  0.9139  3.6600
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.10151    0.24082   29.49   <2e-16 ***
## Honey        0.74615    0.06688   11.16   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.253 on 1271 degrees of freedom
## Multiple R-squared:  0.0892, Adjusted R-squared:  0.08849
## F-statistic: 124.5 on 1 and 1271 DF,  p-value: < 2.2e-16
```

```
plot(hc1$residuals ~ Honey_Colonies$Honey)
abline(h=0, lty=3)
```



What we see is a fairly even difference in the residuals (showing no evident patterns), indicating that it could be reasonable to use a linear regression to fit these two data points. There could be interest at the high end of the chart at log 5 Honey Produced, so we can take a look at another two types of charts: histograms and a normal probability plot of the residuals.

```
hist(hc1$residuals)
qqnorm(hc1$residuals)
qqline(hc1$residuals)
```



It appears that the nearly normal residuals condition passes, as does the constant variabilty condition (despite slight curvature) with an “iffy” result for the linearity of the plot indicating a possible but an unclear positive relationship, however I don’t believe that it passes the condition for independent observations. These four conditions are generally required for fitting a least squares line and the conditions for independent observations warn of applying a linear regression on a time series, which this certainly is. With the warning in hand, ignoring the independence condition, we can identify the least squares line as (from our linear model summary):

$$\hat{lbsofhoneyproducedpercolony} = -7.10151 + 0.74615 * colonies$$

Note: Remember that all of these data points are in log form, and the data is time series so a linear regression may not be the best estimator.

Linear Regression Conclusion: What we see is that the positive linear relationship between the amount of honey produced per colony in lb’s and the number of bee colonies. This positive correlation indicates that for every increase in colony there is a corresponding increase in the amount of honey produced per colony and the number of colonies. *TL;DR: More honey bee colonies: the more honey each colony produces.**

Combining this conculsion with our previous analysis of the number of bee colonies, we could be facing an exponential decline in honey.

Multiple Linear Regression: Drawing a Big Line to the Hive

With our newfound understanding of the relationship between honey produced per colony and the number of colonies, we are no closer to answering our ultimate question: do we really rely on the bees?

There are three ways we can perform the regression analysis: [1] the full model, [2] the backward elimination model, and [3] the forward selection model. To save time, each of these models were run on the backend with the backup data and process in a seperate file stored in our GitHub folder for Appendix Two: Multiple Regression Analysis (http://rpubs.com/chrisgmartin/HISB_AppendixTwo)

Full Model

The full model uses every available explanatory variable to attempt to predict the variable at hand. As our state dataset includes an intimidating 108 different variables, using the full model is an incredibly messy task. For ease, we'll switch to the county dataset which includes "only" 21 DataItem's, remove all incomplete cases, and look at total production measured in lb's (rather than production per colony. Note that this case requires us to force null values to 0 despite my previous warnings not to do so, after removing cases with incomplete results (N/A's or Null Values) for Honey Produced because there are simply not enough data points without null values in one of the many columns (there are zero, in fact):

```
HoneyC_FullModel <- honey_county3[,c(1,5,6:26)]
colnames(HoneyC_FullModel) <- c("Year", "CountyANSI", "ChemExp", "ChemOps", "CropOps", "CropSales", "CropOrgOps", "CropOrgSales",
"FertExp", "FertOps", "HoneyOpsProd", "HoneyOpsSales", "HoneyProd", "HoneySales", "HoneyColOps", "HoneyColSales", "HortExcTVSTOp
s", "HortExcTVSTSales", "HortExcTAcres", "HortExcTIrgAcres", "HortExcTIrgOps", "HortUndProt")

#return only complete cases
HoneyC_FullModel <- HoneyC_FullModel[complete.cases(HoneyC_FullModel[,13]),]
HoneyC_FullModel[is.na(HoneyC_FullModel)] <- 0

#export table for use in Appendix Two
#write.csv(HoneyC_FullModel, file = 'D:/GitHub/HoneyIShrunkTheBees/Appendix/HoneyC_FullModel.csv')

m_full <- lm(HoneyProd ~ ChemExp + ChemOps + CropOps + CropSales + CropOrgOps + CropOrgSales + FertExp + FertOps + HoneyOpsProd +
HoneyOpsSales + HoneySales + HoneyColOps + HoneyColSales + HortExcTVSTOps + HortExcTVSTSales + HortExcTAcres + HortExcTIrgAcres +
HortExcTIrgOps + HortUndProt, data = HoneyC_FullModel)
summary(m_full)
```

```
##
## Call:
## lm(formula = HoneyProd ~ ChemExp + ChemOps + CropOps + CropSales +
##     CropOrgOps + CropOrgSales + FertExp + FertOps + HoneyOpsProd +
##     HoneyOpsSales + HoneySales + HoneyColOps + HoneyColSales +
##     HortExcTVSTOps + HortExcTVSTSales + HortExcTAcres + HortExcTIrgAcres +
##     HortExcTIrgOps + HortUndProt, data = HoneyC_FullModel)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -718898  -49324  -27017    2728 2106250
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -1.921e+04  2.069e+04  -0.928  0.35335
## ChemExp          2.938e-03  3.895e-04   7.542 5.22e-14 ***
## ChemOps          2.137e+02  2.406e+01   8.881  < 2e-16 ***
## CropOps          4.327e+01  1.724e+01   2.510  0.01209 *
## CropSales        3.304e+03  1.364e+03   2.422  0.01545 *
## CropOrgOps      -1.049e+02  2.520e+02  -0.417  0.67705
## CropOrgSales     1.311e-02  1.852e-03   7.082 1.56e-12 ***
## FertExp         -1.826e-03  3.947e-04  -4.627 3.78e-06 ***
## FertOps         -2.023e+02  1.984e+01 -10.193  < 2e-16 ***
## HoneyOpsProd     3.263e+03  3.789e+02   8.610  < 2e-16 ***
## HoneyOpsSales   -7.302e+03  5.190e+02 -14.070  < 2e-16 ***
## HoneySales       4.914e-01  8.573e-03  57.325  < 2e-16 ***
## HoneyColOps     -7.638e+02  2.500e+03  -0.306  0.75999
## HoneyColSales    2.603e+02  1.877e+01  13.869  < 2e-16 ***
## HortExcTVSTOps  -2.331e+02  7.446e+01  -3.130  0.00175 **
## HortExcTVSTSales  2.162e-04  1.111e-04   1.945  0.05181 .
## HortExcTAcres   -4.309e+02  2.002e+02  -2.153  0.03135 *
## HortExcTIrgAcres -1.714e+01  6.103e+00  -2.809  0.00498 **
## HortExcTIrgOps   3.256e+01  8.003e+00   4.068 4.79e-05 ***
## HortUndProt      1.635e+03  3.627e+02   4.507 6.69e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 146100 on 7078 degrees of freedom
## Multiple R-squared:  0.4337, Adjusted R-squared:  0.4322
## F-statistic: 285.3 on 19 and 7078 DF,  p-value: < 2.2e-16
```

This full model can give us a model that is simply too cumbersome for casual output, but the main takeaway can be our Adjusted R-squared of 0.4322. This value we can use to compare how effective the model is likely to be against our other potential models: backward elimination and forward selection.

Backward Elimination

With backward elimination, we start with the full model and move backward by eliminating those variables that reduce our Adjusted R-squared until we get a maximized Adjusted R-squared model. To save time, I've run this on the backend on Appendix Two, and show only the results here. In fact, there was only a very minimal increase in the Adjusted R-squared so the backward model looks strikingly similar to the full model. We've really only ruled out any impact to our model by eliminating the number of Crop Organizations with Operations and the number of Honey Colonies with Operations.

```
m_backward <- lm(HoneyProd ~ ChemExp + ChemOps + CropOps + CropSales + CropOrgSales + FertExp + FertOps + HoneyOpsProd + HoneyOpsS
ales + HoneySales + HoneyColSales + HortExcTVSTOps + HortExcTVSTSales + HortExcTAcres + HortExcTIrgAcres + HortExcTIrgOps + HortUn
dProt, data = HoneyC_FullModel)
summary(m_backward)
```

```
##
## Call:
## lm(formula = HoneyProd ~ ChemExp + ChemOps + CropOps + CropSales +
##     CropOrgSales + FertExp + FertOps + HoneyOpsProd + HoneyOpsSales +
##     HoneySales + HoneyColSales + HortExcTVSTOps + HortExcTVSTSales +
##     HortExcTAcres + HortExcTIrgAcres + HortExcTIrgOps + HortUndProt,
##     data = HoneyC_FullModel)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -712524  -49322  -26912    2842 2104209
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -1.893e+04  2.068e+04  -0.915 0.360125
## ChemExp           2.940e-03  3.894e-04   7.550 4.92e-14 ***
## ChemOps           2.154e+02  2.385e+01   9.030  < 2e-16 ***
## CropOps           4.217e+01  1.695e+01   2.488 0.012886 *
## CropSales         3.293e+03  1.364e+03   2.415 0.015766 *
## CropOrgSales      1.271e-02  1.572e-03   8.087 7.12e-16 ***
## FertExp          -1.821e-03  3.943e-04  -4.619 3.92e-06 ***
## FertOps          -2.032e+02  1.973e+01 -10.296  < 2e-16 ***
## HoneyOpsProd      3.206e+03  3.597e+02   8.914  < 2e-16 ***
## HoneyOpsSales    -7.200e+03  4.809e+02 -14.971  < 2e-16 ***
## HoneySales        4.913e-01  8.570e-03  57.334  < 2e-16 ***
## HoneyColSales     2.580e+02  1.761e+01  14.654  < 2e-16 ***
## HortExcTVSTOps   -2.420e+02  7.189e+01  -3.367 0.000764 ***
## HortExcTVSTSales  2.195e-04  1.109e-04   1.978 0.047959 *
## HortExcTAcres    -4.177e+02  1.963e+02  -2.128 0.033337 *
## HortExcTIrgAcres -1.723e+01  6.100e+00  -2.824 0.004759 **
## HortExcTIrgOps    3.262e+01  8.001e+00   4.076 4.62e-05 ***
## HortUndProt       1.632e+03  3.614e+02   4.516 6.40e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 146100 on 7080 degrees of freedom
## Multiple R-squared:  0.4337, Adjusted R-squared:  0.4323
## F-statistic: 318.9 on 17 and 7080 DF,  p-value: < 2.2e-16
```

Our new best model came from the Backward Elimination method and gave us an Adjusted R-squared of 0.4323.

Forward Selection

With forward selection we start from the start by comparing each variable in a linear model and adding new variables if they improve our Adjusted R-squared. As it turns out, our final model from forward selection is the same for backward elimination. This isn't a surprise exactly, but it gives us some understanding of what factors relate to one another.

```
m_forward <- lm(HoneyProd ~ HortUndProt + HoneySales + HoneyColSales + ChemExp + HoneyOpsSales + CropOrgSales + HoneyOpsProd + Hor
tExcTAcres + ChemOps + FertOps + FertExp + HortExcTIrgOps + CropSales + HortExcTVSTOps + HortExcTIrgAcres + CropOps + HortExcTVSTS
ales, data = HoneyC_FullModel)
summary(m_forward)
```

```
##
## Call:
## lm(formula = HoneyProd ~ HortUndProt + HoneySales + HoneyColSales +
##     ChemExp + HoneyOpsSales + CropOrgSales + HoneyOpsProd + HortExcTAcres +
##     ChemOps + FertOps + FertExp + HortExcTIrgOps + CropSales +
##     HortExcTVSTOps + HortExcTIrgAcres + CropOps + HortExcTVSTSales,
##     data = HoneyC_FullModel)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -712524  -49322  -26912    2842 2104209
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -1.893e+04  2.068e+04  -0.915 0.360125
## HortUndProt       1.632e+03  3.614e+02   4.516 6.40e-06 ***
## HoneySales        4.913e-01  8.570e-03  57.334  < 2e-16 ***
## HoneyColSales     2.580e+02  1.761e+01  14.654  < 2e-16 ***
## ChemExp           2.940e-03  3.894e-04   7.550 4.92e-14 ***
## HoneyOpsSales    -7.200e+03  4.809e+02 -14.971  < 2e-16 ***
## CropOrgSales      1.271e-02  1.572e-03   8.087 7.12e-16 ***
## HoneyOpsProd      3.206e+03  3.597e+02   8.914  < 2e-16 ***
## HortExcTAcres    -4.177e+02  1.963e+02  -2.128 0.033337 *
## ChemOps           2.154e+02  2.385e+01   9.030  < 2e-16 ***
## FertOps          -2.032e+02  1.973e+01 -10.296  < 2e-16 ***
## FertExp          -1.821e-03  3.943e-04  -4.619 3.92e-06 ***
## HortExcTIrgOps    3.262e+01  8.001e+00   4.076 4.62e-05 ***
## CropSales         3.293e+03  1.364e+03   2.415 0.015766 *
## HortExcTVSTOps   -2.420e+02  7.189e+01  -3.367 0.000764 ***
## HortExcTIrgAcres -1.723e+01  6.100e+00  -2.824 0.004759 **
## CropOps           4.217e+01  1.695e+01   2.488 0.012886 *
## HortExcTVSTSales  2.195e-04  1.109e-04   1.978 0.047959 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 146100 on 7080 degrees of freedom
## Multiple R-squared:  0.4337, Adjusted R-squared:  0.4323
## F-statistic: 318.9 on 17 and 7080 DF,  p-value: < 2.2e-16
```

Checking Model Assumptions

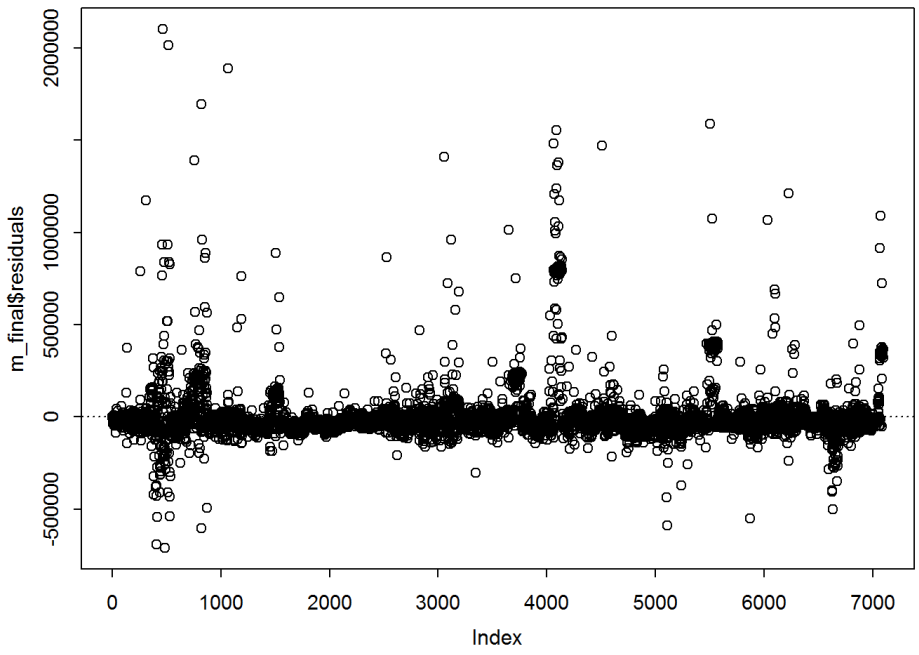
Now that we have our multiple regression analysis complete, we'll need to check the model assumptions. First, set the final model:

```
m_final <- lm(HoneyProd ~ HortUndProt + HoneySales + HoneyColSales + ChemExp + HoneyOpsSales + CropOrgSales + HoneyOpsProd + HortE
xcTAcres + ChemOps + FertOps + FertExp + HortExcTIrgOps + CropSales + HortExcTVSTOps + HortExcTIrgAcres + CropOps + HortExcTVSTSal
es, data = HoneyC_FullModel)
summary(m_final)
```

```
##
## Call:
## lm(formula = HoneyProd ~ HortUndProt + HoneySales + HoneyColSales +
##     ChemExp + HoneyOpsSales + CropOrgSales + HoneyOpsProd + HortExcTAcres +
##     ChemOps + FertOps + FertExp + HortExcTIrgOps + CropSales +
##     HortExcTVSTOps + HortExcTIrgAcres + CropOps + HortExcTVSTSales,
##     data = HoneyC_FullModel)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -712524  -49322  -26912    2842 2104209
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -1.893e+04  2.068e+04  -0.915 0.360125
## HortUndProt       1.632e+03  3.614e+02   4.516 6.40e-06 ***
## HoneySales        4.913e-01  8.570e-03  57.334  < 2e-16 ***
## HoneyColSales     2.580e+02  1.761e+01  14.654  < 2e-16 ***
## ChemExp           2.940e-03  3.894e-04   7.550 4.92e-14 ***
## HoneyOpsSales    -7.200e+03  4.809e+02 -14.971  < 2e-16 ***
## CropOrgSales      1.271e-02  1.572e-03   8.087 7.12e-16 ***
## HoneyOpsProd      3.206e+03  3.597e+02   8.914  < 2e-16 ***
## HortExcTAcres    -4.177e+02  1.963e+02  -2.128 0.033337 *
## ChemOps           2.154e+02  2.385e+01   9.030  < 2e-16 ***
## FertOps          -2.032e+02  1.973e+01 -10.296  < 2e-16 ***
## FertExp          -1.821e-03  3.943e-04  -4.619 3.92e-06 ***
## HortExcTIrgOps    3.262e+01  8.001e+00   4.076 4.62e-05 ***
## CropSales         3.293e+03  1.364e+03   2.415 0.015766 *
## HortExcTVSTOps   -2.420e+02  7.189e+01  -3.367 0.000764 ***
## HortExcTIrgAcres -1.723e+01  6.100e+00  -2.824 0.004759 **
## CropOps           4.217e+01  1.695e+01   2.488 0.012886 *
## HortExcTVSTSales  2.195e-04  1.109e-04   1.978 0.047959 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 146100 on 7080 degrees of freedom
## Multiple R-squared:  0.4337, Adjusted R-squared:  0.4323
## F-statistic: 318.9 on 17 and 7080 DF,  p-value: < 2.2e-16
```

As we did in the linear regression model, we can check the residuals on a residual plot with a line at 0 (variable predicted fit as expected):

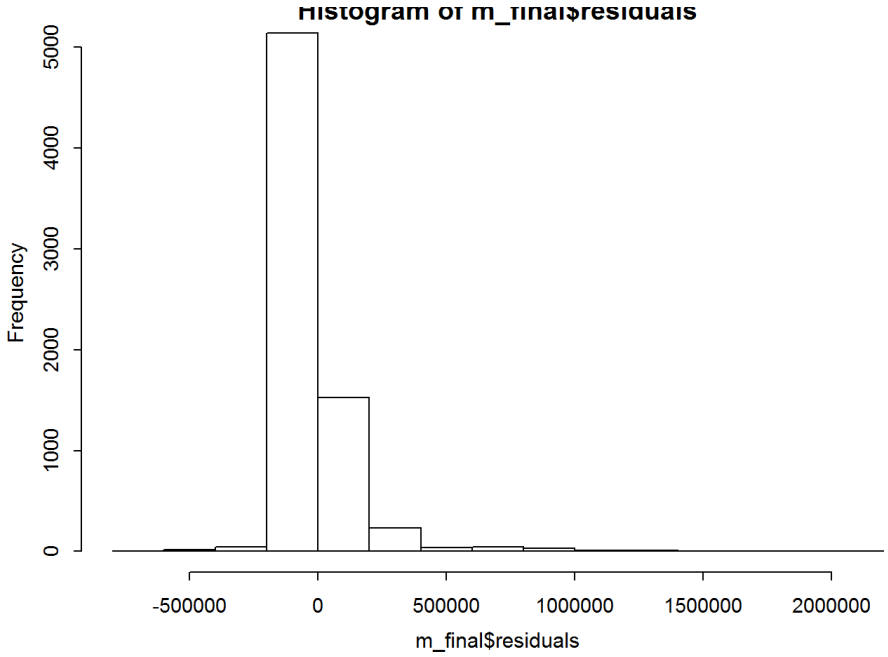
```
plot(m_final$residuals)
abline(h = 0, lty = 3)
```



There are many more residuals at the high end indicating that there is possible skew to this dataset at the high end.

A histogram makes this more apparent:

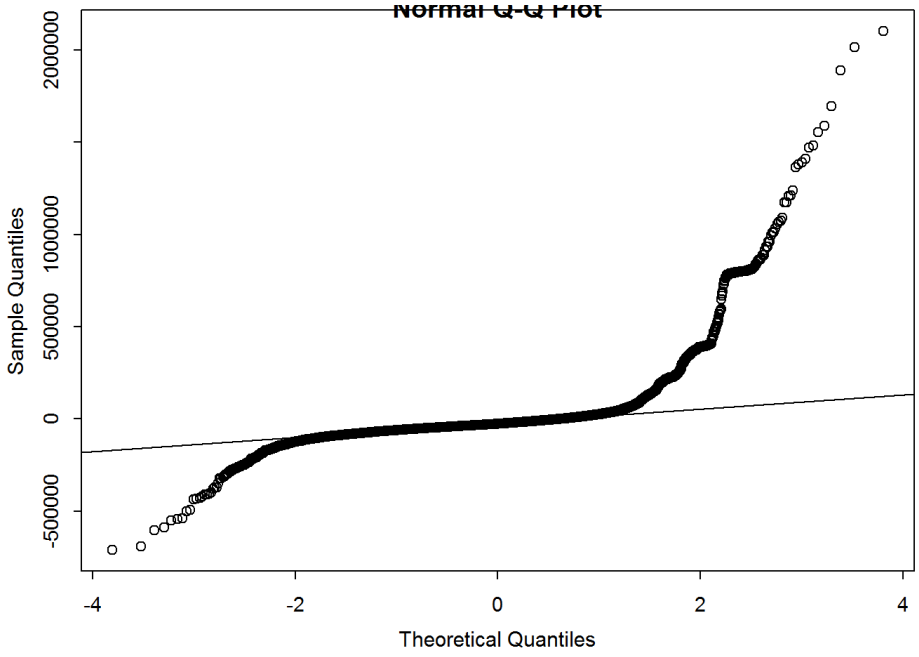
```
hist(m_final$residuals)
```



The histogram shows a definite skew at the high end.

Finally a probabilty plot shows a major concern for outliers:

```
qqnorm(m_final$residuals)
qqline(m_final$residuals)
```



With these three graphics showing similar features, our concern is confirmed: we have serious outliers and the assumptions in our multiple linear regression do not make the grade. We cannot comfortably recommend the multiple linear regression to predict the production of honey, and recommend going back to the drawing board.

Part 5 - Conclusion: Plight of the Bumblebees

I fear that we are no closer to answering the question posed at the start: Do we rely on the bees? It is perhaps due to my selection of explantory variables, or heavy skewing of the data by using averages and 0's for missing values, or it could be simply that we were dealing with a time series (square peg in a round hole). Regardless, the analysis was very successful in uncovering some interesting information in the decline of honey production per colony since 1987. Our fears may be different than what I had expected. Rather than focusing our attention on the number of bees (which we can inconclusively say we should continue to worry about), **we should focus our attention on a possible exponential decline in honey.**

Problems Encountered

As I mentioned, the biggest error I've made in this project was using a multiple linear regression analysis on a time series data. This error essentially nullified the work of the analysis as a multiple linear regression analysis is not the best type of analysis to perform on this data. Solutions would have been to use a single year's worth of data (i.e. 2012) rather than the entire dataset, but unfortunetly I did not feel like that was appropriate with the relatively limited amount of data. Of course more data can always be found, and a cost:benefit (time:accuracy) analysis should be considered when contemplating adding more data.

Next Steps

There are several directions we could go from here. One direction could be analysing the number of bees in a colony and see how they influence the amount of honey a colony can produce: is there a point of marginal returns whereby adding bees to a colony actually hinders the production of honey? How good are the bees at pollinating and how efficient are they? In the end, my goal is to save the bees and help man-kind learn to live alongside them (and nature in general).

Another analysis would be to see the reverse of what I've done here: rather than see how these variables explain the honey production of a colony, see how honey production influences the other variables. This would give us much more insight into the over-arching question I posed at the outset: Do we rely on the bees?

References

[1] Crane, Eva (1983) The Archaeology of Beekeeping, Cornell University Press, ISBN 0-8014-1609-4

Appendix (optional)

Appendix One: Data Merging in MySQL: MySQL Query (https://github.com/chrisgmartin/HoneyIShrunkTheBees/blob/master/HoneyIShrunkTheBees.sql), Appendix 1 RMarkdown write-up (https://github.com/chrisgmartin/HoneyIShrunkTheBees/blob/master/Appendix/Appendix1-DataMySQL.Rmd), and RPubs write-up (http://rpubs.com/chrisgmartin/HISB_AppendixOne)

Appendix Two: Multiple Regression Analysis: Appendix 2 RMarkdown write-up (https://github.com/chrisgmartin/HoneyIShrunkTheBees/blob/master/Appendix/Appendix2-MultipleRegressionAnalysis.Rmd), table exported for use (https://github.com/chrisgmartin/HoneyIShrunkTheBees/blob/master/Appendix/HoneyC_FullModel.csv), and RPubs write-up (http://rpubs.com/chrisgmartin/HISB_AppendixTwo).