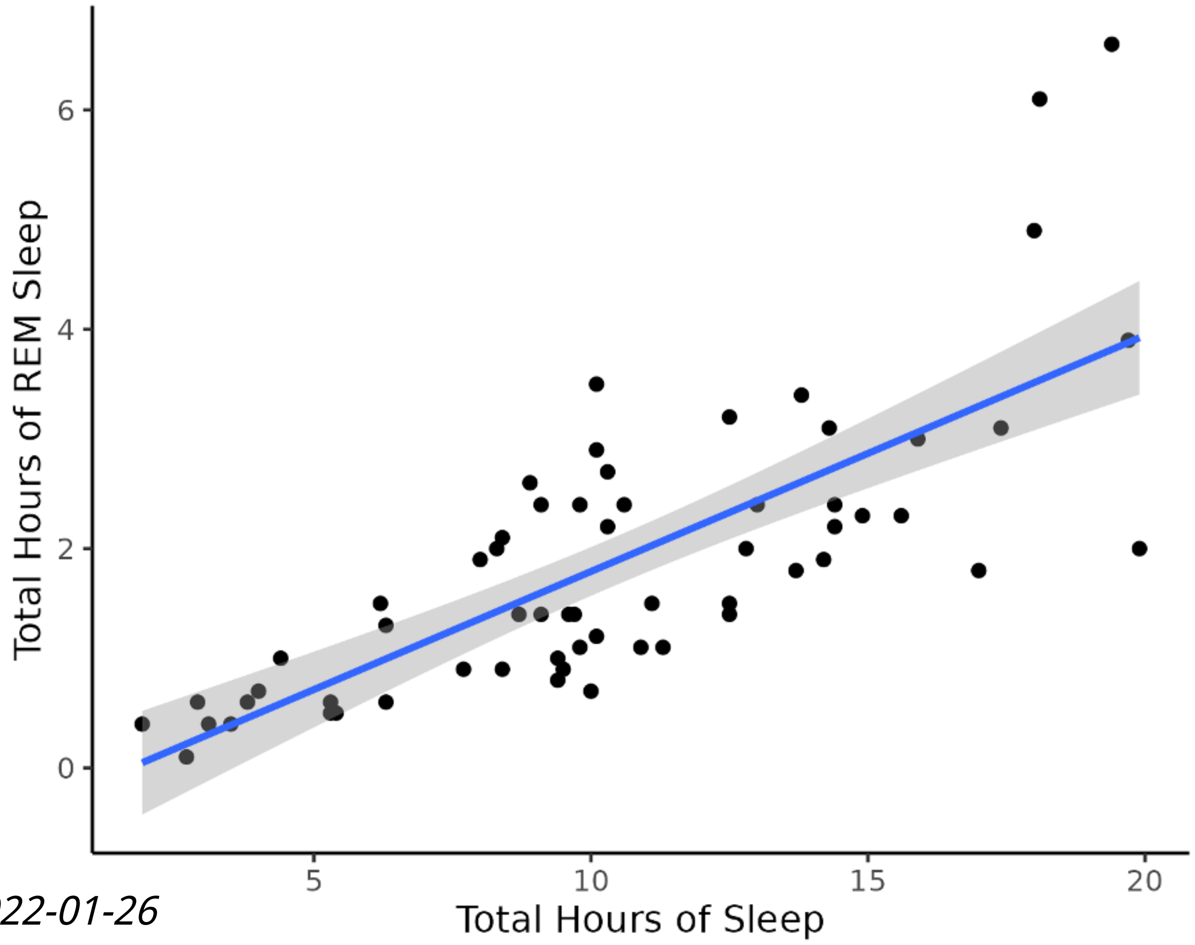
Workshop: Dealing with Data in R

Statistics in R

Now that your data are ready



Steffi LaZerte https://steffilazerte.ca | *Compiled: 2022-01-26*

Basic Statistics

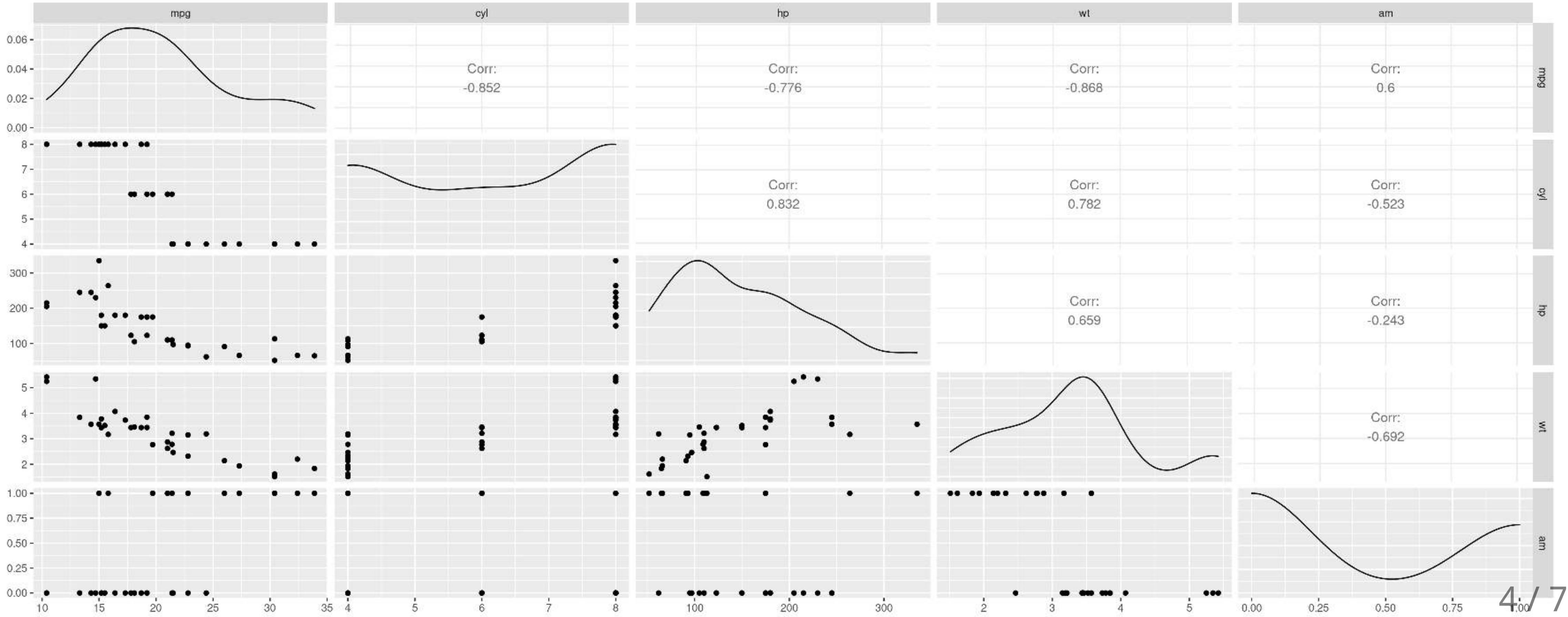
Looking at your data

```r
library(skimr)
skim(mtcars)
```

```
## ── Data Summary ────────────────────────
##                            Values
## Name                       mtcars
## Number of rows             32
## Number of columns          11
## _______________________
## Column type frequency:
##   numeric                  11
## ________________________
## Group variables            None
##
## ── Variable type: numeric ──────────────────────────────────────────────────────────────────────
##   skim_variable n_missing complete_rate    mean      sd     p0     p25     p50     p75    p100 hist
##  1 mpg                  0             1  20.1      6.03   10.4    15.4    19.2    22.8    33.9  ▃▇▅▁▂
##  2 cyl                  0             1   6.19     1.79    4       4       6       8       8    ▆▁▃▁▇
##  3 disp                 0             1 231.     124.     71.1   121.    196.    326     472    ▇▃▃▃▂
##  4 hp                   0             1 147.      68.6    52      96.5   123     180     335    ▇▇▆▃▁
##  5 drat                 0             1   3.60     0.535   2.76    3.08    3.70    3.92    4.93 ▇▃▇▅▁
##  6 wt                   0             1   3.22     0.978   1.51    2.58    3.32    3.61    5.42 ▃▃▇▁▂
```

Looking at your data

```
library(GGally)
ggpairs(dplyr::select(mtcars, mpg, cyl, hp, wt, am))
```



T-Tests

Comparing two samples

```
t.test(values ~ group, data = data)
```

- **`values`** are measurements from the two populations
- **`group`** is the column that differentiates the two groups

OR

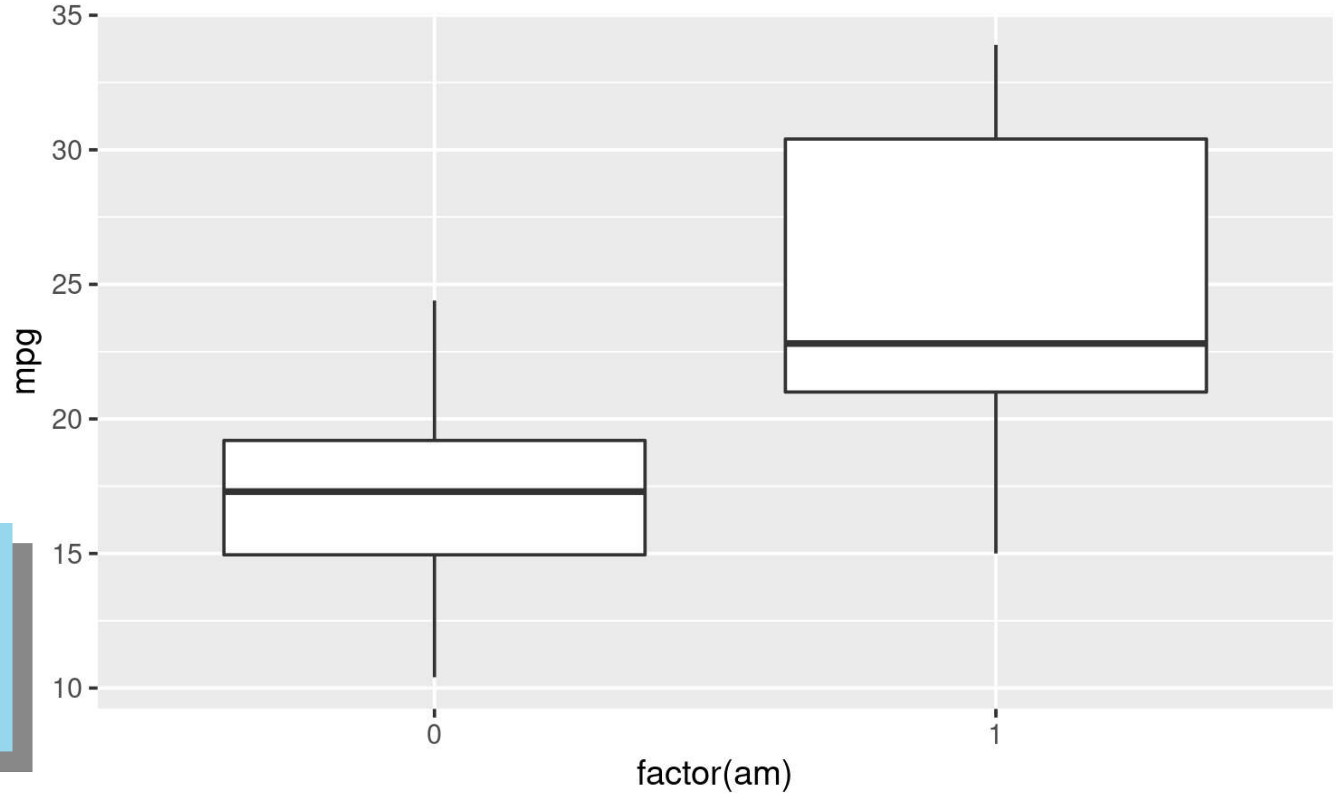
```
t.test(sample1, sample2)
```

- **`sample1`** and **`sample2`** are the two samples to be compared

T-Tests

Miles-per-gallon significantly different between Automatic and Manual cars?

```
ggplot(mtcars, aes(x = factor(am), y = mpg)) +
  geom_boxplot()
```



?mtcars shows
0 = automatic
1 = manual

T-Tests

Miles-per-gallon significantly different between Automatic and Manual cars?

```
t.test(mpg ~ am, data = mtcars)
```

```
##
##      Welch Two Sample t-test
##
## data:  mpg by am
## t = -3.7671, df = 18.332, p-value = 0.001374
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -11.280194  -3.209684
## sample estimates:
## mean in group 0 mean in group 1
##        17.14737        24.39231
```

P = 0.00137, so yes!
Manual cars (1) get more miles per gallon than Automatic cars (0)

Other tests

- Fisher's Exact Test - **`fisher.test()`**
- Chi-Square Test - **`chisq.test()`**

> Here it's mostly about getting your data into a matrix

Getting data into matrix for Chi-Square

Example Data

```
my_data <- data.frame(expected = c(10, 10),
                      observed = c(16, 4),
                      site = c("A", "B"))
my_data
```

```
##   expected observed site
## 1       10       16    A
## 2       10        4    B
```

As a matrix (only expected and observed)

```
my_matrix <- dplyr::select(my_data, expected, observed)
my_matrix <- as.matrix(my_matrix)
my_matrix
```

```
##      expected observed
## [1,]       10       16
## [2,]       10        4
```

Getting data into matrix for Chi-Square

Example Data

```
my_data <- data.frame(expected = c(10, 10),
                      observed = c(16, 4),
                      site = c("A", "B"))
my_data
```

```
##   expected observed site
## 1       10       16    A
## 2       10        4    B
```

As a matrix (only expected and observed)

```
my_matrix <- dplyr::select(my_data, expected, observed)
my_matrix <- as.matrix(my_matrix)
my_matrix
```

```
##      expected observed
## [1,]       10       16
## [2,]       10        4
```

Chi-Square Test

```
chisq.test(my_matrix)
```

```
##
##      Pearson's Chi-squared test with Yates'
continuity correction
##
## data:  my_matrix
## X-squared = 2.7473, df = 1, p-value = 0.09742
```

Non-parametric Statistics

Non-parametric Statistics

Wilcoxon Rank Sum (Mann-Whitney) Test

```
air <- filter(airquality, Month %in% c(5, 8))
```

Is there a difference in air quality between May (5th month) and August (8th month)?

```
wilcox.test(Ozone ~ Month, data = air, exact = FALSE)
```

```
##
##      Wilcoxon rank sum test with continuity correction
##
## data:  Ozone by Month
## W = 127.5, p-value = 0.0001208
## alternative hypothesis: true location shift is not equal to 0
```

Yes!

Non-parametric Statistics

Kruskal-Wallis Rank Sum Test

Is there a difference in air quality among months?

```
kruskal.test(Ozone ~ Month, data = airquality)
```

```
##
##      Kruskal-Wallis rank sum test
##
## data:  Ozone by Month
## Kruskal-Wallis chi-squared = 29.267, df = 4, p-value = 6.901e-06
```

Yes, there is at least one month that is different from the rest.

Linear Models

Linear Models

Running models in R

```
lm(y ~ x1 + x2, data = data)
```

- **`y`** is **dependent** variable
- **`x1`** and **`x2`** are **independent** variables

Linear Models

Running models in R

```
lm(y ~ x1 + x2, data = data)
```

- **y** is **dependent** variable
- **x1** and **x2** are **independent** variables

Different types of models

- If both **x**'s are continuous, this is a **linear regression**
- If both **x**'s are categorical, this is an **ANOVA**
- If **x1** is continuous and **x2** is categorical, this is an **ANCOVA**

Linear Models

Running models in R

```
lm(y ~ x1 + x2, data = data)
```

- **`y`** is **dependent** variable
- **`x1`** and **`x2`** are **independent** variables

Different types of models

- If both **`x`**'s are continuous, this is a **linear regression**
- If both **`x`**'s are categorical, this is an **ANOVA**
- If **`x1`** is continuous and **`x2`** is categorical, this is an **ANCOVA**

R will figure it out for you

Linear Models: Interactions

Main effects only

```
m <- lm(y ~ x1 + x2, data = data)
```

Linear Models: Interactions

Main effects only

```
m <- lm(y ~ x1 + x2, data = data)
```

Main effects and **interaction**

```
m <- lm(y ~ x1 + x2 + x1:x2, data = data)
```

Linear Models: Interactions

Main effects only

```
m <- lm(y ~ x1 + x2, data = data)
```

Main effects and **interaction**

```
m <- lm(y ~ x1 + x2 + x1:x2, data = data)
```

Main effects and **interaction**

```
m <- lm(y ~ x1 * x2, data = data)
```

Linear Models: Interactions

Main effects only

```
m <- lm(y ~ x1 + x2, data = data)
```

Main effects and interaction

```
m <- lm(y ~ x1 + x2 + x1:x2, data = data)
```

Main effects and interaction

```
m <- lm(y ~ x1 * x2, data = data)
```

> **`x1 * x2`** equivalent to **`x1 + x2 + x1:x2`**

Linear Regression

Example with msleep

```
lm(sleep_cycle ~ bodywt, data = msleep)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep)
##
## Coefficients:
## (Intercept)       bodywt
##     0.38549      0.00107
```

Linear Regression

Example with `msleep`

```
lm(sleep_cycle ~ bodywt, data = msleep)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep)
##
## Coefficients:
## (Intercept)
##     0.38549
```

Intercept

Linear Regression

Example with `msleep`

```
lm(sleep_cycle ~ bodywt, data = msleep)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep)
##
## Coefficients:
## (Intercept)       bodywt
##     0.38549      0.00107
```

Slope

Linear Regression

Example with `msleep`

```
lm(sleep_cycle ~ bodywt, data = msleep)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep)
##
## Coefficients:
## (Intercept)       bodywt
##     0.38549      0.00107
```

Hmm, not a lot of detail

Linear Regression

Assign model to **m**

```
m <- lm(sleep_cycle ~ bodywt, data = msleep)
```

m is a model object

```
class(m)
```

```
## [1] "lm"
```

This contains all the information about the model

Linear Regression

```
summary(m)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36081 -0.20228 -0.08506  0.03564  1.04817
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.3854937  0.0623726   6.180 8.43e-07 ***
## bodywt      0.0010700  0.0004248   2.519   0.0173 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3313 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.1746,    Adjusted R-squared:  0.147
## F-statistic: 6.344 on 1 and 30 DF,  p-value: 0.01734
```

Linear Regression

```r
summary(m)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36081 -0.20228 -0.08506  0.03564  1.04817
##
## Coefficients:
##              Estimate Std. Err
## (Intercept) 0.3854937  0.06237
## bodywt      0.0010700  0.00042
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3313 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.1746,	Adjusted R-squared:  0.147
## F-statistic: 6.344 on 1 and 30 DF,  p-value: 0.01734
```

Wait!
Shouldn't interpret until we know the model is solid

Model Diagnostics

Model Assumptions

- Normality (of residuals)
- Constant Variance (no heteroscedasticity)

Other cautions

- Influential variables (Cook's D)
- Multiple collinearity (with more than one **x** or explanatory variables)

Model Diagnostics

Diagnostics by Hand

- Depending on model, different diagnostic functions (e.g., **`plot()`**)
- But you can check any model by hand if you pull out the right data

First let's get our relevant variables, **`residuals`** and **`fitted values`**:

```
d <- data.frame(residuals = residuals(m),    # Residuals
                fitted = fitted(m),          # Fitted values
                cooks = cooks.distance(m))   # Cook's D

d <- mutate(d, observation = 1:nrow(d))      # Observation number
```

Model Diagnostics

Diagnostics by Hand

- Depending on model, different diagnostic functions (e.g., **`plot()`**)
- But you can check any model by hand if you pull out the right data

First let's get our relevant variables, **`residuals`** and **`fitted values`**:

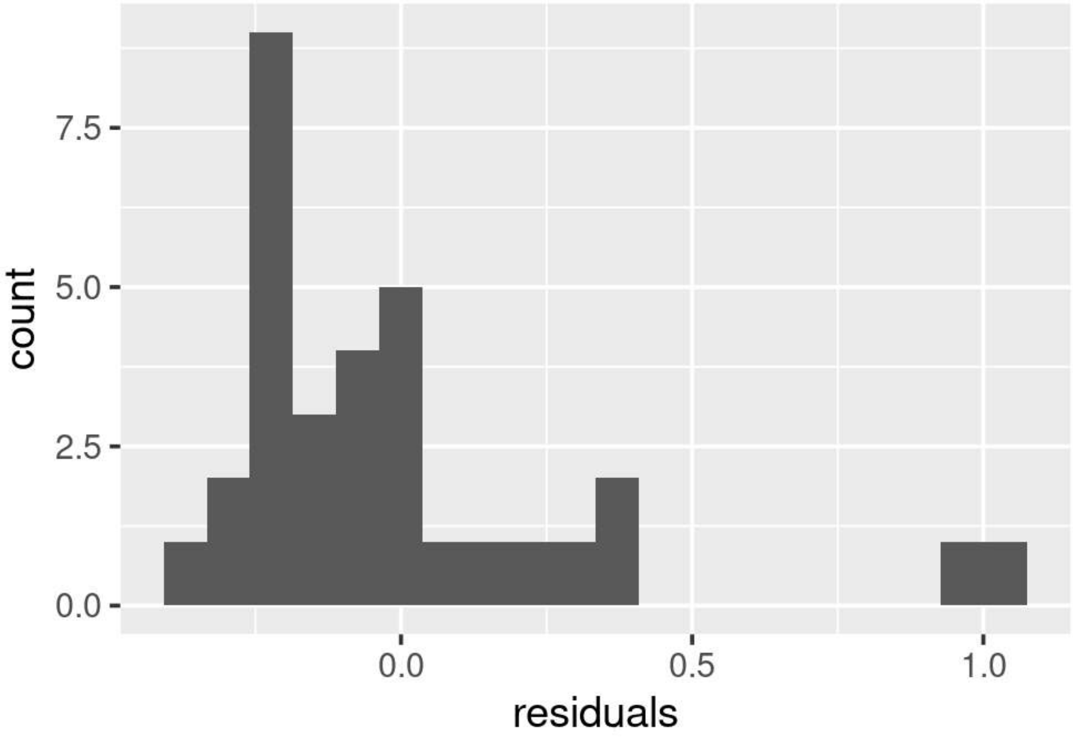
```
d <- data.frame(residuals = residuals(m),     # Residuals
                fitted = fitted(m),           # Fitted values
                cooks = cooks.distance(m))    # Cook's D

d <- mutate(d, observation = 1:nrow(d))       # Observation number
```

```
##     residuals    fitted        cooks observation
## 4 -0.25218072 0.3855141 1.104107e-02           1
## 5 -0.36081280 1.0274795 1.403343e+00           2
## 6  0.37705353 0.3896131 2.422542e-02           3
## 7 -0.02408421 0.4074175 9.247658e-05           4
## 9 -0.06714006 0.4004734 7.353601e-04           5
```

Normality

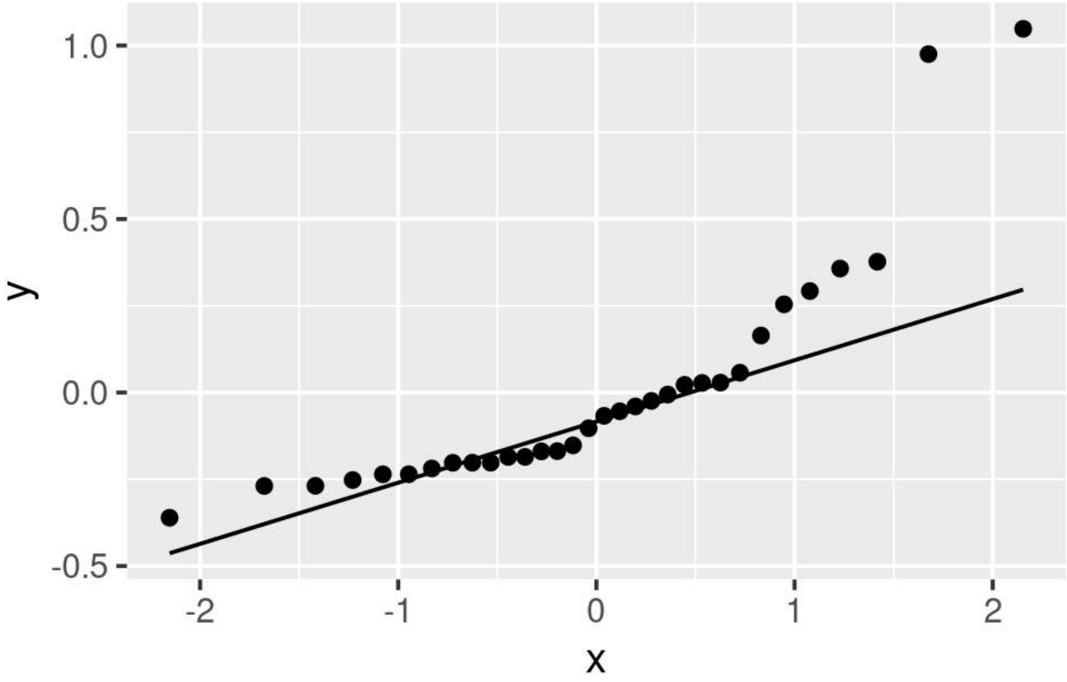
Histogram of residuals

```
ggplot(data = d, aes(x = residuals)) +
  geom_histogram(bins = 20)
```



QQ Normality plot of residuals

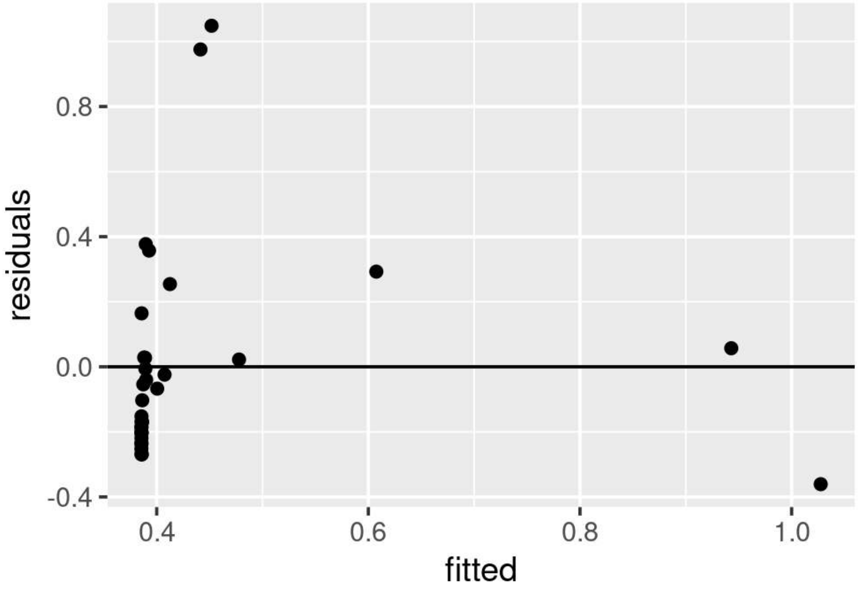
```
ggplot(data = d, aes(sample = residuals)) +
  stat_qq() +
  stat_qq_line()
```



Variance and Influence

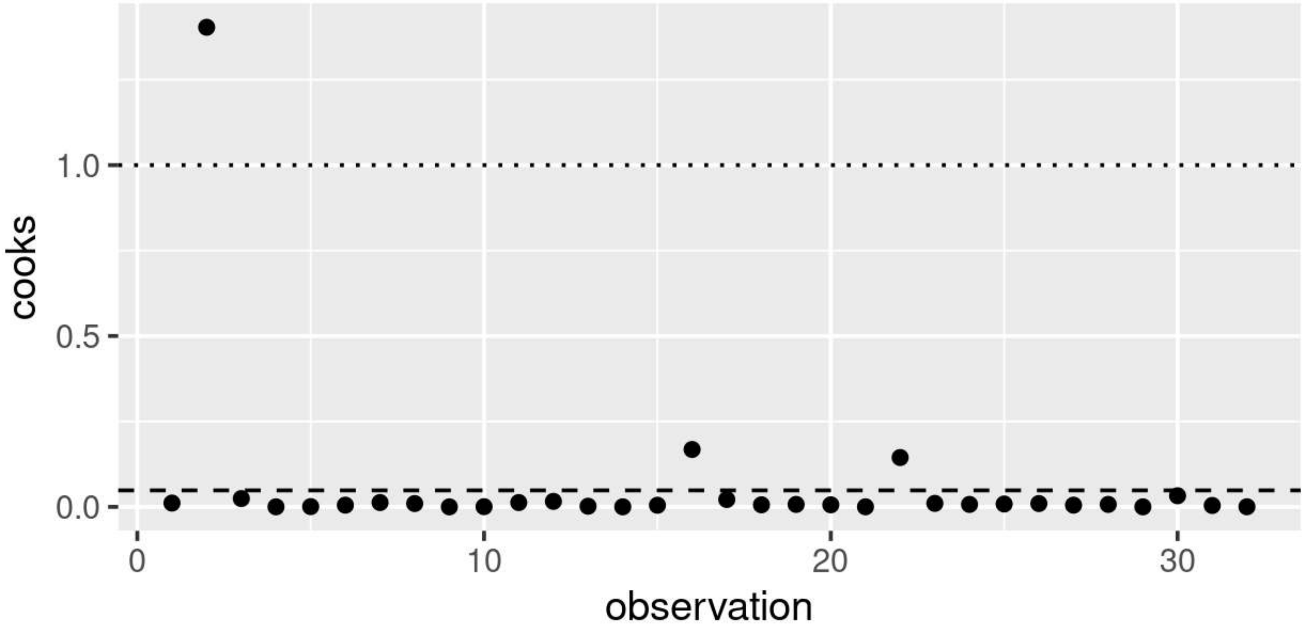
Check heteroscedasticity

```
ggplot(d, aes(x = fitted, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0)
```



Cook's D

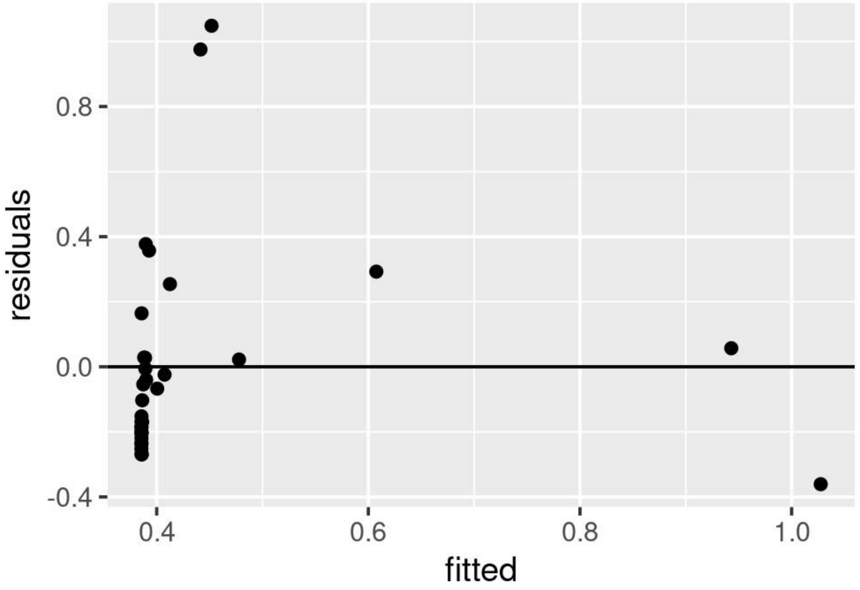
```
ggplot(d, aes(x = observation, y = cooks)) +
  geom_point() +
  geom_hline(yintercept = 1, linetype = "dotted") +
  geom_hline(yintercept = 4/nrow(msleep), linetype = "dashed")
```



Variance and Influence

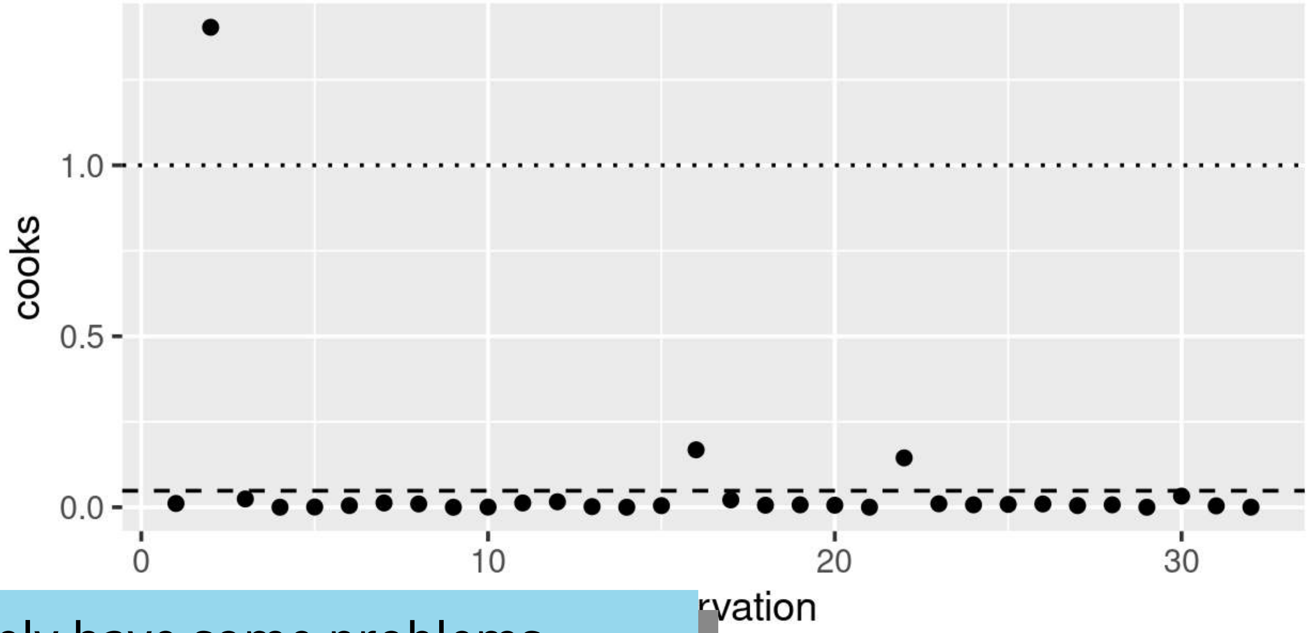
Check heteroscedasticity

```
ggplot(d, aes(x = fitted, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0)
```



Cook's D

```
ggplot(d, aes(x = observation, y = cooks)) +
  geom_point() +
  geom_hline(yintercept = 1, linetype = "dotted") +
  geom_hline(yintercept = 4/nrow(msleep), linetype = "dashed")
```

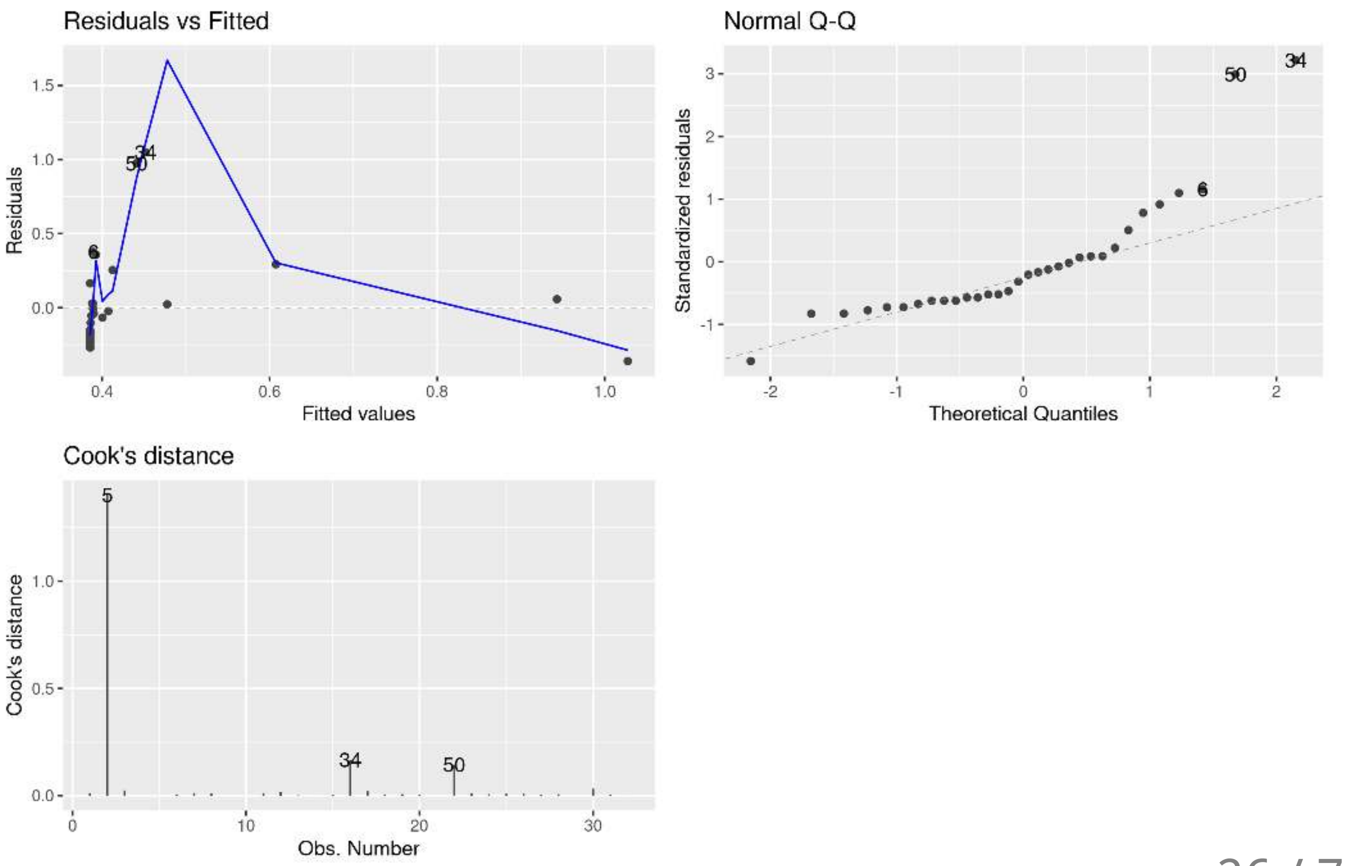


Definitely have some problems

Diagnostics with ggfortify

- Uses **autoplot**
- Choose **which** plots to show
  - 1 = Residuals vs. fitted
  - 2 = QQ Norm
  - 4 = Cook's Distance
  - Others available

```
library(ggfortify)
autoplot(m, which = c(1, 2, 4))
```



Transformations

Let's try a log transformation

- Normally you would only transform the `y` value
- But mass data often works best with log10 or ln transformations (isometry)
- So we'll transform both x and y here

```
msleep_log <- mutate(msleep,
                     sleep_cycle = log10(sleep_cycle),
                     bodywt = log10(bodywt),
                     brainwt = log10(brainwt))

m_log <- lm(sleep_cycle ~ bodywt, data = msleep_log)
```

Note:

- By default **`log()`** takes the ln. Use **`log10()`** if you want base 10

Multicollinearity (collinearity)

Only relevant with more than one explanatory variable

```r
m_mult <- lm(sleep_cycle ~ bodywt + brainwt, data = msleep_log)
```

Multicollinearity (collinearity)

Only relevant with more than one explanatory variable

```r
m_mult <- lm(sleep_cycle ~ bodywt + brainwt, data = msleep_log)
```

Use the `car` package to get the `vif()` function

```r
library(car)
vif(m_mult)
```

```
##   bodywt  brainwt 
## 13.30615 13.30615
```

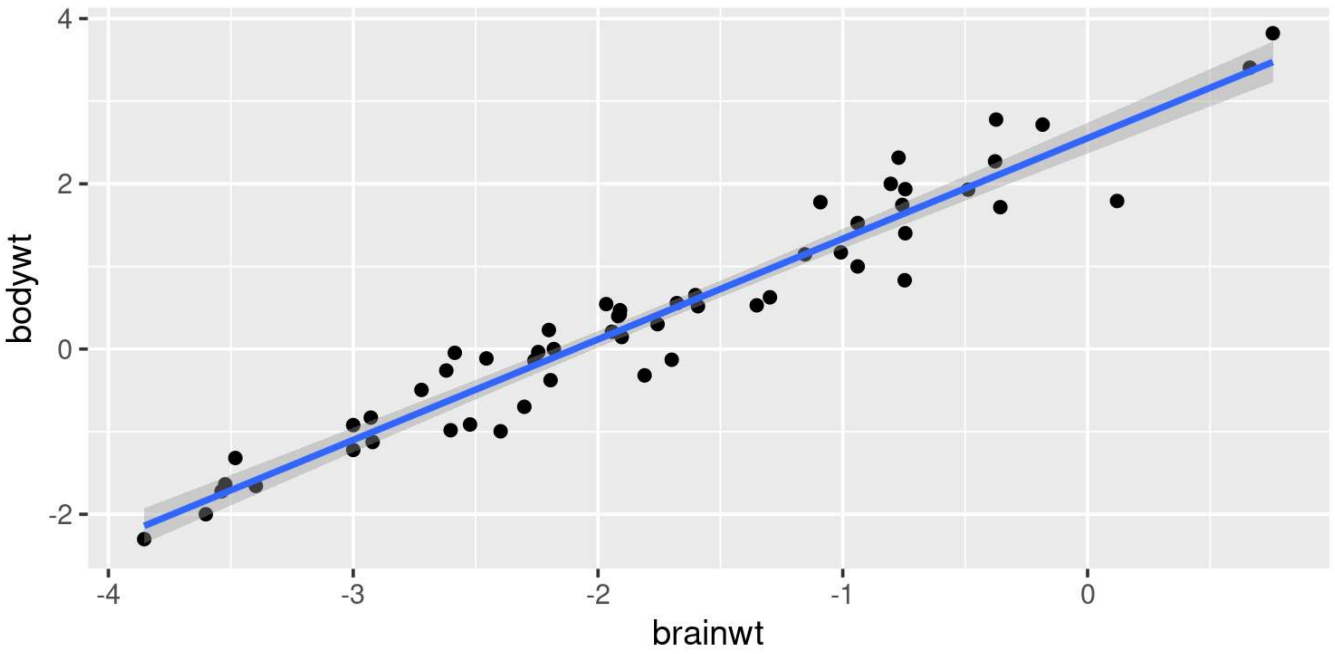
Hmm, that's pretty high (looking for < 10)

Multicollinearity (collinearity)

Look at our two explanatory variables:

```
ggplot(data = msleep_log, aes(x = brainwt, y = bodywt)) +
  geom_point() +
  stat_smooth(method = "lm")
```

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

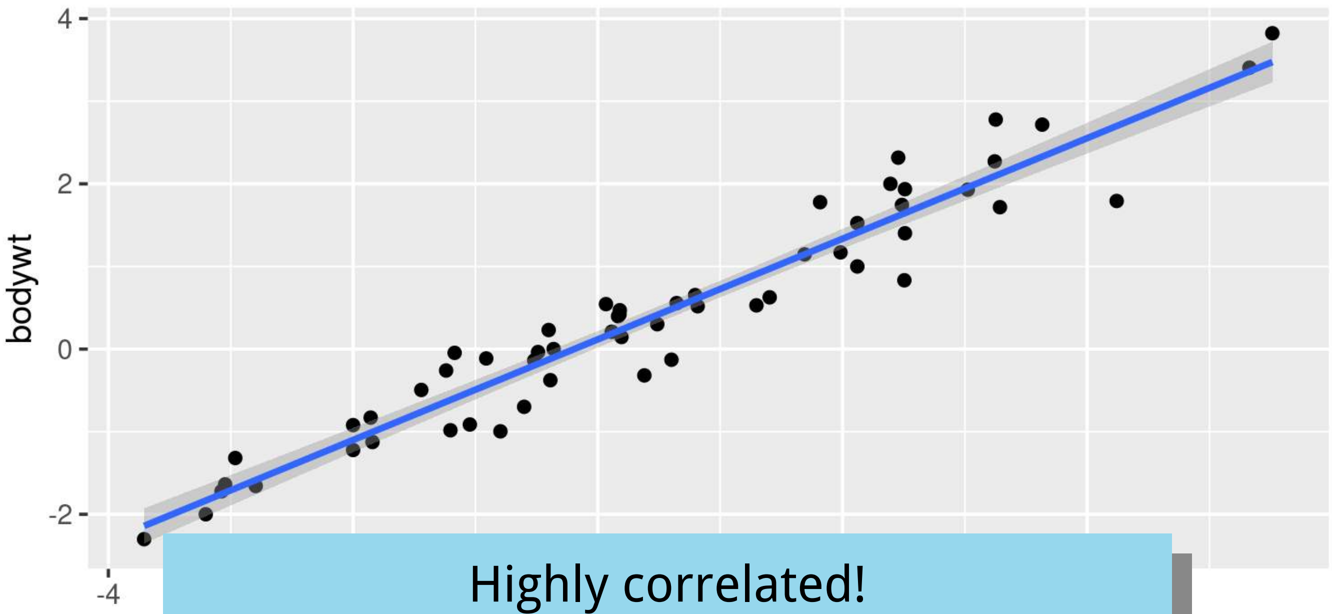


Multicollinearity (collinearity)

Look at our two explanatory variables:

```
ggplot(data = msleep_log, aes(x = brainwt, y = bodywt)) +
  geom_point() +
  stat_smooth(method = "lm")
```

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



Highly correlated!

Interpreting linear models

```
summary(m_log)
```

```
## 
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.36819 -0.13517 -0.01879  0.05897  0.36550 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976 
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Model

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Effects!

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439                      2e-16 ***
## bodywt       0.18705                      3e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Intercept

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439                      2e-16
## bodywt       0.18705                      8e-09
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Intercept

For **bodywt** of 0 kg (log10 units),
species has sleep cycle of -0.494 hours (log10 units)

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705                      8e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Slope

Interpreting linear models

```r
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16
## bodywt       0.18705                     3e-09
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Slope

For each 1 kg (log10 units) increase in **bodywt** sleep cycle increases by 0.187 hours (log10 units)

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

$y = mx + b$

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

$y = 0.187x - 0.494$

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Variability in the estimate

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Significance of the results

Interpreting linear models

```
summary(m_log)
```

```
## 
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.36819 -0.13517 -0.01879  0.05897  0.36550 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,	Adjusted R-squared:  0.6976 
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Is the **intercept** significantly different from zero? (Yes, P < 0.0001)

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Is the **slope** significantly different from zero?
(Yes, P < 0.0001)

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

Is there a significant relationship between our variables? (**Yes**)

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,              uared:  0.6976
## F-statistic: 72.51 on 1 and 30             1.679e-09
```

R^2

Interpreting linear models

```
summary(m_log)
```

```
##
## Call:
## lm(formula = sleep_cycle ~ bodywt, data = msleep_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36819 -0.13517 -0.01879  0.05897  0.36550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49439    0.03074 -16.082 2.72e-16 ***
## bodywt       0.18705    0.02197   8.515 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1734 on 30 degrees of freedom
##   (51 observations deleted due to missingness)
## Multiple R-squared:  0.7073,    Adjusted R-squared:  0.6976
## F-statistic: 72.51 on 1 and 30 DF,  p-value: 1.679e-09
```

R^2 Adjusted for the number of variables

ANOVAs

Same as before, but now with categorical variables (vore & conservation)

```
m <- lm(sleep_total ~ vore + conservation, data = msleep)
```

ANOVAs

Same as before, but now with categorical variables (vore & conservation)

```
m <- lm(sleep_total ~ vore + conservation, data = msleep)
```

What are these variables?

```
count(msleep, vore)
```

```
## # A tibble: 5 × 2
##   vore        n
##   <chr>   <int>
## 1 carni      19
## 2 herbi      32
## 3 insecti     5
## 4 omni       20
## 5 <NA>        7
```

```
count(msleep, conservation)
```

```
## # A tibble: 7 × 2
##   conservation     n
##   <chr>        <int>
## 1 cd               2
## 2 domesticated    10
## 3 en               4
## 4 lc              27
## 5 nt               4
## 6 vu               7
## 7 <NA>            29
```

cd = conservation dependent, lc = least concern, vu = vulnerable, nt = non-threatened, en = endangered, etc.

ANOVAs

Same as before, but now with categorical variables (vore & conservation)

```
m <- lm(sleep_total ~ vore + conservation, data = msleep)
```

What are these variables?

```
count(msleep, vore)
```

```
## # A tibble: 5 × 2
##   vore        n
##   <chr>   <int>
## 1 carni      19
## 2 herbi      32
## 3 insecti     5
## 4 omni       20
## 5 <NA>        7
```

```
count(msleep, conservation)
```

```
## # A tibble: 7 × 2
##   conservation     n
##   <chr>        <int>
## 1 cd               2
## 2 domesticated    10
## 3 en               4
## 4 lc              27
## 5 nt               4
```

Note: This makes no sense!
Why would conservation status ever predict sleep?

cd = conservation [illegible] d, en = endangered, etc.

Interpreting ANOVA *summaries*

```
summary(m)
```

```
##
## Call:
## lm(formula = sleep_total ~ vore + conservation, data = msleep)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1632 -2.7702  0.2547  2.8866  6.4368
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    3.851      3.041   1.266  0.21231
## voreherbi                     -3.101      1.431  -2.167  0.03585 *
## voreinsecti                    1.853      2.800   0.662  0.51174
## voreomni                      -1.401      1.945  -0.720  0.47525
## conservationdomesticated       6.040      3.265   1.850  0.07121 .
## conservationen                10.262      3.688   2.783  0.00797 **
## conservationlc                 9.414      3.141   2.997  0.00452 **
## conservationnt                11.450      3.638   3.147  0.00299 **
## conservationvu                 4.407      3.353   1.314  0.19575
## ---
```

Interpreting ANOVA *summaries*

```
summary(m)
```

```
##
## Call:
## lm(formula = sleep_total ~ vore + conservation, data = msleep)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1632 -2.7702  0.2547  2.8866  6.4368
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  3.851      3.041   1.266  0.21231
## voreherbi                   -3.101      1.431  -2.167  0.03585 *
## voreinsecti                  1.853      2.800   0.662  0.51174
## voreomni                    -1.401      1.945  -0.720  0.47525
## conservationdomesticated     6.040      3.265   1.850  0.07121 .
## conservationen              10.262      3.688   2.783  0.00797 **
## conservationlc               9.414      3.141   2.997  0.00452 **
## conservationnt              11.450      3.638   3.147  0.00299 **
## conservationvu               4.407      3.353   1.314  0.19575
## ---
```

Treatment Contrasts

Effect of **`vore`** categories, each compared to first category (**`carni`**)

Interpreting ANOVA *summaries*

```
summary(m)
```

```
##
## Call:
## lm(formula = sleep_total ~ vore + conservation, data = msleep)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1632 -2.7702  0.2547  2.8866  6.4368
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  3.851      3.041   1.266  0.21231
## voreherbi                   -3.101      1.431  -2.167  0.03585 *
## voreinsecti                  1.853      2.800   0.662  0.51174
## voreomni                    -1.401      1.945  -0.720  0.47525
## conservationdomesticated     6.040      3.265   1.850  0.07121 .
## conservationen              10.262      3.688   2.783  0.00797 **
## conservationlc               9.414      3.141   2.997  0.00452 **
## conservationnt              11.450      3.638   3.147  0.00299 **
## conservationvu               4.407      3.353   1.314  0.19575
## ---
```

Treatment Contrasts

For example

Total sleep in herbivores (**`herbi`**) is significantly (P = 0.03585) lower (Est = -3.101) than in carnivores (**`carni`**, baseline category)

Interpreting ANOVA *summaries*

```
summary(m)
```

```
##
## Call:
## lm(formula = sleep_total ~ vore + conservation, data = msleep)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1632 -2.7702  0.2547  2.8866  6.4368
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     3.851      3.041   1.266  0.21231
## voreherbi                      -3.101      1.431  -2.167  0.03585 *
## voreinsecti                     1.853      2.800   0.662  0.51174
## voreomni                       -1.401      1.945  -0.720  0.47525
## conservationdomesticated        6.040      3.265   1.850  0.07121 .
## conservationen                 10.262      3.688   2.783  0.00797 **
## conservationlc                  9.414      3.141   2.997  0.00452 **
## conservationnt                 11.450      3.638   3.147  0.00299 **
## conservationvu                  4.407      3.353   1.314  0.19575
## ---
```

Treatment Contrasts

Effect of **`conservation`** categories, each compared to first category (**`cd`**)

Interpreting ANOVA *tables*

```
anova(m)
```

```
## Analysis of Variance Table
## 
## Response: sleep_total
##              Df Sum Sq Mean Sq F value   Pr(>F)   
## vore          3 167.57  55.856  3.1960 0.032757 * 
## conservation  5 342.97  68.595  3.9249 0.005095 **
## Residuals    43 751.51  17.477                    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpreting ANOVA *tables*

```
anova(m)
```

```
## Analysis of Variance Table
##
## Response: sleep_total
##              Df Sum Sq Mean Sq F value   Pr(>F)
## vore          3 167.57  55.856  3.1960 0.032757 *
## conservation  5 342.97  68.595  3.9249 0.005095 **
## Residuals    43 751.51  17.477
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Overall differences among categories of **`vore`**

Interpreting ANOVA *tables*

```
anova(m)
```

```
## Analysis of Variance Table
##
## Response: sleep_total
##              Df Sum Sq Mean Sq F value   Pr(>F)
## vore          3 167.57  55.856  3.1960 0.032757 *
## conservation  5 342.97  68.595  3.9249 0.005095 **
## Residuals    43 751.51  17.477
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
```

Overall differences among categories of **`conservation`**

Interpreting ANOVA *tables*

```
anova(m)
```

```
## Analysis of Variance Table
##
## Response: sleep_total
##              Df Sum Sq Mean Sq F value   Pr(>F)
## vore          3 167.57  55.856  3.1960 0.032757 *
## conservation  5 342.97  68.595  3.9249 0.005095 **
## Residuals    43 751.51  17.477
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This is a **Type I ANOVA**

Interpreting ANOVA *tables*

Type II ANOVA

```r
library(car)
Anova(m, type = "2")
```

```
## Anova Table (Type II tests)
##
## Response: sleep_total
##              Sum Sq Df F value   Pr(>F)
## vore         121.75  3  2.3220 0.088502 .
## conservation 342.97  5  3.9249 0.005095 **
## Residuals    751.51 43
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*'
0.05 '.' 0.1 ' ' 1
```

Type III ANOVA

```r
library(car)
Anova(m, type = "3")
```

```
## Anova Table (Type III tests)
##
## Response: sleep_total
##              Sum Sq Df F value   Pr(>F)
## (Intercept)   28.01  1  1.6029 0.212313
## vore         121.75  3  2.3220 0.088502 .
## conservation 342.97  5  3.9249 0.005095 **
## Residuals    751.51 43
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*'
0.05 '.' 0.1 ' ' 1
```

ANOVAs and Post-Hoc Tests

Chicks and Diet

Prep data

```
chicks <- filter(ChickWeight,
                 Time == 21,
                 !(Chick %in% 1:7))
head(chicks)
```

```
## Grouped Data: weight ~ Time | Chick
##   weight Time Chick Diet
## 1     98   21     9    1
## 2    124   21    10    1
## 3    175   21    11    1
## 4    205   21    12    1
## 5     96   21    13    1
## 6    266   21    14    1
```

How many chicks per diet?

```
count(chicks, Diet)
```

```
## Grouped Data: weight ~ Time | Chick
##   Diet  n
## 1    1  9
## 2    2 10
## 3    3 10
## 4    4  9
```

Chicks and Diet

4 different diets, how do chicks gain weight on each diet?

```
m <- lm(weight ~ Diet, data = chicks)
```

Chicks and Diet

4 different diets, how do chicks gain weight on each diet?

```
m <- lm(weight ~ Diet, data = chicks)
```

```
anova(m)
```

```
## Analysis of Variance Table
##
## Response: weight
##             Df Sum Sq Mean Sq F value   Pr(>F)
## Diet         3  68673 22891.0  5.5334 0.003331 **
## Residuals 34 140654  4136.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1
```

Chicks and Diet

4 different diets, how do chicks gain weight on each diet?

```
m <- lm(weight ~ Diet, data = chicks)
```

```
anova(m)
```

```
## Analysis of Variance Table
##
## Response: weight
##           Df Sum Sq Mean Sq F value   Pr(>F)
## Diet       3  68673 22891.0  5.5334 0.003331 **
## Residuals 34 140654  4136.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1
```

```
summary(m)
```

```
##
## Call:
## lm(formula = weight ~ Diet, data = chicks)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -140.700  -39.414   -1.056   40.908  116.300
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   153.33      21.44   7.152 2.87e-08 ***
## Diet2          61.37      29.55   2.077 0.045472 *
## Diet3         116.97      29.55   3.958 0.000365 ***
## Diet4          85.22      30.32   2.811 0.008143 **
## ---
```

Chicks and Diet

4 different diets, how do chicks gain weight on each diet?

```
m <- lm(weight ~ Diet, data = chicks)
```

```
anova(m)
```

```
## Analysis of Variance Table
##
## Response: weight
##            Df Sum Sq Mean Sq F value   Pr(>F)
## Diet        3  68673 22891.0  5.5334 0.003331 **
## Residuals 34 140654  4136.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1
```

Need post-hoc tests to
test each Diet against the others

```
summary(m)
```

```
##
## Call:
## lm(formula = weight ~ Diet, data = chicks)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -140.700  -39.414   -1.056   40.908  116.300
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   153.33      21.44   7.152 2.87e-08 ***
## Diet2          61.37      29.55   2.077 0.045472 *
## Diet3         116.97      29.55   3.958 0.000365 ***
## Diet4          85.22      30.32   2.811 0.008143 **
## ---
```

Post-Hoc Tests

```r
library(multcomp)
mult_pairwise <- glht(m, linfct = mcp(Diet = "Tukey"))   # All Pair-wise comparisons
```

- Package **`multcomp`**
- Function **`glht()`** (**g**eneral **l**inear **h**ypothesis **t**esting)
- Model of interest (here, **m**)
- Argument **`linfct`** (**lin**ear **f**un**ct**ion, i.e., Which post-hoc tests?)
- Function **`mcp()`** (**m**ultiple **c**om**p**arisons)
- Specify the variable you want to compare (Here, **Diet**)
- Specify the way the categories should be compared:
  - **`"Tukey"`** reflects **Tukey Contrasts** (i.e., all pairwise comparisons)
  - **`"Dunnett"`** reflects **Dunnett's comparison with a control**

Post-Hoc Tests

All pair-wise comparisons

```
summary(mult_pairwise)
```

```
##      Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: lm(formula = weight ~ Diet, data = chicks)
##
## Linear Hypotheses:
##             Estimate Std. Error t value Pr(>|t|)
## 2 - 1 == 0    61.37      29.55   2.077  0.18124
## 3 - 1 == 0   116.97      29.55   3.958  0.00186 **
## 4 - 1 == 0    85.22      30.32   2.811  0.03873 *
## 3 - 2 == 0    55.60      28.76   1.933  0.23359
## 4 - 2 == 0    23.86      29.55   0.807  0.85056
## 4 - 3 == 0   -31.74      29.55  -1.074  0.70723
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Each group compared to each other
`2 - 1 == 0` reflects hypothesis that
Diet 2 - Diet 1 is equal to 0
(i.e., no difference)

Post-Hoc Tests

All pair-wise comparisons

```
summary(mult_pairwise)
```

```
##       Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: lm(formula = weight ~ Diet, data = chicks)
##
## Linear Hypotheses:
##              Estimate Std. Error t value Pr(>|t|)
## 2 - 1 == 0     61.37      29.55   2.077  0.18124
## 3 - 1 == 0    116.97      29.55   3.958  0.00186 **
## 4 - 1 == 0     85.22      30.32   2.811  0.03873 *
## 3 - 2 == 0     55.60      28.76   1.933  0.23359
## 4 - 2 == 0     23.86      29.55   0.807  0.85056
## 4 - 3 == 0    -31.74      29.55  -1.074  0.70723
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Each group compared to each other
`2 - 1 == 0` reflects hypothesis that
Diet 2 - Diet 1 is equal to 0
(i.e., no difference)

Post-Hoc Tests

Specify the P-Value adjustment

```
summary(mult_pairwise, test = adjusted("BH"))
```

```
## 	 Simultaneous Tests for General Linear Hypotheses
## 
## Multiple Comparisons of Means: Tukey Contrasts
## 
## 
## Fit: lm(formula = weight ~ Diet, data = chicks)
## 
## Linear Hypotheses:
##             Estimate Std. Error t value Pr(>|t|)   
## 2 - 1 == 0    61.37      29.55   2.077  0.09094 . 
## 3 - 1 == 0   116.97      29.55   3.958  0.00219 **
## 4 - 1 == 0    85.22      30.32   2.811  0.02443 * 
## 3 - 2 == 0    55.60      28.76   1.933  0.09241 . 
## 4 - 2 == 0    23.86      29.55   0.807  0.42515   
## 4 - 3 == 0   -31.74      29.55  -1.074  0.34837   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- BH method)
```

"BH" = Benjamini-Hochberg,
also known as FDR test
(see here for more details)

Post-Hoc Tests

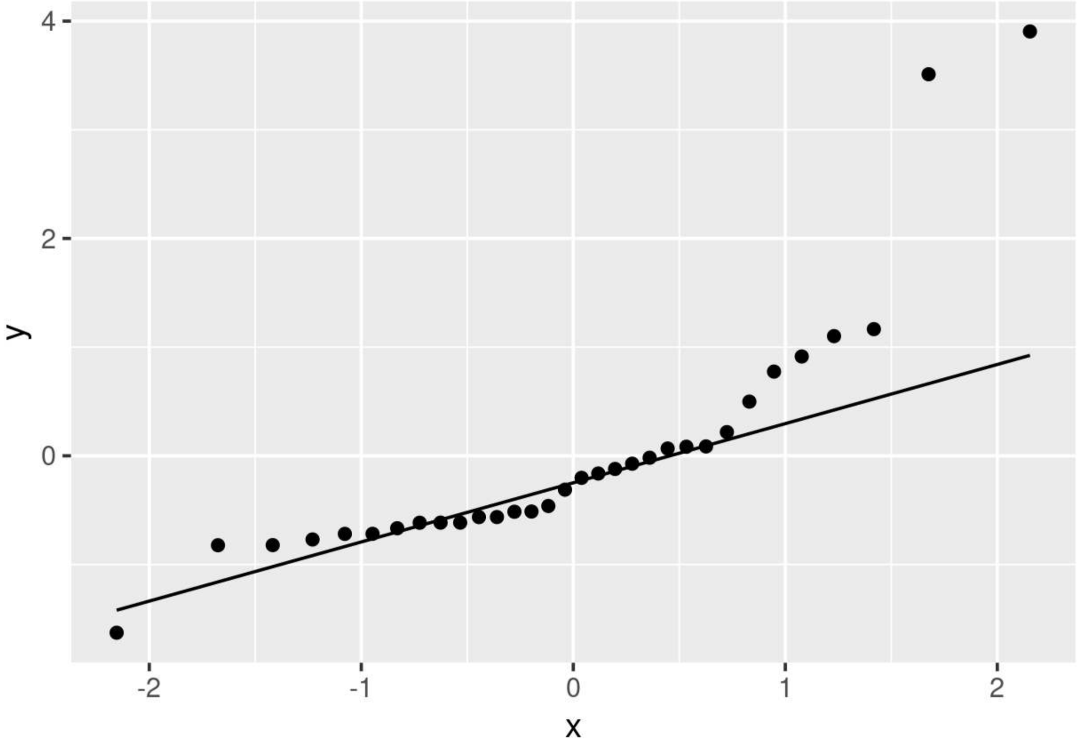
Argument	P-Value Adjustment
`single-step`	Adjusted p values based on the joint normal or t distribution of the linear function
`Shaffer`	Shaffer Test
`Westfall`	Westfall Test
`free`	Multiple testing procedures under free combinations
`holm`	Holm Test
`hochberg`	Hochberg Test
`hommel`	Hommel Test
`bonferroni`	Bonferroni Correction
`BH or fdr`	Benjamini-Hochberg Test or False Discovery Rate Test
`BY`	Benjamini-Yekutieli Test
`none`	No P-Value Adjustment

Data Transformations

Transformations

```r
m <- lm(sleep_cycle ~ bodywt, data = msleep)

d <- data.frame(residuals = residuals(m),
                std_residuals = rstudent(m),
                fitted = fitted(m),
                cooks = cooks.distance(m))

d <- mutate(d, observation = 1:nrow(d))
```

```r
ggplot(data = d, aes(sample = std_residuals)) +
  stat_qq() +
  stat_qq_line()
```



Good for addressing non-normality of residuals, and problems with variance

Transformations

Order of Operations

1. See the need (e.g., non-normal residuals, heteroscedacity)
2. Figure out which transformation
3. Apply the transformation
4. Check model assumptions
5. Rinse and repeat as needed

Transformations: Common options

Table of transformations in R

```
data_trans <- mutate(data, y_trans = 1/y^2)

data_trans <- mutate(data, y_trans = 1/y)

data_trans <- mutate(data, y_trans = 1/sqrt(y))

data_trans <- mutate(data, y_trans = log(y))

data_trans <- mutate(data, y_trans = log10(y))

data_trans <- mutate(data, y_trans = sqrt(y))

data_trans <- mutate(data, y_trans = y^2)

data_trans <- mutate(data, y_trans = asin(sqrt(y/100)))

data_trans <- mutate(data, y_trans = (y^lambda - 1)/lambda)
```

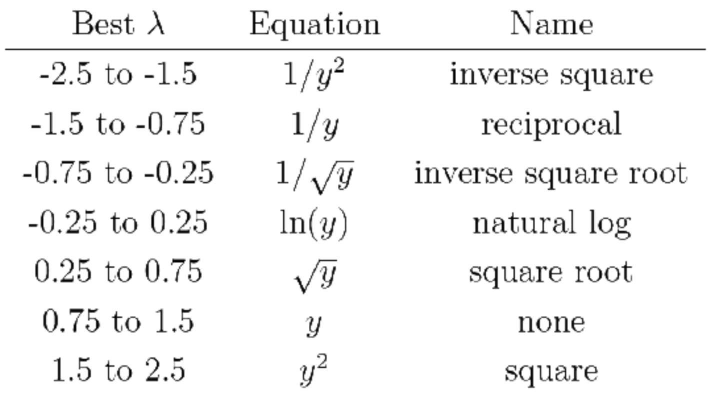
Transformation	R Code
Inverse square	`1/y^2`
Reciprocal	`1/y`
Inverse square root	`1/sqrt(y)`
Natural log (ln)	`log(y)`
Log base 10	`log10(y)`
Square root	`sqrt(y)`
Square	`y^2`
Box Cox	`(y^lambda - 1) / lambda`
Arcsine-sqare-root	`asin(sqrt(y/100))`

`data_trans` is the NEW data frame, **`y_trans`** is your TRANSFORMED y-value

Transformations: How to choose?

- Based on what you know (often discipline specific standards for certain data types)
- Based on what you see (does it look exponential or logarithmic?)
- Based on trial and error (try different transformations and see how it goes)
- Based on Box-Cox lambda (λ)

Can EITHER apply λ through Box-Cox transformation OR use it to indicate best transformation



Best λ	Equation	Name
-2.5 to -1.5	$1/y^2$	inverse square
-1.5 to -0.75	$1/y$	reciprocal
-0.75 to -0.25	$1/\sqrt{y}$	inverse square root
-0.25 to 0.25	$\ln(y)$	natural log
0.25 to 0.75	$\sqrt{y}$	square root
0.75 to 1.5	y	none
1.5 to 2.5	y^2	square

Transformations: Find λ

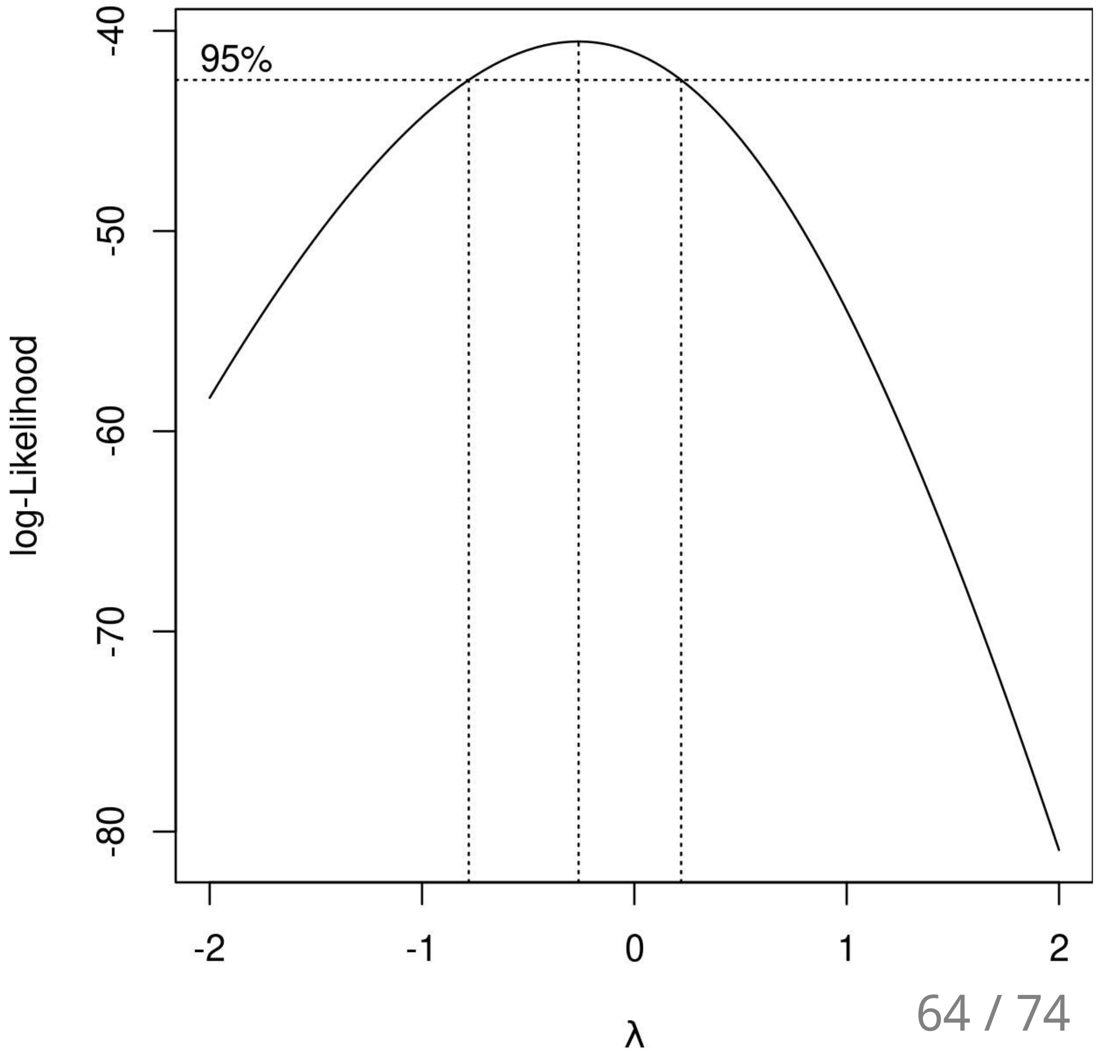
Plot of λ

```
library(MASS)
b <- boxcox(m)
```

Exact λ

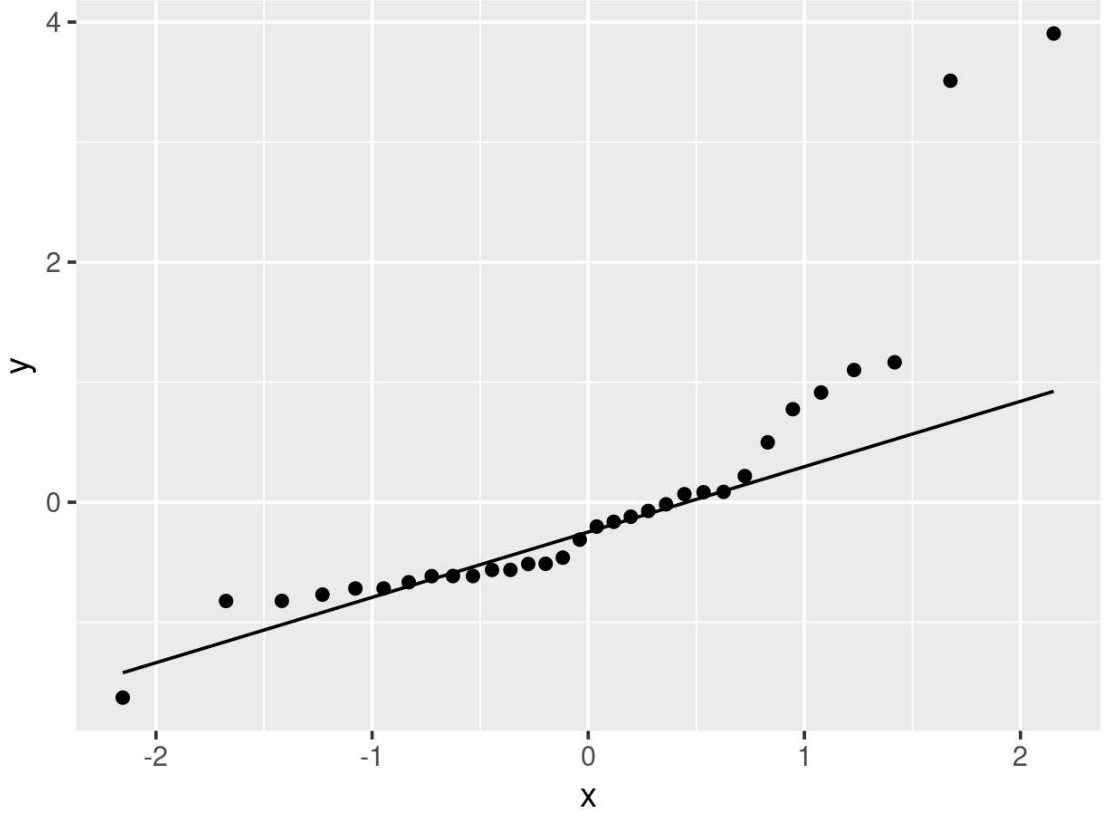
```
b$x[b$y == max(b$y)]
```

```
## [1] -0.2626263
```

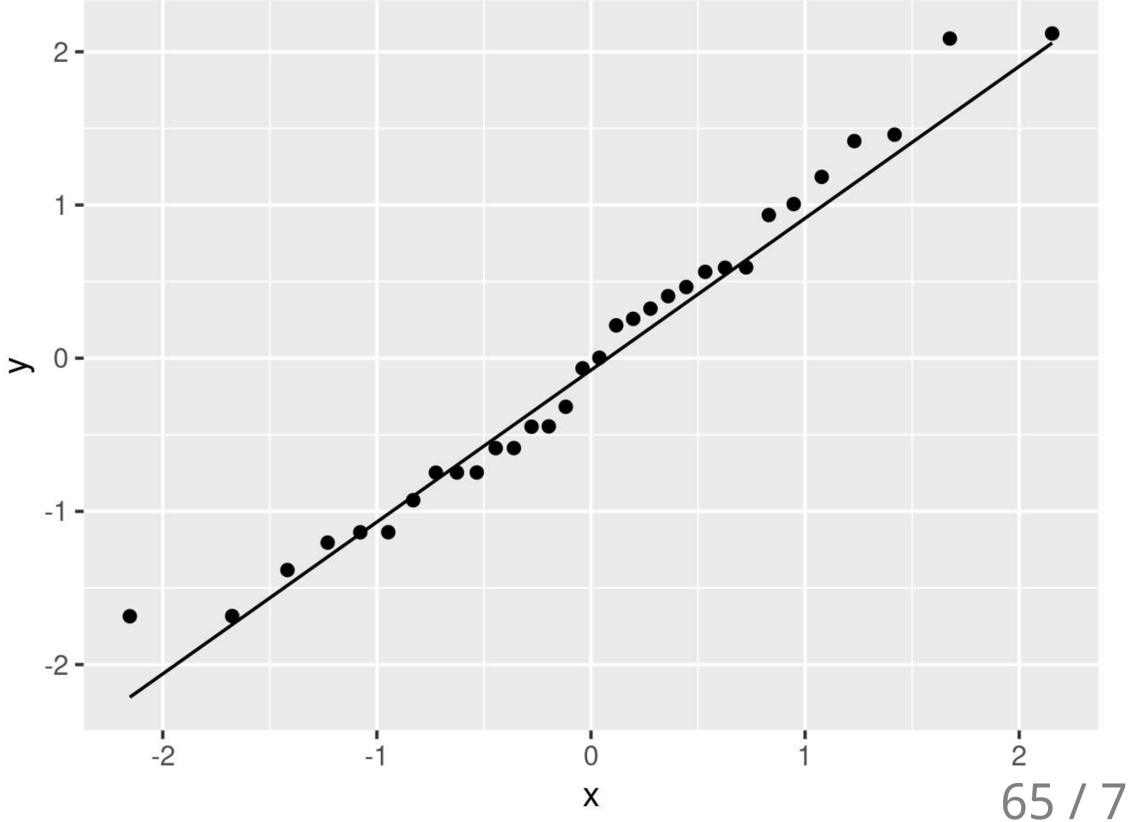


Apply the transformation

```
msleep_trans <- mutate(msleep, sleep_cycle = (sleep_cycle^(-0.26) - 1) / -0.26)
m_trans <- lm(sleep_cycle ~ bodywt, data = msleep_trans)
```



Q-Q Normality Plot: No transformation



Q-Q Normality Plot: Box-Cox transformation

Generalized Linear Models

Generalized Linear Models

Normal distribution - Gaussian Distribution

```
lm(y ~ x1 * x2, data = my_data)
```

Count data - Poisson Family

```
glm(counts ~ x1 * x2, family = "poisson", data = my_data)
```

Binary (0/1, Logistic Regression) - Binomial Distribution

```
glm(y ~ x1 * x2, family = "binomial", data = my_data)
```

Proportion with binary outcomes (10 yes, 5 no) - Binomial Distribution

```
glm(cbind(Yes, No) ~ x1 * x2, family = "binomial", data = my_data)
```

Poisson - Run

Download data

```
p <- read.csv("https://stats.idre.ucla.edu/stat/data/poisson_sim.csv")
p <- mutate(p, program = factor(prog, levels = 1:3,
                                labels = c("General", "Academic","Vocational")))
head(p)
```

```
##    id num_awards prog math    program
## 1  45          0    3   41 Vocational
## 2 108          0    1   41    General
## 3  15          0    3   44 Vocational
## 4  67          0    3   42 Vocational
## 5 153          0    3   40 Vocational
## 6  51          0    1   42    General
```

Run model

```
m <- glm(num_awards ~ program + math, family = "poisson", data = p)
```

Poisson - Evaluate

```
summary(m)
```

```
## glm(formula = num_awards ~ program + math, family = "poisson",
##     data = p)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2043  -0.8436  -0.5106   0.2558   2.6796
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -5.24712    0.65845  -7.969 1.60e-15 ***
## programAcademic    1.08386    0.35825   3.025  0.00248 **
## programVocational  0.36981    0.44107   0.838  0.40179
## math               0.07015    0.01060   6.619 3.63e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 287.67  on 199  degrees of freedom
## Residual deviance: 189.45  on 196  degrees of freedom
```

Look at deviance vs. df:

```
deviance(m)
```

```
## [1] 189.4496
```

```
df.residual(m)
```

```
## [1] 196
```

```
deviance(m) / df.residual(m)
```

```
## [1] 0.9665797
```

Nice! (should be close to 1)

Binary (0/1 - Logistic Regression)

Data

```
binary <- read.csv("https://stats.idre.ucla.edu/stat/data/binary.csv")
head(binary)
```

```
##   admit gre  gpa rank
## 1     0 380 3.61    3
## 2     1 660 3.67    3
## 3     1 800 4.00    1
## 4     1 640 3.19    4
## 5     0 520 2.93    4
## 6     1 760 3.00    2
```

Run model

```
m <- glm(admit ~ gpa, family = "binomial", data = binary)
```

Binary (0/1 - Logistic Regression)

Check results

```
summary(m)
```

```
## Call:
## glm(formula = admit ~ gpa, family = "binomial", data = binary)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -1.1131  -0.8874  -0.7566   1.3305   1.9824  
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  -4.3576     1.0353  -4.209 2.57e-05 ***
## gpa           1.0511     0.2989   3.517 0.000437 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 486.97  on 398  degrees of freedom
```

Convert to Odd's Ratios

```
exp(coef(m))
```

```
## (Intercept)         gpa 
##  0.01280926  2.86082123
```

e.g., The odds of being admitted increase by a factor of 2.86 (x2.86 times more likely) for every unit increase in GPA.

Binary Outcomes

Proportion with binary outcomes (e.g., 10 yes, 5 no)

Get the data

```
admissions <- as.data.frame(UCBAdmissions)
admissions <- spread(admissions, Admit, Freq)
head(admissions)
```

```
##   Gender Dept Admitted Rejected
## 1   Male    A      512      313
## 2   Male    B      353      207
## 3   Male    C      120      205
## 4   Male    D      138      279
## 5   Male    E       53      138
## 6   Male    F       22      351
```

Run model

```
m <- glm(cbind(Admitted, Rejected) ~ Gender, family = "binomial", data = admissions)
```

Binary Outcomes

Check results

```
summary(m)
```

```
## Call:
## glm(formula = cbind(Admitted, Rejected) ~ Gender, family = "binomial",
##     data = admissions)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -16.7915   -4.7613   -0.4365    5.1025   11.2022
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.22013    0.03879  -5.675 1.38e-08 ***
## GenderFemale -0.61035    0.06389  -9.553  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 877.06  on 11  degrees of freedom
## Residual deviance: 783.61  on 10  degrees of freedom
```

```
deviance(m)
```

```
## [1] 783.607
```

```
df.residual(m)
```

```
## [1] 10
```

```
deviance(m) / df.residual(m)
```

```
## [1] 78.3607
```

Oops, over-dispersed

Binary Outcomes

Try again with 'quasibinomial' family

```
m <- glm(cbind(Admitted, Rejected) ~ Gender, family = "quasibinomial", data = admissions)
summary(m)
```

```
## Call:
## glm(formula = cbind(Admitted, Rejected) ~ Gender, family = "quasibinomial",
##     data = admissions)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -16.7915   -4.7613   -0.4365    5.1025   11.2022
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.2201     0.3281  -0.671    0.517
## GenderFemale  -0.6104     0.5404  -1.129    0.285
##
## (Dispersion parameter for quasibinomial family taken to be 71.52958)
##
##     Null deviance: 877.06  on 11  degrees of freedom
## Residual deviance: 783.61  on 10  degrees of freedom
```