Data 412 HW 5

Ashley Totten

April 3, 2025

R Package Presentations

Cowsay, featuring the all knowing paper clip and Yoda!

```
library(cowsay)
library(fortunes)
say("fortune", #tells the program what to say
    by = "clippy" #specifies the alligator animal
)
```

```
 -----------------------------------------------------------
/ Running as administrator is like heroin... any problems \
| it solves it replaces with worse problems.  Jeff        |
| Newmiller about running R CMD check as administrator    |
\ R-help February 2013                                    /
 -----------------------------------------------------------
 \
  \

     _
    / \
    | |
    @ @
   || ||
   || ||
   |\_/|
   \___/ GB
```

```r
say("Do or do not, there is no try.",
    by="yoda"
    )
```

```
 --------------------------------
< Do or do not, there is no try. >
 --------------------------------
                \
                 \

                   ----
                _.' :  `._
            .-.'`.  ;   .'`.-.
   __      / : ___\ ;  /___ ; \      __
  ,'_ ""--.:__;".-.";: :".-.":__;.--"" _`,
  :' `.t""--.. '<@.`;_  ',@>` ..--""j.' `;
       `:-.._J '-.-'L__ `-- ' L_..-;'
          "-.__ ;  .-"  "-.  : __.-"
             L ' /.------.\ ' J
             "-.   "--"   .-"
             __.l"-:_JL_;-";.__
         .-j/'.;  ;""""  / .'\"-.
         .' /:`. "-.:     .-" .';  `.
      .-"  / ;  "-. "-..-" .-"  :    "-.
  .+"-.  : :      "-.__.-"       ;-._   \
  ; \  `.; ;                    : : "+. ;
  :  ;   ; ;                    : ;  : \:
  ;  :   ; :                    ;:   ;  :
  : \  ;  :  ;                  : ;  /  ::
  ;  ; :   ; :                  ;   :   ;:
  :  :  ;  :  ;                : :  ;  : ;
  ;\    :   ; :                ; ;     ; ;
  : `."-;   :  ;              :  ;    /  ;
 ;    -:   ; :              ;  : .-"   :
  :\     \  :  ;            : \.-"      :
  ;`.    \  ; :            ;.'_..--  / ;
  :  "-.  "-:  ;          :/."      .'  :
   \         \ :          ;/  __        :
    \       .-`.\        /t-""  ":-+.   :
     `.  .-"    `l    __/ /`. :  ; ; \  ;
       \   .-" .-"-.-"  .' .'j \  /   ;/
```

```
      \ / .-"   /.      .'.' ;_:'    ;
:-""-.`./-.'     /    `.___.'
             \ `t  ._  /  bug
              "-.t-._:'
```

I picked this presentation because it seemed like a fun package, and it was. The presentation did a good job of conveying the necessary material to use the package. It accomplished everything I was expecting it to. I explored different features of the package including the different animals and shapes it could draw. The code in the slides about listing all of the animals it could draw didn't work, so I imputted 'dog' as a test. It turns out there is no 'dog' animal in this package, but the error message provided a list of all of the available animals, which is how I found clippy and yoda along with many other funny options like grumpycat and wired cow. There are also options you can put in the "what" argument to provide cool results, like fortune which lists a fortune and catfacts which lists a random cat fact. This is a great package.

Plotthis

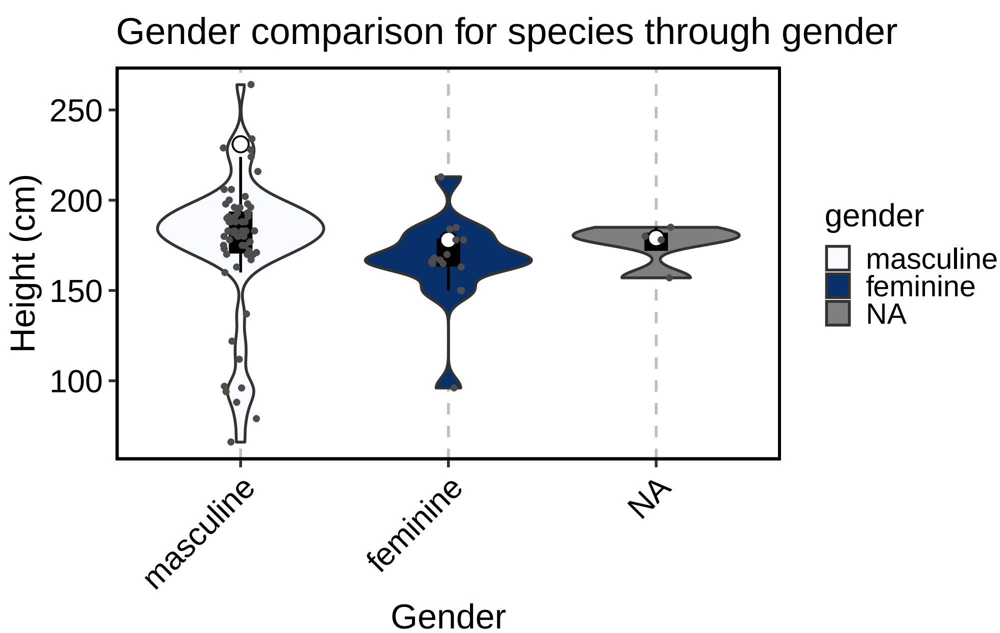
```r
library(plotthis)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

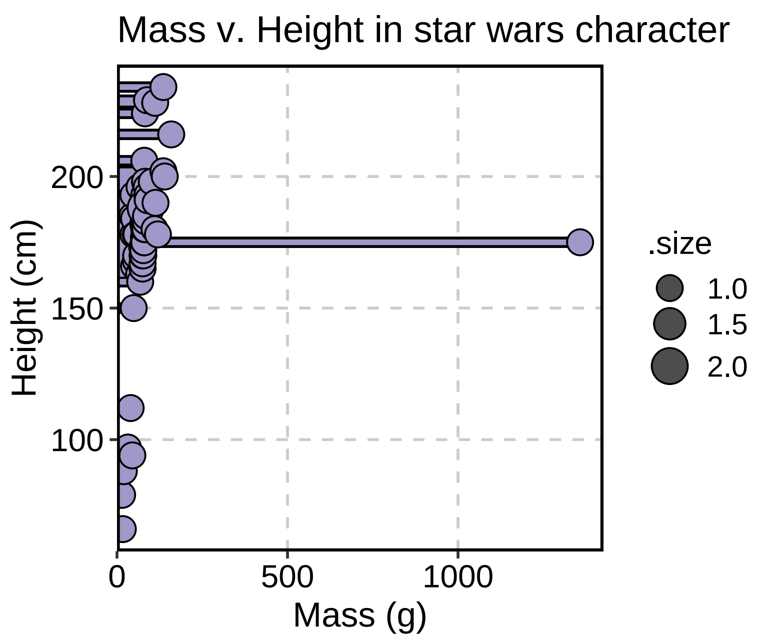
```r
data(starwars)

ViolinPlot(
  starwars, # Data
  x = "gender", # X Variable
  y = "height", # Y Variable
  group_by = "species", # Creates Groups for the column
  fill_mode = "x", # Fills violin plot s
```

```
  add_box = T, # Adds a box plot
  add_point = T, # Adds variable points
  palette = "Blues", # Selects palette
  title = "Gender comparison for species through gender", # Adds title
  xlab = "Gender", # Changes X label
  ylab = "Height (cm)", # Changes Y label
  )
```

```
Warning: Removed 6 rows containing non-finite outside the scale range
(`new_stat_ydensity()`).

Warning: Removed 6 rows containing non-finite outside the scale range
(`stat_boxplot()`).

Warning: Removed 40 rows containing non-finite outside the scale range
(`stat_summary()`).

Warning: Removed 6 rows containing missing values or values outside the scale range
(`geom_point()`).
```



```r
library(dplyr)
starwars %>%
  filter(mass > 0, height > 0)%>%
  LollipopPlot(
    x = "mass", # X Variable
    y = "height", # Y Variable
    palette = "Purples", # Selects palette
    title = "Mass v. Height in star wars character", # Adds title
    xlab = "Mass (g)", # Changes X label
    ylab = "Height (cm)", # Changes Y label
    )
```



I chose this package because it seemed like it had a lot to explore. After spending some time with it, the package has almost too much. I tried to explore the different plots, but I have not heard of most of these plots and the documentation is not clear on how to code with the different functions. The presentation gave a basic overview, but didn't really go into too much detail, which is understandable given how large the package is. I would not use this instead of ggplot normally, maybe if I wanted to make a graph very visually pleasing. Ultimately, it didn't end up accomplishing everything I had wanted, if I have some more time I'll go back and dive deeper into this package.

Global Patterns of Language Diversity

```
Global_Lang <- read_csv("/Users/tottena17/Downloads/AU 24-25/Data-412 R/Global_Patterns_of_La
```

```
Rows: 444 Columns: 4
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (3): Continent, Country, Measurement
dbl (1): Value

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Global_Lang %>%
  pivot_wider(
    id_cols = starts_with("C"),
    names_from = Measurement,
    values_from = Value
  ) -> lang
```

Model

```
lang %>%
  filter(Std < 2) %>%
  filter(Country !='Benin', Country != 'Burkina Faso',Country !='Cameroon', Country != 'Cote

lm(log(Langs) ~ MGS + log(Area), data = lang_filter) -> lm1
summary(lm1)
```

```
Call:
lm(formula = log(Langs) ~ MGS + log(Area), data = lang_filter)

Residuals:
    Min      1Q  Median      3Q     Max
-3.1097 -0.4778 -0.1023  0.7016  2.0293

Coefficients:
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.32010    1.96596  -2.706 0.009958 **
MGS          0.23305    0.04771   4.885 1.71e-05 ***
log(Area)    0.57604    0.14218   4.051 0.000228 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.047 on 40 degrees of freedom
Multiple R-squared:  0.4295,    Adjusted R-squared:  0.4009
F-statistic: 15.06 on 2 and 40 DF,  p-value: 1.335e-05
```

```r
nrow(lang_filter)
```

```
[1] 43
```

```r
#the adjusted R^2 was .4009
sqrt(.4009)
```

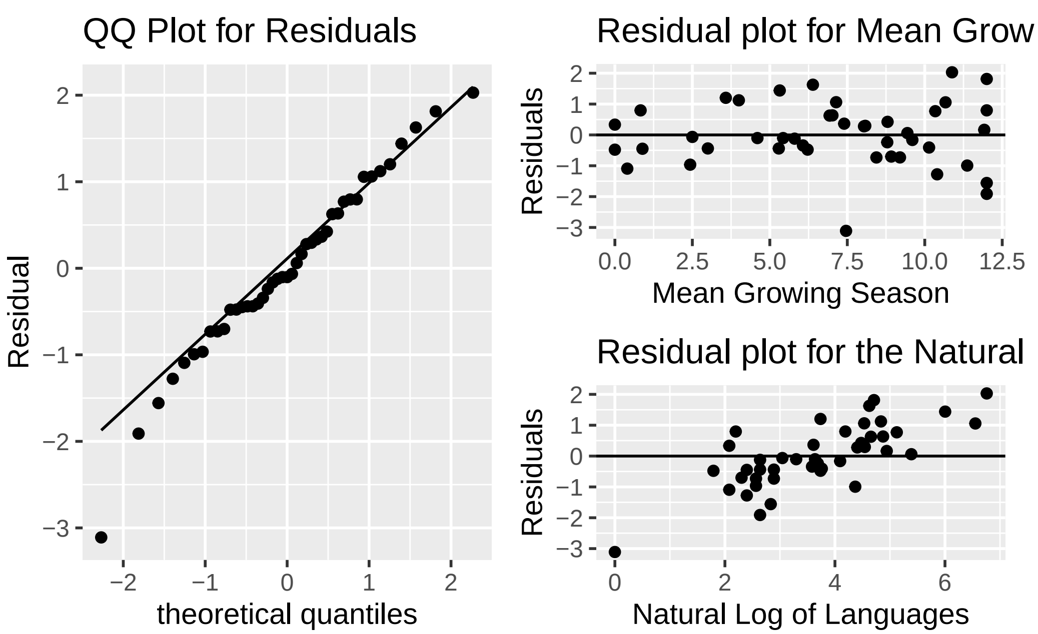
```
[1] 0.6331666
```

Assuming all other variables hold constant and controlling for area, for 1 unit increase in the log of languages, the average increase in mean growing season is 0.23305. This is statistically significant with a p-value of <0.01. The adjusted R-squared is .4009, which means this model explains 40.09% of the relationship between the log of languages and the mean growing season. There are 40 degrees of freedom in the model. An R of .6332 shows that there is a moderately strong correlation between mean growing season and the log of the languages.

Assumptions

```r
library(patchwork)
library(ggplot2)
res_lang <- residuals(lm1)

p1 <- tibble(lang_filter, res_lang) %>%
  ggplot(aes(x = MGS, y = res_lang))+
  geom_point()+
  geom_hline(yintercept = 0)+
  labs(x = "Mean Growing Season", y = "Residuals", title = "Residual plot for Mean Growing Se
```

```
p2 <- tibble(lang_filter, res_lang) %>%
  ggplot(aes(x = log(Langs), y = res_lang))+
  geom_point()+
  geom_hline(yintercept = 0)+
  labs(x = "Natural Log of Languages", y = "Residuals", title = "Residual plot for the Natura

p3 <- tibble(lang_filter, res_lang) %>%
  ggplot(aes(sample = res_lang))+
  stat_qq()+
  stat_qq_line()+
  labs(x = "theoretical quantiles", y = "Residual", title = "QQ Plot for Residuals")

p3 + p1 / p2
```



The QQ plot shows a roughly straight line with one outlier, so the assumption of normality can be fulfilled. The residual plot for mean growing season a mostly random pattern, so the assumption of assciation is fulfilled. However, the residual plot for Natural Log of languages shows a linear association. This means that the assumption of association between the natural log of languages and mean growing season cannot be fulfilled. There may be other variables that contribute to the relationship between mean growing season and the natural log of langauges that are not included in the model. Another possibility is that the data points may not follow a linear relationship, given the graph of the model and the R^2 statistic, the more likely situation is that there are other variables that contribute to the relationship between natural log of languages and growing season that are not included in the model.

EDA Workflow and Flint data

Basic info

```
library(tidyverse)
Flint <- read_csv("/Users/tottena17/Downloads/AU 24-25/Data-412 R/Flint_Facilities_Testing_20
```

```
Rows: 264 Columns: 11
-- Column specification ----------------------------------------------------------
Delimiter: ","
chr  (7): Sample Number, Lead, Sample Description, Copper, Street Name, City...
dbl  (3): Result ppb...3, Result ppb...6, Zip Code
dttm (1): SUBDATE

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#number of rows and columns and column names
nrow(Flint)
```

```
[1] 264
```

```
ncol(Flint)
```

```
[1] 11
```

```
colnames(Flint)
```

```
 [1] "Sample Number"      "Lead"               "Result ppb...3"
 [4] "Sample Description" "Copper"             "Result ppb...6"
 [7] "SUBDATE"            "Street Name"        "City"
[10] "Zip Code"           "Facility Name"
```

```r
#checking for duplicates
Flint %>%
  filter(duplicated(Flint))
```

```
# A tibble: 0 x 11
# i 11 variables: Sample Number <chr>, Lead <chr>, Result ppb...3 <dbl>,
#   Sample Description <chr>, Copper <chr>, Result ppb...6 <dbl>,
#   SUBDATE <dttm>, Street Name <chr>, City <chr>, Zip Code <dbl>,
#   Facility Name <chr>
```

```r
#checking for missing values. All False means there are no missing values
Flint %>%
  summarize_all(~ any(is.na(.)))
```

```
# A tibble: 1 x 11
  `Sample Number` Lead  `Result ppb...3` `Sample Description` Copper
  <lgl>           <lgl> <lgl>            <lgl>                <lgl>
1 FALSE           FALSE FALSE            FALSE                FALSE
# i 6 more variables: `Result ppb...6` <lgl>, SUBDATE <lgl>,
#   `Street Name` <lgl>, City <lgl>, `Zip Code` <lgl>, `Facility Name` <lgl>
```

Cleaning observations

```r
#checking for unique values in facility name and sample description
Flint %>%
  distinct(`Facility Name`)
```

```
# A tibble: 33 x 1
   `Facility Name`
   <chr>
 1 ALLEREE BILLINGS
 2 ANGIE MCNEAL
 3 BETTY JOE PEA
 4 BRIDON`S CDC
 5 CATHEDRAL OF FAITH HEAD START
 6 CUMMINGS/ GREAT EXPECTATIONS
 7 CUMMINGS/GREAT EXPECTATIONS
 8 GAIL SEWELL
```

```
 9 GENESEE COUNTY JOB CORPS
10 GLORIA`S LITTLE ANGELS
# i 23 more rows
```

```
Flint %>%
  distinct(`Sample Description`)
```

```
# A tibble: 239 x 1
   `Sample Description`
   <chr>
 1 01KC003 KITCHEN
 2 01KC001 KITCHEN
 3 01BF002 BATHROOM
 4 001KC004 KITCHEN
 5 001BF002 1ST FLOOR BATH
 6 001BF001 1ST FLOOR BATH
 7 LLBF001 BOY`S RESTROOM
 8 LLKC004 DAY CARE KITCHEN
 9 LLBF002 GIRLS RESTROOM
10 LLCF005 DAY CARE CLASSROOM
# i 229 more rows
```

```
Flint %>% #replacing facility names that were duplicates
  mutate(`Facility Name` = str_replace(`Facility Name`, "CUMMINGS/ GREAT EXPECTATIONS", "CUMM
  mutate(`Facility Name` = str_replace(`Facility Name`, "CUMMING/ GREAT EXPECTATIONS", "CUMMI
  mutate(`Facility Name` = str_replace(`Facility Name`, "TEDDY BEARS/ PATRICE MOORE", "TEDDY
  #replacing sample descriptions with simplier descriptions
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "KC"), "Kitchen sinl
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "BF"), "Restroom sin
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "NS"), "Nurse's stat
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "WC"), "Water coole
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "DW"), "Water founta
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "CF"), "Classroom s
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "SP"), "Chapel sink'
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "CK"), "Kitchen sinl
  mutate(`Sample Description` = if_else(str_detect(`Sample Description`, "KS"), "Kitchen sinl

Flint1 %>%
  distinct(`Facility Name`)
```

```
# A tibble: 30 x 1
```

```
   `Facility Name`
   <chr>
 1 ALLEREE BILLINGS
 2 ANGIE MCNEAL
 3 BETTY JOE PEA
 4 BRIDON`S CDC
 5 CATHEDRAL OF FAITH HEAD START
 6 CUMMINGS/GREAT EXPECTATIONS
 7 GAIL SEWELL
 8 GENESEE COUNTY JOB CORPS
 9 GLORIA`S LITTLE ANGELS
10 HEAVENLY ANGELS (LATISH SMITH)
# i 20 more rows
```

```
Flint1 %>%
  distinct(`Sample Description`)
```

```
# A tibble: 6 x 1
  `Sample Description`
  <chr>
1 Kitchen sink
2 Restroom sink
3 Classroom sink
4 Chapel sink
5 Water cooler
6 Water fountain
```

Cleaning Columns and Data Set Structure

```
colnames(Flint1)
```

```
 [1] "Sample Number"      "Lead"                "Result ppb...3"
 [4] "Sample Description" "Copper"              "Result ppb...6"
 [7] "SUBDATE"            "Street Name"         "City"
[10] "Zip Code"           "Facility Name"
```

```
#View(Flint1)

Flint1 %>%  #removing unnecessary columns, all of the samples are from Flint, MI so that inf
```

```
  mutate(City = NULL) %>%
  #pivoting the data set so Lead and Copper have their own columns
  pivot_wider(
    names_from = "Lead",
    values_from = "Result ppb...3"
  ) %>%
  pivot_wider(
    names_from = "Copper",
    values_from = "Result ppb...6"
  ) %>% #standardizing the column names
  rename_all(~ str_replace_all(., " ", "_")) %>%
  rename(Date = SUBDATE) %>%
  rename(Lead_ppb = Lead_250_mL_Sample) %>%
  rename(Copper_ppb = Copper_250_mL_Sample)-> Flint2

Flint2
```

```
# A tibble: 264 x 8
   Sample_Number Sample_Description Date                Street_Name   Zip_Code
   <chr>         <chr>              <dttm>              <chr>            <dbl>
 1 LH59064       Kitchen sink       2017-12-21 14:05:00 DAMON ST         48505
 2 LH55198       Kitchen sink       2017-11-29 11:11:07 BALDWIN BLVD     48505
 3 LH55197       Restroom sink      2017-11-29 11:11:06 BALDWIN BLVD     48505
 4 LH58462       Kitchen sink       2017-12-19 14:05:42 CLEMENT ST       48504
 5 LH58463       Restroom sink      2017-12-19 14:05:43 CLEMENT ST       48504
 6 LH58464       Restroom sink      2017-12-19 14:05:44 CLEMENT ST       48504
 7 LH55194       Restroom sink      2017-11-29 11:11:00 KEARSLEY         48503
 8 LH55196       Kitchen sink       2017-11-29 11:11:01 EAST KEARSLEY    48503
 9 LH55193       Restroom sink      2017-11-29 11:10:59 EAST KEARSLEY    48503
10 LH55195       Classroom sink     2017-11-29 11:11:01 EAST KEARSLEY    48503
# i 254 more rows
# i 3 more variables: Facility_Name <chr>, Lead_ppb <dbl>, Copper_ppb <dbl>
```

Cleaning variable types

```
Flint2 %>%
  mutate(Date = as.Date(Date)) %>%
  mutate(Sample_Description = as.factor(Sample_Description)) %>%
  mutate(Facility_Name = as.factor(Facility_Name)) %>%
  mutate(Zip_Code = as.factor(Zip_Code))->Flint3
```

```r
Flint3
```

```
# A tibble: 264 x 8
   Sample_Number Sample_Description Date       Street_Name   Zip_Code
   <chr>         <fct>              <date>     <chr>         <fct>
 1 LH59064       Kitchen sink       2017-12-21 DAMON ST      48505
 2 LH55198       Kitchen sink       2017-11-29 BALDWIN BLVD  48505
 3 LH55197       Restroom sink      2017-11-29 BALDWIN BLVD  48505
 4 LH58462       Kitchen sink       2017-12-19 CLEMENT ST    48504
 5 LH58463       Restroom sink      2017-12-19 CLEMENT ST    48504
 6 LH58464       Restroom sink      2017-12-19 CLEMENT ST    48504
 7 LH55194       Restroom sink      2017-11-29 KEARSLEY      48503
 8 LH55196       Kitchen sink       2017-11-29 EAST KEARSLEY 48503
 9 LH55193       Restroom sink      2017-11-29 EAST KEARSLEY 48503
10 LH55195       Classroom sink     2017-11-29 EAST KEARSLEY 48503
# i 254 more rows
# i 3 more variables: Facility_Name <fct>, Lead_ppb <dbl>, Copper_ppb <dbl>
```

I changed the type for some of the varaibles. Date will be in date format. Zip code, facility name, and sample description will be factors so they be more easily used for categorizing data.

Statistical and graphical summaries

```r
#Lead
mean(Flint3$Lead_ppb)
```

```
[1] 4.102273
```

```r
sd(Flint3$Lead_ppb)
```

```
[1] 27.07813
```

```r
median(Flint3$Lead_ppb)
```

```
[1] 0
```

```r
IQR(Flint3$Lead_ppb)
```

```
[1] 1
```

```r
#Copper
mean(Flint3$Copper_ppb)
```

```
[1] 87.5
```

```r
sd(Flint3$Copper_ppb)
```

```
[1] 367.5091
```

```r
median(Flint3$Copper_ppb)
```

```
[1] 0
```

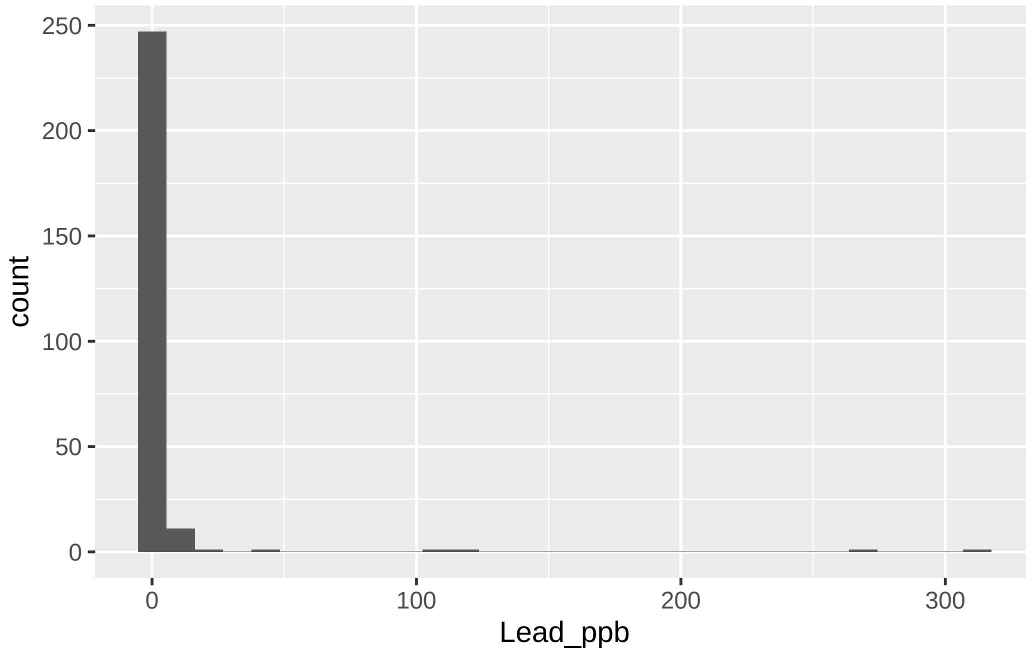
```r
IQR(Flint3$Copper_ppb)
```

```
[1] 80
```

For both lead and copper, the mean is greater than the median and there is a lot of variability in the data set.

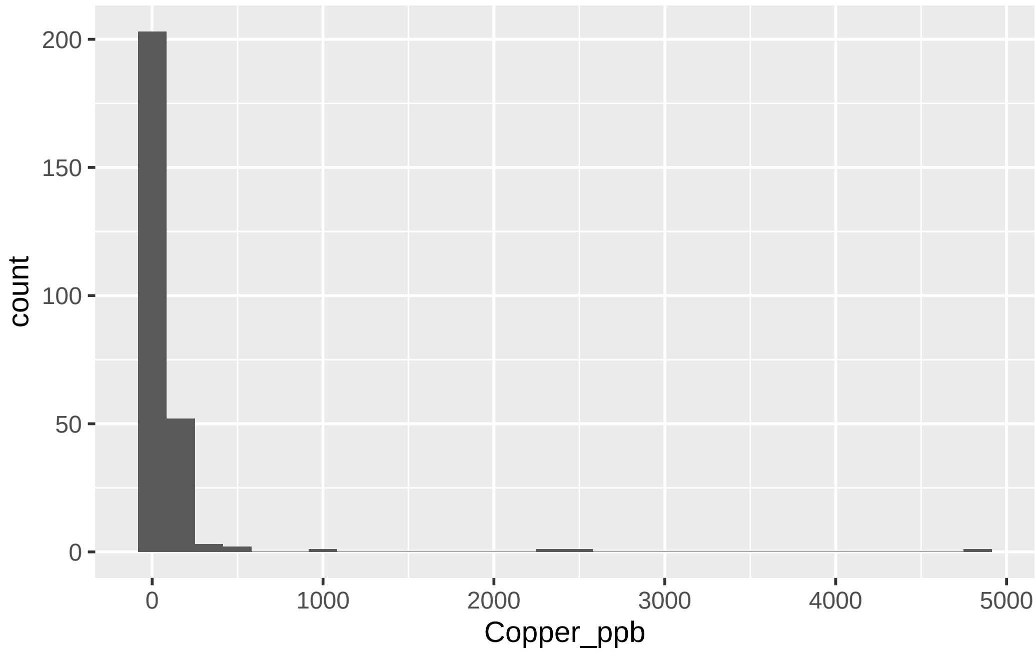
```r
library(ggplot2)
ggplot(aes(x = Lead_ppb), data = Flint3)+
  geom_histogram()
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(aes(x = Copper_ppb), data = Flint3)+
  geom_histogram()
```

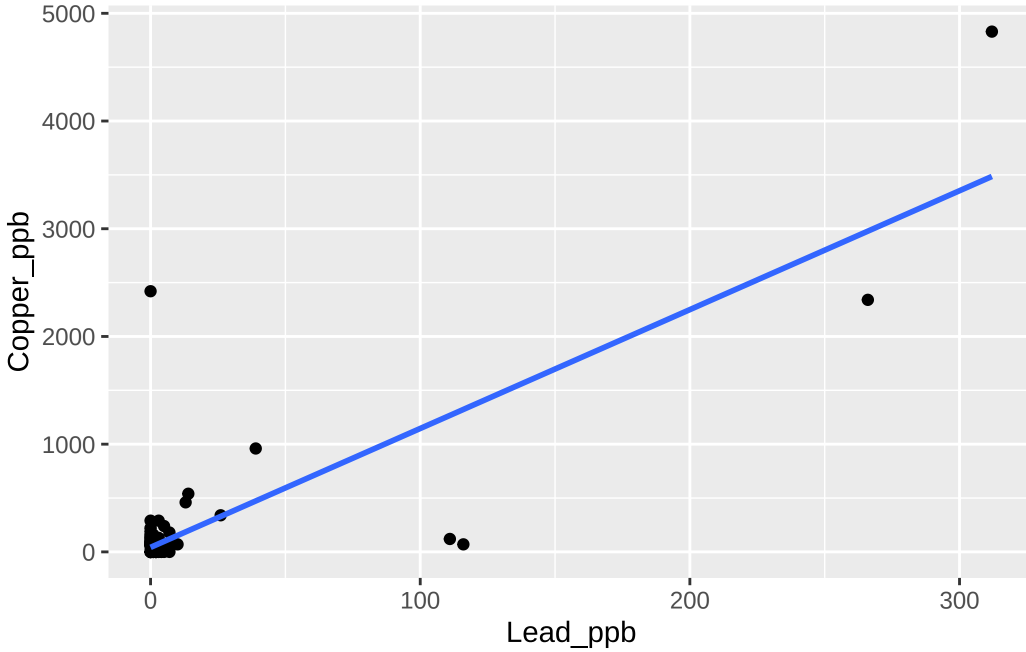
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The graphs of lead levels and copper levels are heavily skewed right.

```
ggplot(aes(x=Lead_ppb, y=Copper_ppb), data = Flint3)+
  geom_point()+
  geom_smooth(method = lm, se = FALSE)
```

```
`geom_smooth()` using formula = 'y ~ x'
```




```r
lm1 <- lm(Copper_ppb ~ Lead_ppb, data = Flint3)
summary(lm1)
```

```
Call:
lm(formula = Copper_ppb ~ Lead_ppb, data = Flint3)

Residuals:
     Min       1Q   Median       3Q      Max
-1252.34   -42.23   -42.23    27.77  2377.77

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  42.2298    13.3434   3.165  0.00173 **
Lead_ppb     11.0354     0.4881  22.608  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 214.3 on 262 degrees of freedom
Multiple R-squared:  0.6611,    Adjusted R-squared:  0.6598
F-statistic: 511.1 on 1 and 262 DF,  p-value: < 2.2e-16
```

Based on the graph, there is not a strong linear correlation between the lead levels in Flint's pipes and copper levels in Flint's pipes. The regression summary shows a strong linear correlation. This is probably because most of the observations from the lead and copper pipes were 0.

```
Flint3 %>%
  filter(Lead_ppb > 0 | Copper_ppb > 0) %>%
  count(Facility_Name)->Flint4
```

There are 21 facilities that had at least 1 sample that contained lead of copper in their water.