RegressionModelsNotes

Coursera Course by John Hopkins University

INSTRUCTORS: Dr. Brian Caffo, Dr. Roger D. Peng, Dr. Jeff Leek

Contents

Intro

This course covers regression analysis, least squares and inference using regression models. Special cases of the regression model, ANOVA and ANCOVA will be covered as well. Analysis of residuals and variability will be investigated. The course will cover modern thinking on model selection and novel uses of regression models including scatterplot smoothing.

GitHub Link for Lectures

Link to the GitHub for this course

Course Book

Regression Models for Data Science in R, through Leanpub

Further Reading: **Advanced Linear Models for Data Science**

Instructor's Note

"*We believe that the key word in Data Science is 'science'. Our course track is focused on providing you with three things:*
1) *An introduction to the key ideas behind working with data in a scientific way that will produce new and reproducible insight*
2) *An introduction to the tools that will allow you to execute on a data analytic strategy, from raw data in a database to a completed report with interactive graphics*
3) *Giving you plenty of hands on practice so you can learn the techniques for yourself.*

Regression Models represents a both fundamental and foundational component of the series, and it presents the single most practical data analysis toolset. Using only a bare minimum of mathematics, we will attempt to provide you with the fundamentals for the application and practice of regression. We are excited about the opportunity to attempt to scale Data Science education. We intend for the courses to be self-contained, fast-paced, and interactive, and we intend to run them frequently to give people with busy schedules the opportunity to work on material at their own pace.

Brian Caffo and the Data Science Track Team"

Data Science Specialization Community Site

The site is created using GitHub Pages

In addition, Johns Hopkins has **a site on Statistical Methods and Applications for Research in Technology** that Dr. Caffo helps manage.

Least Squares and Linear Regression

Regression

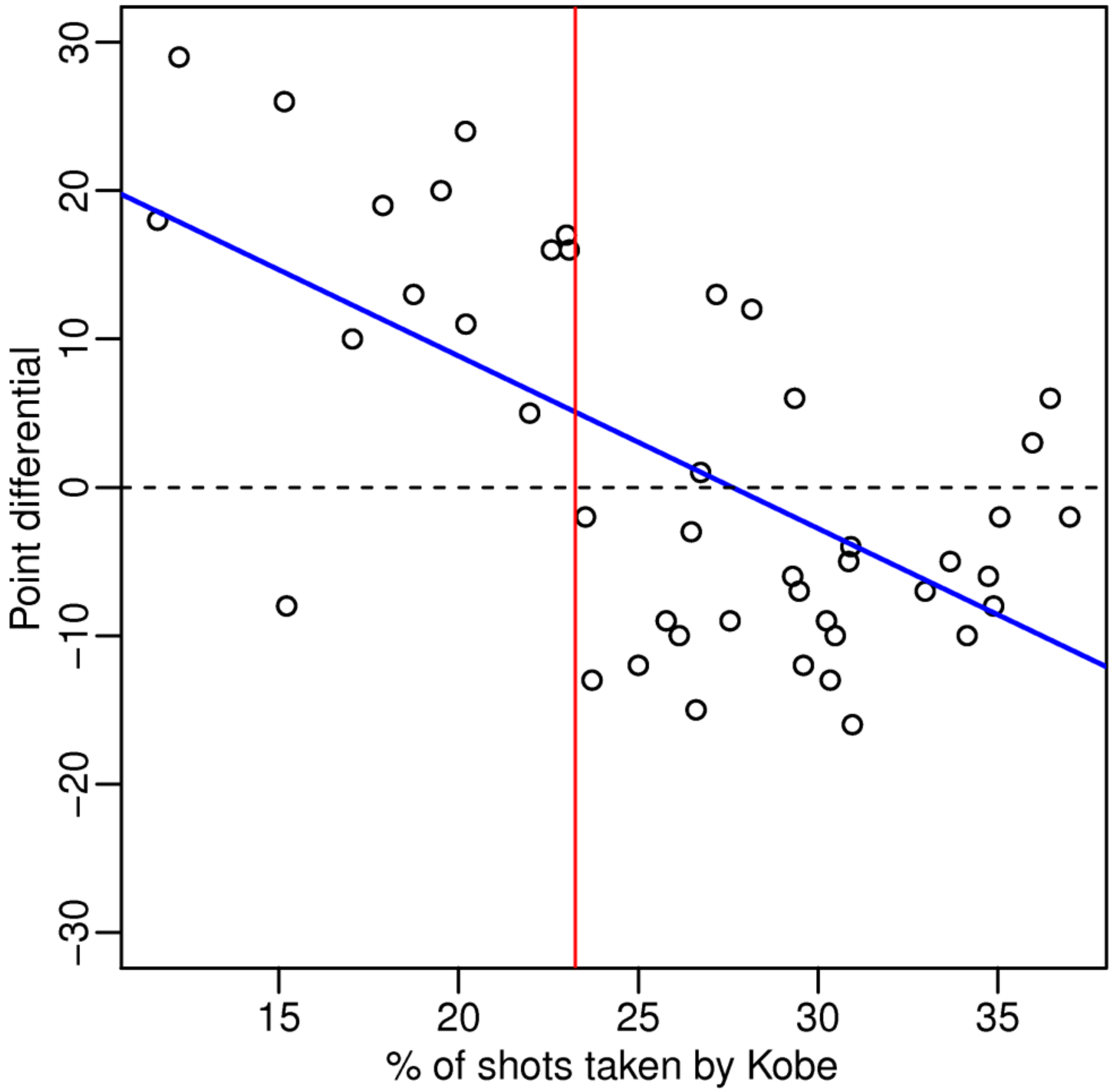
Introduction to Regression

- The simplicity and intrepretability offered by regression models should make them a first tool of choice for any practical problem.

- First discovered by **Francis Galton** who coined most of the terminology we use today.

Relevant Simply Statistics Post

Simply Statistics is a blog by Jeff Leek, Roger Peng and Rafael Irizarry, who wrote this post



- "Data supports claim that if Kobe stops ball hogging the Lakers will win more"

- "Linear regression suggests that an increase of 1% in percent of shots taken by Kobe results in a drop of 1.16 (+/- 0.22) in score differential."
  + Standard error given as "+/- 0.22"

Questions for this Class

In reference to Galton's parent/children height data, which can be accessed from the `galton` dataset in the `UsingR` package.
Consider trying to answer the following kinds of questions:
* To use the parents' heights to predict childrens' heights.

* To try to find a parsimonious (explain the data), easily described mean relationship between parent and children's heights.
* To investigate the variation in childrens' heights that appears unrelated to parents' heights (residual variation).
* To quantify what impact genotype information has beyond parental height in explaining child height.
* To figure out how/whether and what assumptions are needed to generalize findings beyond the data in question.
* Why do children of very tall parents tend to be tall, but a little shorter than their parents and why children of very short parents tend to be short, but a little taller than their parents? (This is a famous question called "Regression to the mean".)

Introduction to Basic Least Squares

- Let's look at the data first used by Francis Galton in 1885.

- Galton was a statistician who invented the term and concepts of regression and correlation, founded the journal Biometrika, and was the cousin of Charles Darwin.

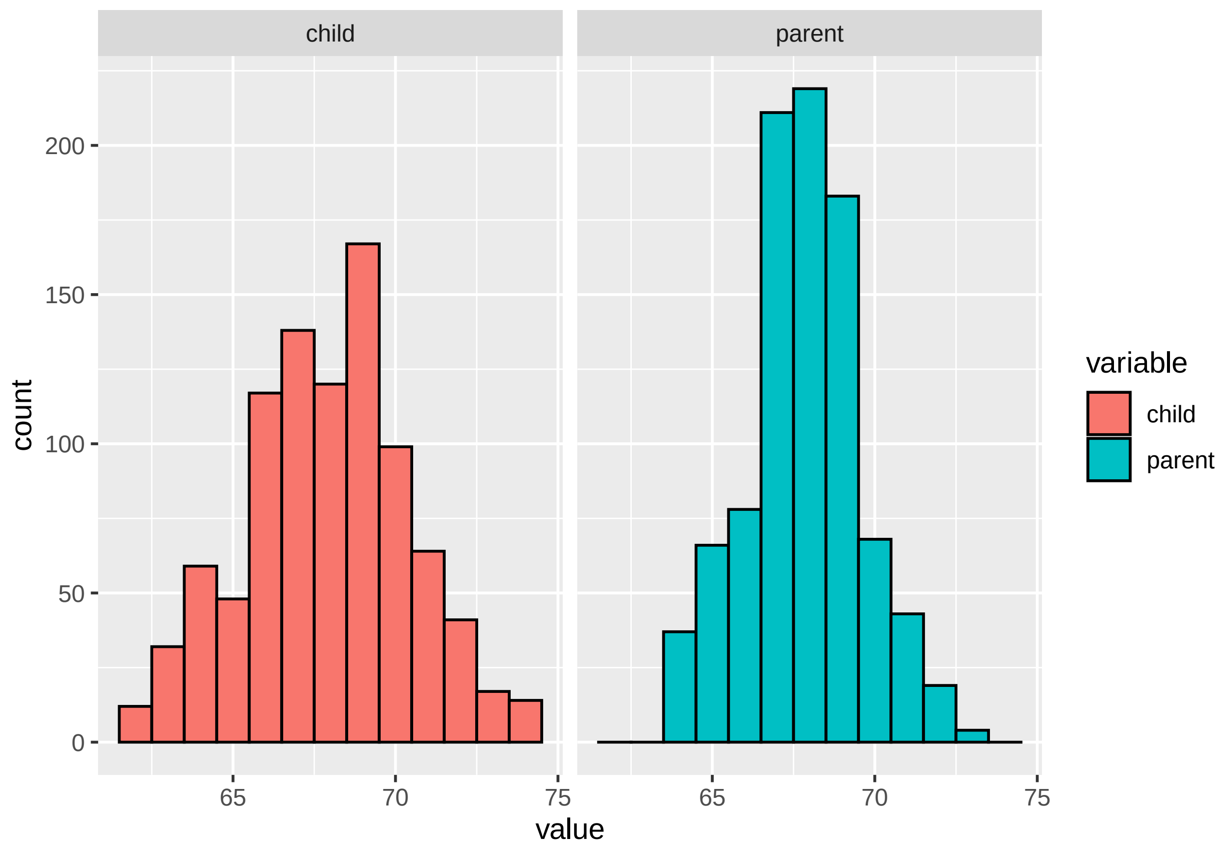
- Let's look at the marginal (parents disregarding children and children disregarding parents) distributions first.
  + Parent distribution is all heterosecual couples.
  + Correction for gender via multiplying female heights by 1.08.
  + Overplotting is an issue from discretization.

```
library(UsingR); data(galton); library(reshape2); library(tidyverse)
```

```
long <- melt(galton)
```

```
## No id variables; using all as measure variables
```

```
plot <- ggplot(long, aes(x = value, fill = variable)) +
        geom_histogram(colour = "#000000", binwidth = 1)
plot + facet_grid(.~variable)
```



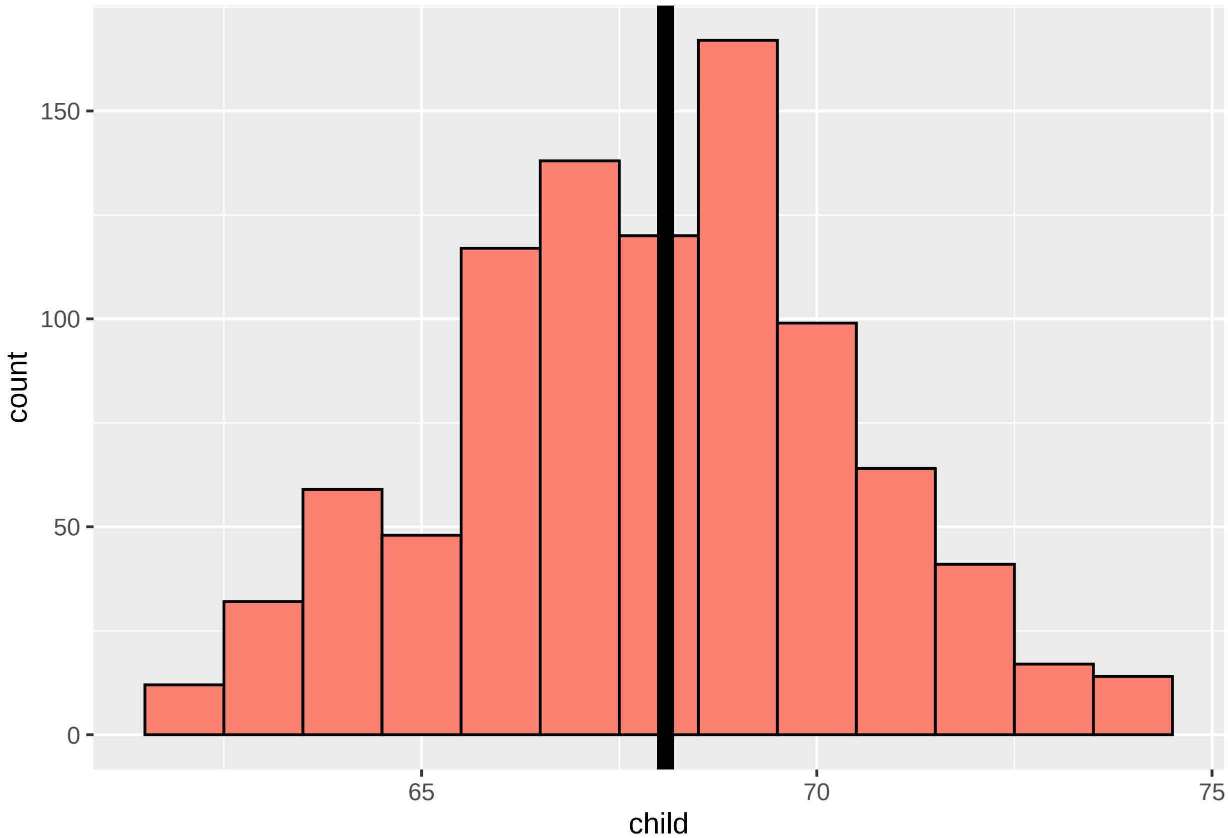
Finding the Middle via Least Squares

- Consider only the children's heights
  + How could one describe the "middle"?
  + One definition, let Y_i be the height of child i for $i = 1, ..., n = 928$, then define the middle as the value of μ that minimizes
  $\sum_{i=1}^{n} (Y_i - \mu)^2$

- This is the physical center of mass of the histogram.

- The result of this is that $\mu = \bar{Y}$

```
ggplot(galton, aes(x = child)) +
        geom_histogram(fill = "salmon", colour = "#000000", binwidth = 1) +
        geom_vline(xintercept = mean(galton$child), size = 3)
```



- The above plot of child heights has a mean of 68.0884698

Technical Details

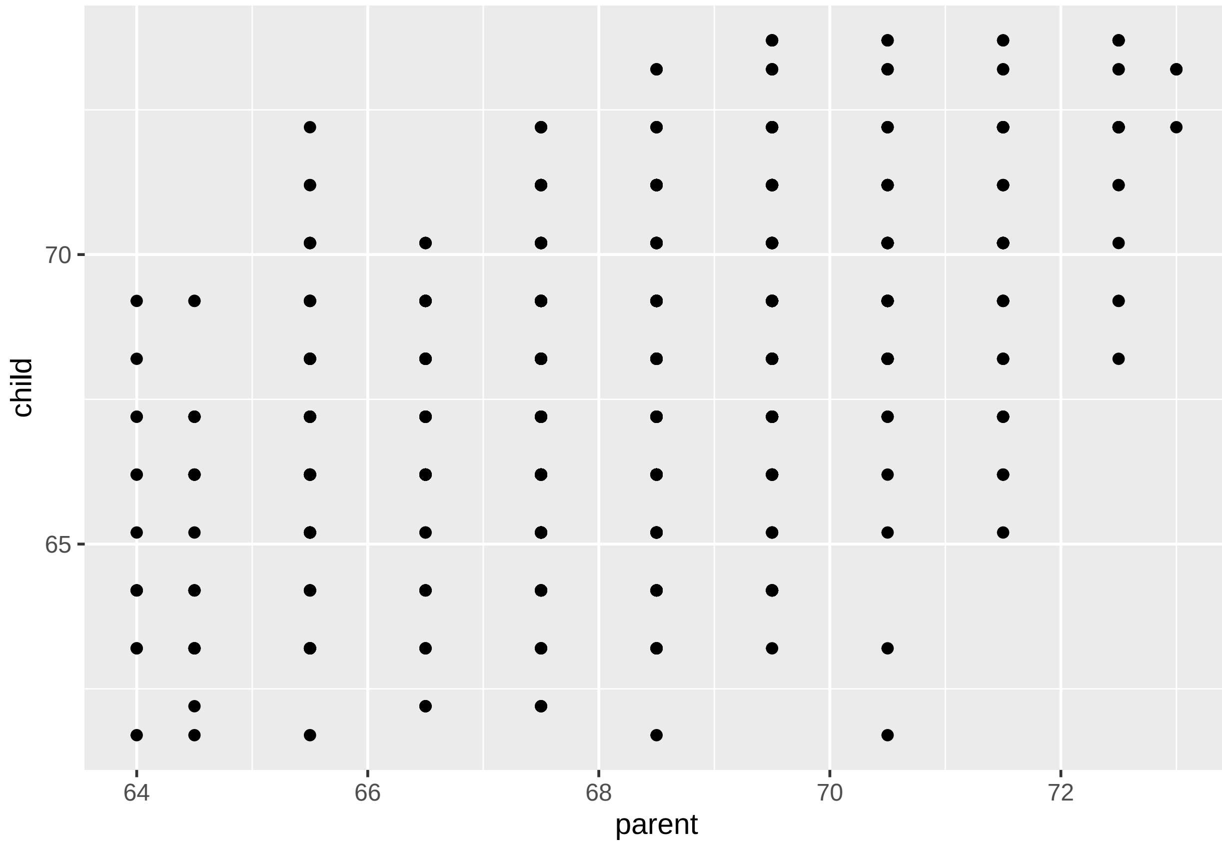
Proof that $\bar{Y}$ is the minimizer for $\sum_{i=1}^{n}(Y_i - \mu)^2$
$\sum_{i=1}^{n}(Y_i - \mu)^2 = \sum_{i=1}^{n}(Y_i - \bar{Y} + \bar{Y} - \mu)^2$
$= \sum_{i=1}^{n}(Y_i - \bar{Y}^2 + 2\sum_{i=1}^{n}(Y_i - \bar{Y})(\bar{Y} - \mu) + \sum_{i=1}^{n}(\bar{Y} - \mu)^2$
$= \sum_{i=1}^{n}(Y_i - \bar{Y})^2 + 2(\bar{Y} - \mu)\sum_{i=1}^{n}(Y_i - \bar{Y}) + \sum_{i=1}^{n}(\bar{Y} - \mu)^2$
$= \sum_{i=1}^{n}(Y_i - \bar{Y})^2 + 2(\bar{Y} - \mu)(\sum_{i=1}^{n} Y_i - n\bar{Y}) + \sum_{i=1}^{n}(\bar{Y} - \mu)^2$
$= \sum_{i=1}^{n}(Y_i - \bar{Y})^2 + 0 + \sum_{i=1}^{n}(\bar{Y} - \mu)^2$
$\geq \sum_{i=1}^{n}(Y_i - \bar{Y})^2$

Therefore, $\sum_{i=1}^{n}(Y_i - \mu)^2$ is minimized when $\bar{Y} = \mu$

Introductory Data Example

Comparing Childrens' Heights and Their Parents' Heights

```
ggplot(galton, aes(x = parent, y = child)) + geom_point()
```

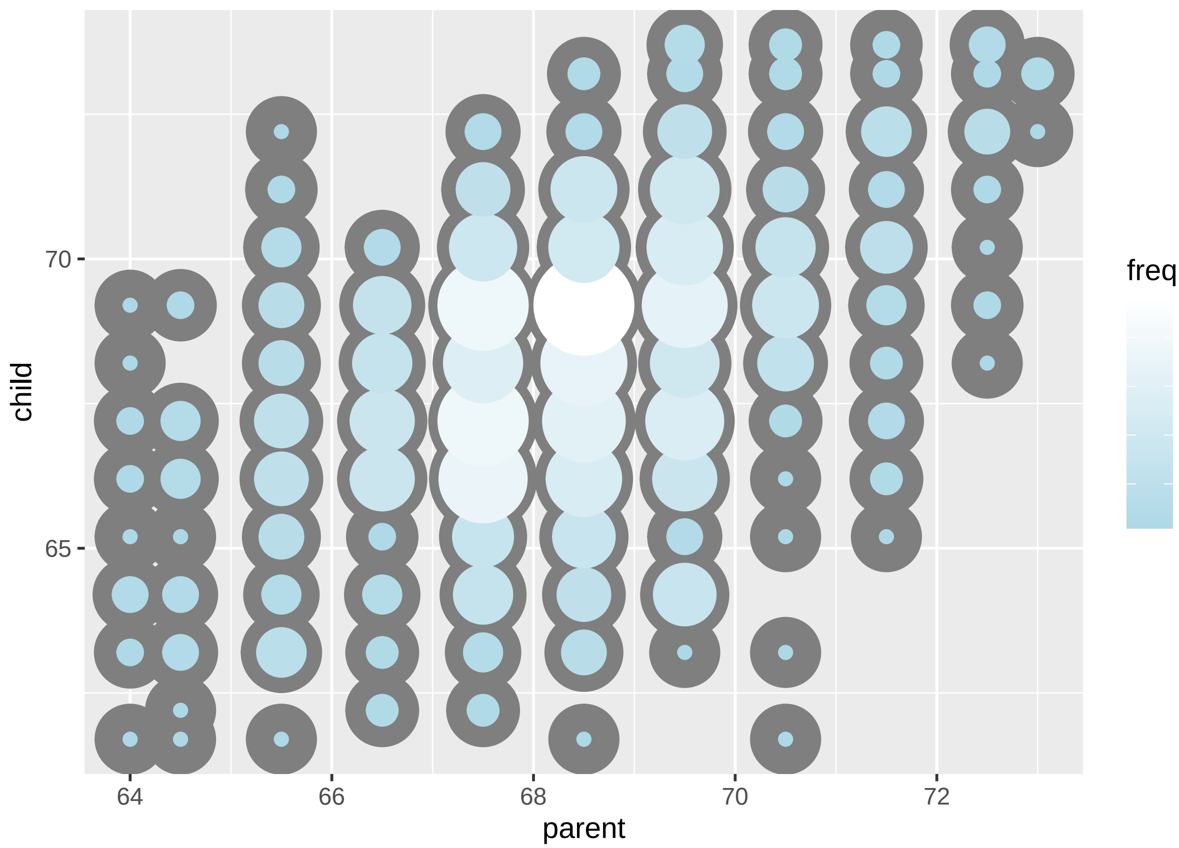


- These points are overplotted, there are multiple overlays at each point, so let's make a better plot

```
freqData <- as.data.frame(table(galton$child, galton$parent))
names(freqData) <- c("child", "parent", "freq")
freqData$child <- as.numeric(as.character(freqData$child))
freqData$parent <- as.numeric(as.character(freqData$parent))
plot <- ggplot(filter(freqData, freq > 0), aes(x = parent, y = child)) +
        scale_size(range = c(2, 20), guide = "none") +
        geom_point(colour = "grey50",
                   aes(size = freq + 20, show_guide = FALSE)) +
        geom_point(aes(colour = freq, size = freq)) +
        scale_colour_gradient(low = "lightblue", high = "#FFFFFF")
```

```
## Warning: Ignoring unknown aesthetics: show_guide
```

```
plot
```

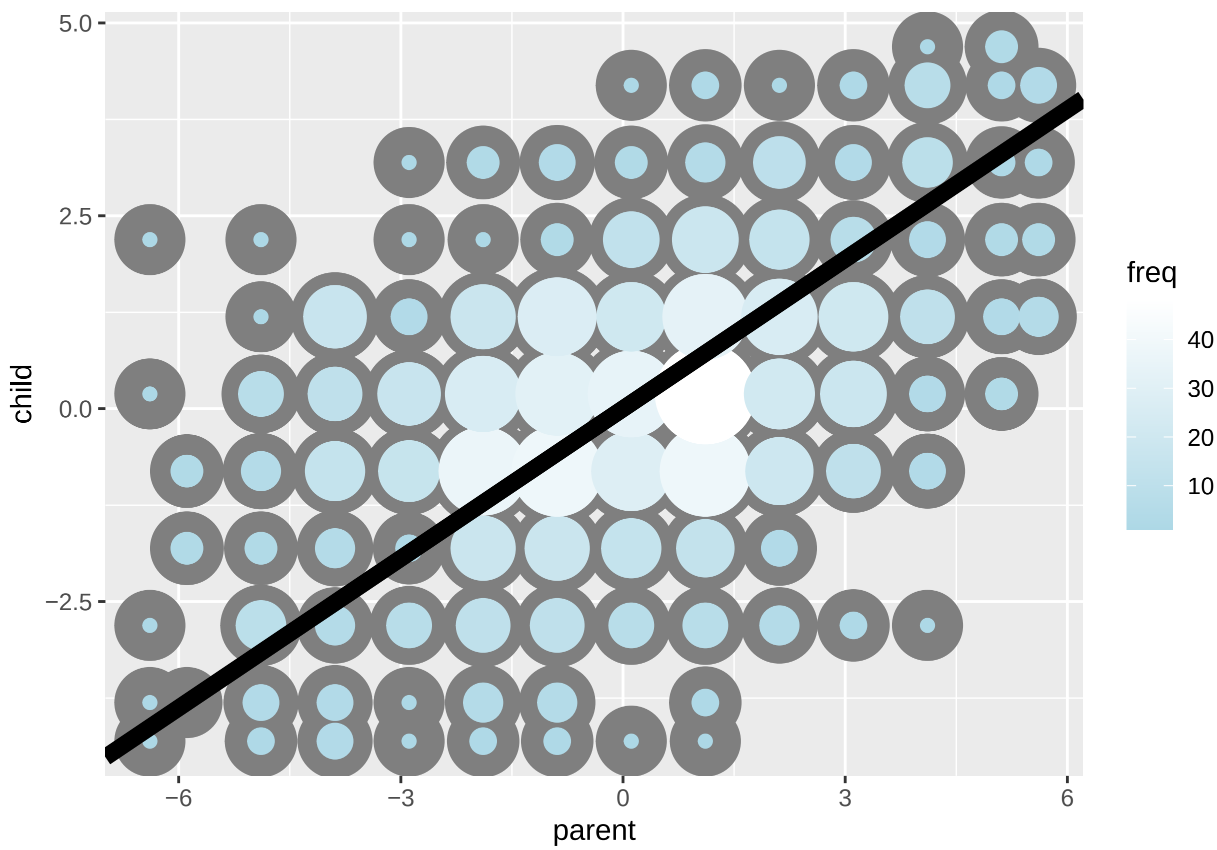


Regression Through the Origin

- Suppose that X_i are the parents' heights
- Consider picking the slope β that minimizes $\sum_{i=1}^n (Y_i - X_i \beta)^2$
- This is exactly using the orgin as a pivot point picking the line that minimizes the sum of squared vertical distances of the points to the line
- Subtract the means so that the orgin is the mean of the parent and children's heights + A plot with a regression line going through true (0,0) often doesn't make sense, so subtracting the means realigns the orgin to be in the middle of the data

```
freqData <- as.data.frame(table(galton$parent - mean(galton$parent),
                                galton$child - mean(galton$child)))
names(freqData) <- c("child", "parent", "freq")
freqData$child <- as.numeric(as.character(freqData$child))
freqData$parent <- as.numeric(as.character(freqData$parent))
plot <- ggplot(filter(freqData, freq > 0), aes(x = parent, y = child)) +
        scale_size(range = c(2, 20), guide = "none") +
        geom_point(colour = "grey50",
                   aes(size = freq + 20)) +
```

```r
       geom_point(aes(colour = freq, size = freq)) +
       scale_colour_gradient(low = "lightblue", high = "#FFFFFF") +
       geom_abline(intercept = 0,

               slope = lm(
                       I(child - mean(child)) ~
                               I(parent - mean(parent)) - 1,
                       data = galton)$coeff,

                   size = 3)
plot
```



- In the next few lectures we'll talk about why this is the solution

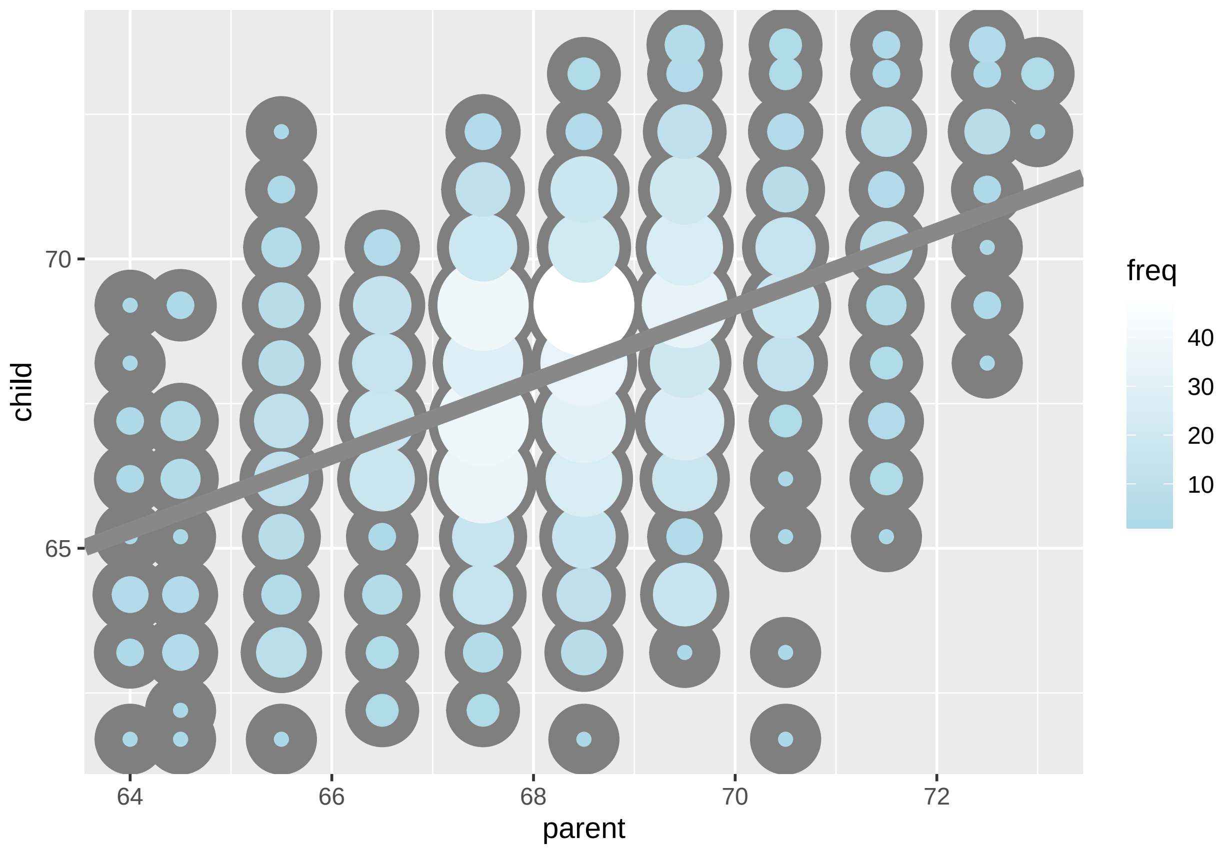
```r
lm(I(child - mean(child)) ~ I(parent - mean(parent)) - 1, data = galton)
```

```
##
## Call:
## lm(formula = I(child - mean(child)) ~ I(parent - mean(parent)) -
##     1, data = galton)
##
## Coefficients:
## I(parent - mean(parent))
##                   0.6463
```

- The `I` function just ignores the intercept, since we already adjusted for that

- We can also fit a line to an un-adjusted model

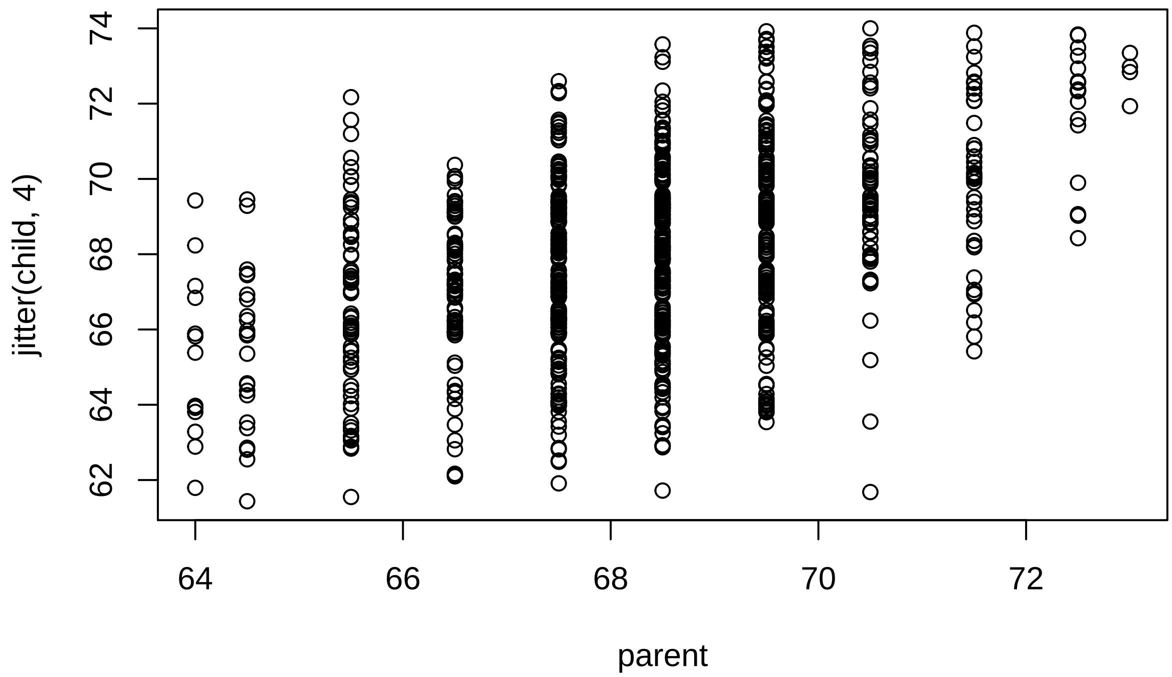
```
freqData <- as.data.frame(table(galton$child, galton$parent))
names(freqData) <- c("child", "parent", "freq")
freqData$child <- as.numeric(as.character(freqData$child))
freqData$parent <- as.numeric(as.character(freqData$parent))
plot <- ggplot(filter(freqData, freq > 0), aes(x = parent, y = child)) +
        scale_size(range = c(2, 20), guide = "none" ) +
        geom_point(colour = "grey50", aes(size = freq + 20)) +
        geom_point(aes(colour = freq, size = freq)) +
        scale_colour_gradient(low = "lightblue", high = "#FFFFFF")
lm1 <- lm(galton$child ~ galton$parent)
plot + geom_abline(intercept = coef(lm1)[1], slope = coef(lm1)[2],
                   size = 3, colour = "#888888")
```



Lesson with `swirl()`: Introduction

- Another way we could have gotten past overlapping plot points is to use the `jitter` function

```
plot(jitter(child,4) ~ parent, galton)
```



Linear Least Squares

- Also called **Ordinary Least Squares (OLS)**; it fits a line through some data.

Notation and Background

Notation

- The empirical mean is defined as
  $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$

- If we subtract the mean from data points, we get data that has a mean of 0. That is, if we define:
  $\tilde{X}_i = X_i - \bar{X}$.
  + The mean of $\tilde{X}_i$ is 0

- This process is called "**centering**" the random variables

- Recall from the previous lecture that the mean is the elast squares solution for minimizing $\sum_{i=1}^{n}(X_i - \mu)^2$

The Emprical Standard Deviation adn Variance

- Define the empirical variance as
  $S^2 = \frac{1}{n-1}\sum_{i=1}^n (X_i - \bar{X})^2 = \frac{1}{n-1}(\sum_{i=1}^n X_i^2 - n\bar{X}^2)$

- The empirical standard deviation is defined as $S = \sqrt{S^2}$.
  + Notice that the standard deviation has the same units as the data.

- The data defined by $\frac{X_i}{s}$ have an empirical standard deviation of 1. + This is called "**scaling**" the data.

Normalization

- The data defined by
  $Z_i = \frac{X_i - \bar{X}}{s}$
  have an empirical mean of 0 and an empirical standard deviation of 1.

- The process of centering then scaling the data is called "**normalizing**" the data.

- Normalized data are centered at 0 and have units equal to standard deviations of the original data.

- For example, a value of 2 from normalized data is saying that data point was two standard deviations larger than the mean.

The Empirical Covariance

- Consider now when we have pairs of data, (X_i, Y_i)

- Their empirical covariance is
  $Cov(X, Y) = \frac{1}{n-1}\sum_{i=1} n(X_i - \bar{X})(Y_i - \bar{Y})$
  $= \frac{1}{n-1}(\sum_{i=1}^n X_i Y_i - n\bar{X}\bar{Y})$

- The correlation is defined as
  $Cor(X, Y) = \frac{Cov(X,Y)}{S_x S_y}$
  + Where S_x and S_y are the estimates of standard deviaitons for the X observations and Y observations, respectively.

Some Facts About Correlation

- $Cor(X, Y) = Cor(Y, X)$

- $-1 \leq Cor(X, Y) \leq 1$

- $Cor(X, Y) = 1$ and $Cor(X, Y) = -1$ only when the X or Y observations fall perfectly on a positive or negative sloped line, repectively.

- $Cor(X, Y)$ measures the strength of the linear relationship between the X and Y data, with stronger relationships as $Cor(X, Y)$ heads towards either -1 or 1 {
- $Cor(X, Y) = 0$ implies no linear relationship

Linear Least Squares

Fitting the Best Line

- Let Y_i be the i^{th} child's height and X_i be the i^{th} (average over the pair of) parents' heights.

- Consider finding the best line
  + Child's Height = β_0 + Parent's Height * β_1
  $\sum_{i=1}^{n} Y_i - (\beta_0 + \beta_1 X_i)^2$
- the least squares model fit to the line $Y = \beta_0 + \beta_1 X$ through the data pairs (X_i, Y_i) with Y_i as the outcome obtains the line $Y = \hat{\beta}_0 + \hat{\beta}_1 X$ where
  $\hat{\beta}_1 = Cor(Y, X)\frac{Sd(Y)}{Sd(X)}$
  $\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X}$

- $\hat{\beta}_1$ has the units of Y/X, $\hat{\beta}_0$ has the units of Y.

- The line passes through the point $(\bar{X}, \bar{Y})$

- The slope of the regression line with X as the outcome and Y as the predictor is $\frac{Cor(Y,X)Sd(X)}{Sd(Y)}$

- The slope si the same one you would get if you centered the data, $(X_i - \bar{X}, Y_i - \bar{Y})$, and made a regression through the orgin

- If you normalized the data, $(\frac{X_i - \bar{X}}{Sd(X)}, \frac{Y_i - \bar{Y}}{Sd(Y)})$, the slope is $Cor(Y, X)$

Linear Least Squares Coding Example

```
y <- galton$child
x <- galton$parent
beta1 <- cor(y,x) * sd(y) / sd(x)
beta0 <- mean(y) - beta1 * mean(x)

#Showing the computations by hand are the same as coef from lm function
rbind(c(beta0, beta1), coef(lm(y~x)))
```

```
##      (Intercept)         x
## [1,]    23.94153 0.6462906
## [2,]    23.94153 0.6462906
```

- `lm` stands for *linear model*

```
#The slope is the same in centered data
yc <- y - mean(y)
xc <- x - mean(x)
beta1 <- sum(yc * xc) / sum(xc^2)
c(beta1, coef(lm(y ~ x))[2])
```

```
##                   x
## 0.6462906 0.6462906
```

```
lm(yc ~ xc - 1)$coef #minus 1 gets rid of intercept
```

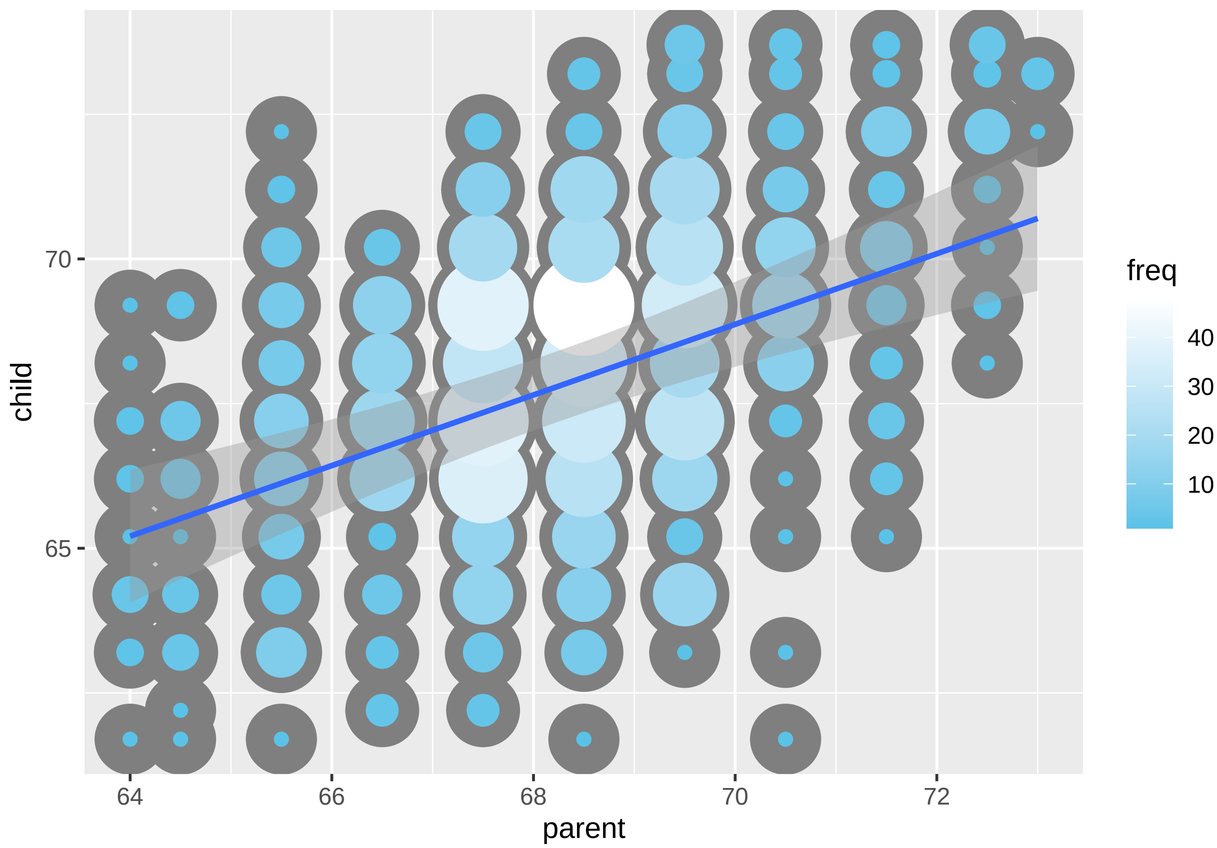
```
##        xc
## 0.6462906
```

```
#Normalizing variables results in the slope being the correlation
yn <- (y - mean(y))/sd(y)
xn <- (x - mean(x))/sd(x)
results <- cbind(cor(y,x), lm(yn ~ xn)$coef[2], cor(yn, xn))
colnames(results) <- c("cor(y,x)", "Slope(yn ~ xn)", "cor(yn, xn)")
results
```

```
##     cor(y,x) Slope(yn ~ xn) cor(yn, xn)
## xn 0.4587624      0.4587624   0.4587624
```

Adding a Linear Regression to ggplot

```
plot <- ggplot(filter(freqData, freq > 0), aes(parent, child)) +
        scale_size(range = c(2, 20), guide = "none") +
        geom_point(colour = "grey50", aes(size = freq + 20)) +
        geom_point(aes(colour = freq, size = freq)) +
        scale_colour_gradient(low = "#5BC2E7", high = "#FFFFFF")

#Adding smoother
#y ~ x is assumed if not given
plot + geom_smooth(method = "lm", formula = y ~ x)
```



- A confidence interval is also given around the line automaticly

Technical Details

Brian Caffo discusses the proof for least squares regression beta_1 value in this video

Lesson with `swirl()`: Least Squares Estimation

(No new content)

Regression to the Mean

Regression to the Mean

- $P(Y < x|X = x)$ gets bigger as x tends towards very large values.
  + Similarly $P(Y > x|X = x)$ gets bigger as x tends towards very small values.

- Regression line is like the intrisic part of this relation
  + Unless $Cor(Y, X) = 1$ the intrinsic part isn't perfect

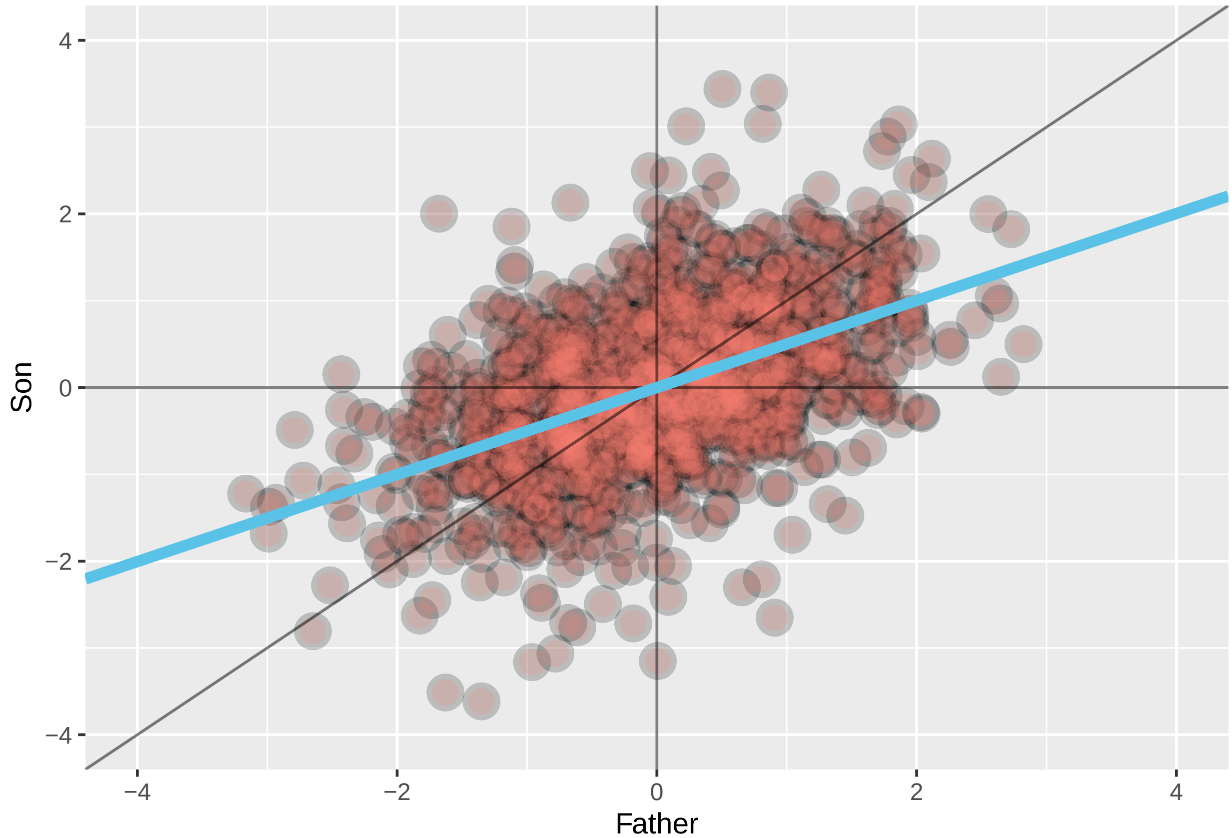
- Suppose we center X (child's hieght) and Y (parent's height) so that they both have a mean of 0
  + Then, recall, our regression line passes through $(0, 0)$

- We then normalize the data points too
  + The slope of the regression line is $Cor(Y, X)$, regardless of which variable is the outcome (since both sds are 1)

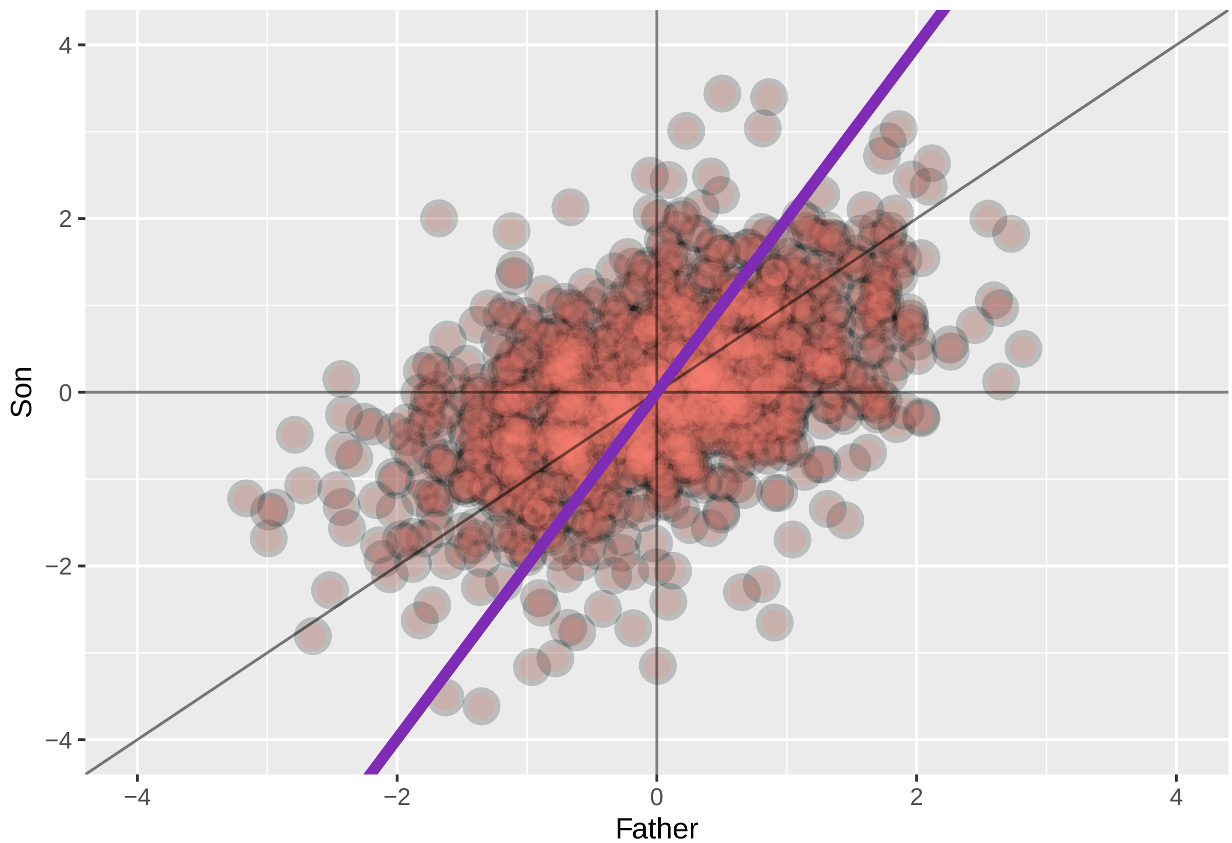
- If the outcome is plotted on the horizontal axis the slope of the least squares line will be $\frac{1}{Cor(Y,X)}$

Plotting the Regression Implicitly

```
library(UsingR); data(father.son)
y <- father.son$sheight
x <- father.son$fheight
y <- (y - mean(y)) / sd(y)
x <- (x - mean(x)) / sd(x)
rho <- cor(x, y) #rho is std greek letter for correlations
plot <- ggplot(data.frame(Father = x, Son = y), aes(Father, Son)) +
        geom_point(size = 6, colour = "#000000", alpha = 0.2) +
        geom_point(size = 4, colour = "salmon", alpha = 0.2) +
        xlim(-4,4) +
        ylim(-4,4) + #Std. norm being +/- 4 is very unlikely
        geom_abline(intercept = 0, slope = 1, alpha = 0.5) +
        geom_vline(xintercept = 0, alpha = 0.5) +
        geom_hline(yintercept = 0, alpha = 0.5)
plot + geom_abline(intercept = 0, slope = rho, size = 2, colour = "#5BC2E7")
```




```
plot + geom_abline(intercept = 0, slope = 1/rho, size = 2, colour = "#7E2CB5")
```



* The blue line is where the Father's height is the predictor and the Son's height is the outcome
* The purple line is where the Son's hieght is the predictor and the Father's height is the outcome (`1/rho` because the outcome is on the horizontal axis)

Lesson with `swirl()`: Residuals

- A residual is the distance between the actual data point and the regression line.
  + I've previously heard it also called the "Unexplained Variation" since the distance form the mean value to data point is the "Total Variation (from the mean)", then the distance from the mean to reg. line is the "Explained Variation".

- You can get some info on a data sets residuals by calling `summary` on the results of `lm` as seen below

```
summary(lm(child ~ parent, galton))
```

```
##
## Call:
## lm(formula = child ~ parent, data = galton)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.8050 -1.3661  0.0487  1.6339  5.9264
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 23.94153    2.81088   8.517   <2e-16 ***
## parent       0.64629    0.04114  15.711   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.239 on 926 degrees of freedom
## Multiple R-squared:  0.2105, Adjusted R-squared:  0.2096
## F-statistic: 246.8 on 1 and 926 DF,  p-value: < 2.2e-16
```

- `est` will return the estimate, $\hat{y}$
- `sqe` will calculate the sum of the squared residuals, also called the Residual Sum of Squares
- var(residuals) = var(data) - var(estimate)
  + As such the variance of residuals is always less than the variance of data
- The residuals shouldn't be correlated to either factor, if it did this may imply a diffrent relationship is present

Quiz 1

1. Given...

```r
x <- c(0.18, -1.54, 0.42, 0.95)
w <- c(2, 1, 3, 1)
```

Give the value of μ that minimizes the least squares equation $\sum_{i=1} nw_i(x_i - \mu)^2$

```r
sum(w * x) / sum(w)
```

```
## [1] 0.1471429
```

2. Given...

```r
x <- c(0.8, 0.47, 0.51, 0.73, 0.36, 0.58, 0.57, 0.85, 0.44, 0.42)
y <- c(1.39, 0.72, 1.55, 0.48, 1.19, -1.59, 1.23, -0.65, 1.49, 0.05)
```

Fit the regression through the orgin and get the slope treating y as the outsome and x as the regressor.

```r
lm(y ~ x - 1)$coef
```

```
##         x
## 0.8262517
```

3. Do `data(mtcars)` from the datasets package and fit the regression model with mpg as the outcome and weight as the predictor. Give the slope coefficient.

```r
data(mtcars)
lm(mpg ~ wt, mtcars)$coef
```

```
## (Intercept)          wt
##   37.285126   -5.344472
```

4. Consider data with an outcome (Y) and a predictor (X). The standard deviation of the predictor is one half that of the outcome. The correlation between the two variables is 0.5. What value would the slope coefficient for the regression model with Y as the outcome and X as the predictor?

```
0.5 * 2/1
```

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

5. Students were given two hard tests and scores were normalized to have empirical mean 0 and variance 1. The correlation between the scores on the two tests was 0.4. What would be the expected score on Quiz 2 for a student who had a normalized score of 1.5 on Quiz 1?

```
beta1 <- 0.4 * 1/1
beta0 <- 0 - beta1*0
yhat <- beta0 + beta1*1.5
yhat
```

```
## [1] 0.6
```

6. Given...

```
x <- c(8.58, 10.46, 9.01, 9.64, 8.86)
```

What is the value of the first measurement if x were normalized?

```
xn <- (x-mean(x))/sd(x)
xn[1]
```

```
## [1] -0.9718658
```

7. Given...

```
x <- c(0.8, 0.47, 0.51, 0.73, 0.36, 0.58, 0.57, 0.85, 0.44, 0.42)
y <- c(1.39, 0.72, 1.55, 0.48, 1.19, -1.59, 1.23, -0.65, 1.49, 0.05)
```

What is the intercept for fitting the model with x as the predictor and y as the outcome?

```
lm(y ~ x)$coef
```

```
## (Intercept)           x
##    1.567461   -1.712846
```

8. You know that both the predictor and response have mean 0. What can be said about the intercept when you fit a linear regression?

- The intercept is the orgin

9. Given...

```
x <- c(0.8, 0.47, 0.51, 0.73, 0.36, 0.58, 0.57, 0.85, 0.44, 0.42)
```

What value minimizes the sum of the squared distances between these points and itself?

```r
mean(x)
```

```
## [1] 0.573
```

10. Let the slope having fit Y as the outcome and X as the predictor be denoted as β_1. Let the slope from fitting X as the outcome and Y as the predictor be denoted as γ_1. Suppose that you divide β_1 by γ_1 What is this ratio always equal to?

- $\beta_1 = Cor(Y,X)\frac{sd(Y)}{sd(X)}$
- $\gamma_1 = Cor(Y,X)\frac{sd(X)}{sd(Y)}$
- $\frac{\beta_1}{\gamma_1} = \frac{Cor(Y,X)*sd(Y)/sd(X)}{Cor(Y,X)*sd(X)/sd(Y)} = \frac{sd(Y)*sd(Y)}{sd(X)*sd(X)} = \frac{Var(Y)}{Var(X)}$

Linear Regression & Multivariable Regression

Statistical Linear Regression Models

Statistical Linear Regression Models

Basic Regression Model with Additive Gaussian Errors

- Consider developing a probabilistic model for linear regression
  $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$
  + Here the ϵ_i are assumed iid $N(0, \sigma^2)$
  - Can be thought of as accumulated variables that aren't modeled by act on the response as iid gaussian errors + $E[Y_i | X_i = x_i] = \mu_i = \beta_0 + \beta_1 x_i$
  + $Var(Y_i | X_i = x_i) = \sigma^2$

Interpreting Coefficients

Intercept

- β_0 is the expected value of the response when the predictor is 0
  $E[Y|X=0] = \beta_0 + \beta_1 \times 0 = \beta_0$
  + This isn't always a value of interest, for example when $X = 0$ is impossible (x represents weight) or far outside of the range of data.

- A solution to non-interpretable intercepts is to shift the equation by some value, a then define a new intercept, $\tilde{\beta_0}$.
  $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i = \beta_0 + a\beta_1 + \beta_1(X_i - a) + \epsilon_i = \tilde{\beta_0} + \beta_1(X_i - a) + \epsilon_i$
  + Shifting your X values by value a changes the intercept, but not the slope.
  + Often a is set to $\bar{X}$ so that the intercept is inteerpretted as the expected response at the average X value.

Slope

- β_1 is the expected change in response for a 1 unit change in the predictor

- Consider the impact of changing the units of X.
  $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i = \beta_o + \frac{\beta_1}{a}(X_i a) + \epsilon_i = \beta_0 + \tilde{\beta_1}(X_i a) + \epsilon_i$
  + Since β_1 is in units of Y/X we divide by the factor, a, that we're multiplying with X_i.

- Example: X is height in m and Y is weight in kg. Then β_1 is kg/m. Converting X to cm implies multiplying X by $100 cm/m$. To get β_1 in the right units, we have to divide by $100 cm/m$ to get it to have the right units.
  $Xm \times \frac{100cm}{m} = (100X)cm$ and $\beta_1 \frac{kg}{m} \times \frac{1m}{100cm} = (\frac{\beta_1}{100}) \frac{kg}{cm}$

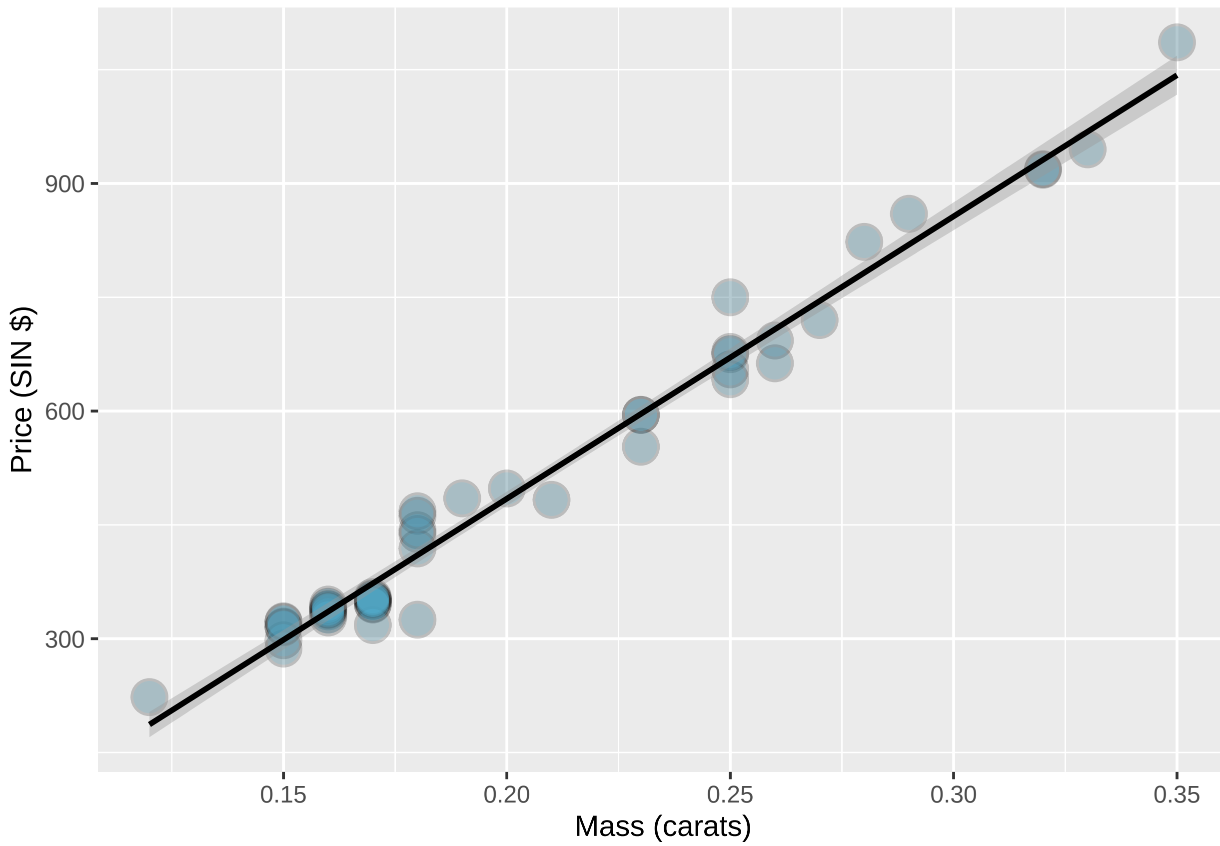
Linear Regression for Prediction

- We can get a prediction for Y, $\hat{y}$ by plugging in the X that we want into our model
  $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_i$

Example using diamond Data

- The data in this example is diamond prices (in Sigapore dollars) and diamond weight in carats (1 carat = 0.2 g).

```r
library(UsingR); data(diamond); library(tidyverse)
plot <- ggplot(diamond, aes(carat, price)) +
        xlab("Mass (carats)") +
        ylab("Price (SIN $)") +
        geom_point(size = 6, colour = "#000000", alpha = 0.2) +
        geom_point(size = 5, colour = "#5BC2E7", alpha = 0.2)
plot + geom_smooth(method = "lm", colour = "#000000", formula = y ~ x)
```



Creating a Model

```
# Fitting the linear regression model
fit <- lm(price ~ carat, data = diamond)
coef(fit)
```

```
## (Intercept)       carat
##   -259.6259   3721.0249
```

- We estimate an expected 3721.02 (SIN) dollar increase in price for every increase of 1 carat in mass of diamonds.
- The intercept, -259.63 is the expected price of a 0 carat diamond, which doesn't make sense to interpret.
  + As such we'll mean center our reg. line
  #### Centering Model on the Mean

```
cfit <- lm(price ~ I(carat - mean(carat)), data = diamond)
cfit$coef
```

```
##             (Intercept) I(carat - mean(carat))
##                500.0833              3721.0249
```

- To do arithmetic operations in the formula in `lm` you have to surround the operation with the `I` function

- The slope has not changed

- The intercept has changed to 500, the expected price for the average sized diamond of the data (0.204 carats).

Changing Units in the Model

- Change unit to 1/10 of a carrat

```
tenthfit <- lm(price ~ I(carat * 10), data = diamond)
coef(tenthfit)
```

```
##   (Intercept) I(carat * 10)
##     -259.6259      372.1025
```

- So now the slope is interpretted as a 372.1 dollar increase for every additional 0.1 carrats of diamond.

Estimating a Value

```
newDiamonds <- c(0.16, 0.27, 0.34)
#Computing manually
fit$coef[1] + fit$coef[2] * newDiamonds
```

```
## [1]  335.7381  745.0508 1005.5225
```

```
#Using predict function
results <- predict(fit, newdata = data.frame(carat = newDiamonds))
names(results) <- as.character(newDiamonds) #renaming not required
results
```

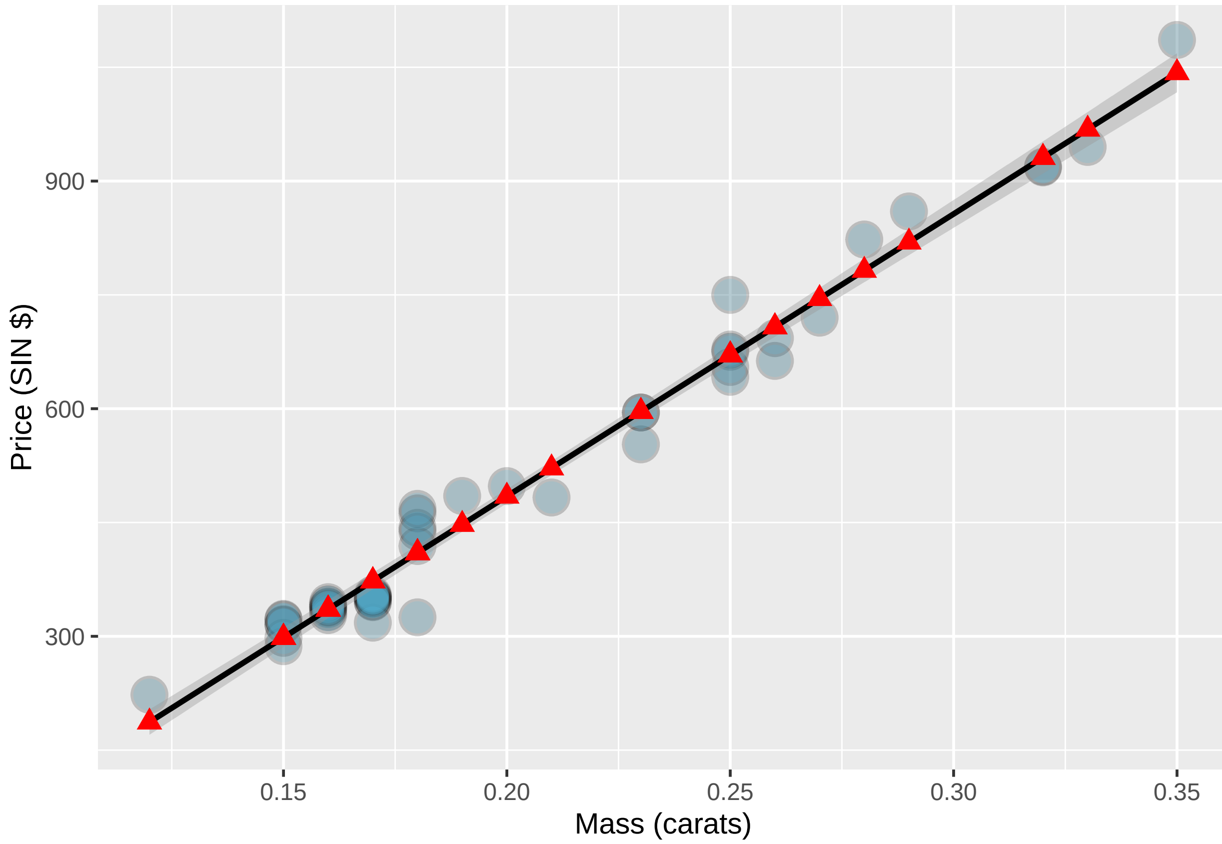
```
##      0.16      0.27      0.34
##  335.7381  745.0508 1005.5225
```

```
#Using predict without `newdata` will return y-hat for given x values
predict(fit)
```

```
##         1         2         3         4         5         6         7         8
##  372.9483  335.7381  372.9483  410.1586  670.6303  335.7381  298.5278  447.3688
##         9        10        11        12        13        14        15        16
##  521.7893  298.5278  410.1586  782.2611  335.7381  484.5791  596.2098  819.4713
##        17        18        19        20        21        22        23        24
##  186.8971  707.8406  670.6303  745.0508  410.1586  335.7381  372.9483  335.7381
##        25        26        27        28        29        30        31        32
##  372.9483  410.1586  372.9483  410.1586  372.9483  298.5278  372.9483  931.1020
##        33        34        35        36        37        38        39        40
##  931.1020  298.5278  335.7381  335.7381  596.2098  596.2098  372.9483  968.3123
##        41        42        43        44        45        46        47        48
##  670.6303 1042.7328  410.1586  670.6303  670.6303  298.5278  707.8406  298.5278
```

```
plot + geom_smooth(method = "lm", colour = "#000000", formula = y ~ x) +
        geom_point(aes(y = as.numeric(predict(fit))),
```

```
                    size = 3, color = "#FF0000", shape = 17)
```



Residuals

Residuals

- The residuals are the variation from the regression line, that is left unexplained by our model, $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$.
- Observed outcome i is Y_i at predictor value X_i
- Predicted outcome i is $\hat{Y}_i$ at predictor value X_i is $\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_i$
- Residual, e_i, is the difference between the observed and predicted outcome: $e_i = Y_i - \hat{Y}_i$. + This is the vertical distanc ebetween the observed data point and the regression line
- Least squares minimizes these residuals, the equation $\sum_{i=1}^{n} e_i^2$
- The e_i can be thought of as estimates of the ϵ_i

Properties of the Residuals

- $E[e_i] = 0$

- If an intercept is included, $\sum_{i=1}^n e_i = 0$

- If a regressor variable, X_i, is included in the model $\sum_{i=1}^n e_i X_i = 0$

- Residuals are useful for investigating poor model fit
  + Residual plots can highlight these poor fits

- Residuals can be though of as the outcome (Y) with the linear association of the predictor (X) removed.

- One differentiates residual variation (variation after removing the predictor) from systematic variation (variation explained by the regression model).

Residuals, Coding Example

- Using diamond dataset again

```
data("diamond")
y <- diamond$price
x <- diamond$carat
fit <- lm(y ~ x)

e <- resid(fit) #Getting residuals

yhat <- predict(fit)

# Showing residuals are the same as y - yhat (within a floating point error)
max(abs(e - (y - yhat)))
```

```
## [1] 5.258016e-13
```

```
# And again, but manually entering the equation for yhat
max(abs(e - (y - (coef(fit)[1] + coef(fit)[2] * x))))
```

```
## [1] 5.258016e-13
```

```
#Showing sum of resid and resid*x are both 0
sum(e)
```

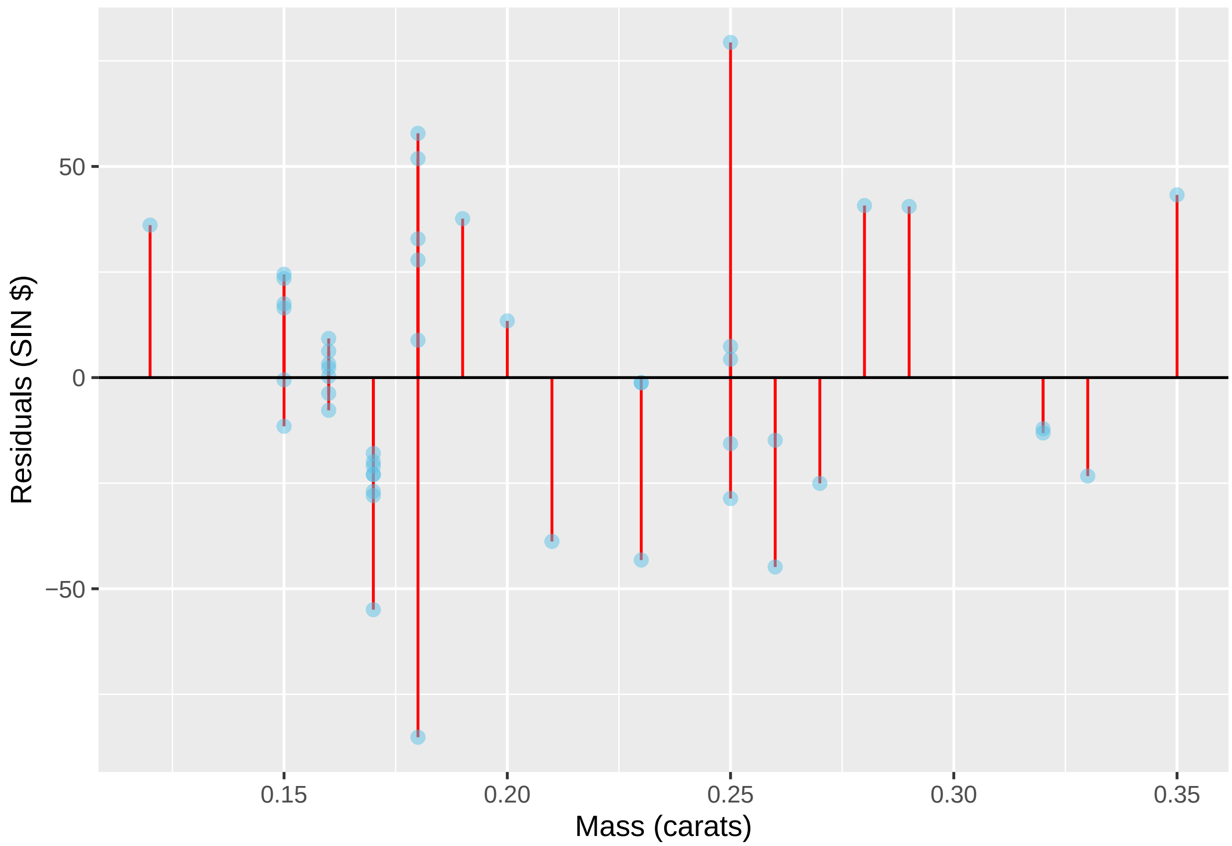
```
## [1] -3.93019e-14
```

```
sum(e * x)
```

```
## [1] -1.249001e-15
```

```
#Plotting the residuals
plot <- ggplot(data.frame(x = x, y = y, resid = e), aes(x, resid)) +

        geom_segment(aes(xend = x, yend = 0), colour = "#FF0000") +
```

```
        geom_point(size = 2, colour = "#5BC2E7", alpha = 0.5) +
        xlab("Mass (carats)") +
        ylab("Residuals (SIN $)") +
        geom_hline(yintercept = 0, color = "#000000")
plot
```



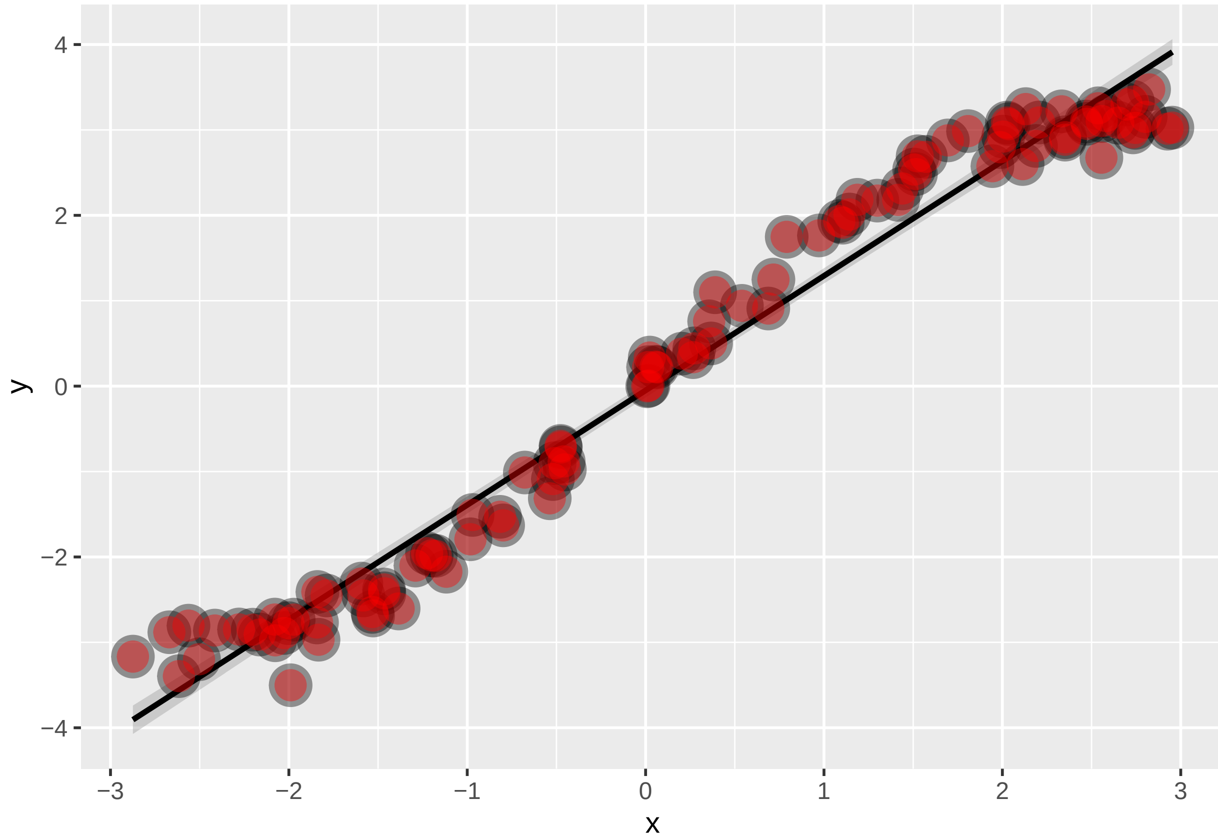
Using Residual Plot to Detect a Poorly Fit Model

- We're going to generate some data that looks linear but actually has an underlying relation to it that will become more apparent after plotting the residuals

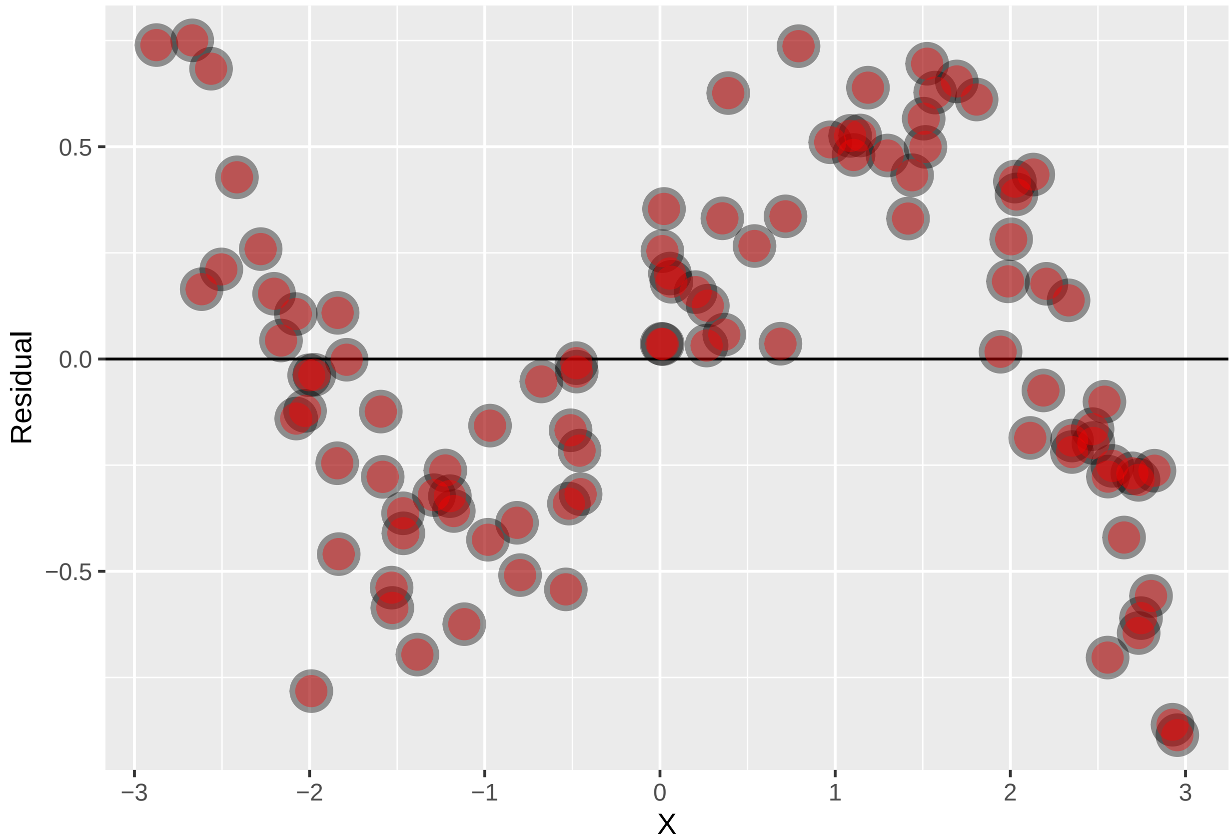
```
set.seed(1618033)
x <- runif(100, -3, 3)
y <- x + sin(x) + #Y is related with sin(x), lm will expose the sin(x) rel.
        rnorm(100, sd = .2) # For noise
plot <- ggplot(data.frame(x = x, y = y), aes(x,y)) +
        geom_smooth(method = "lm", colour = "#000000") +
        geom_point(size = 7, colour = "#000000", alpha = 0.4) +
        geom_point(size = 5, colour = "#FF0000", alpha = 0.4)

residplot <- ggplot(data.frame(x = x, resid = resid(lm(y ~ x))),
              aes(x, resid)) +
```

```
        geom_hline(yintercept = 0) +
        geom_point(size = 7, colour = "#000000", alpha = 0.4) +
        geom_point(size = 5, colour = "#FF0000", alpha = 0.4) +
        labs(x = "X", y = "Residual")
plot
```

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




```
residplot
```

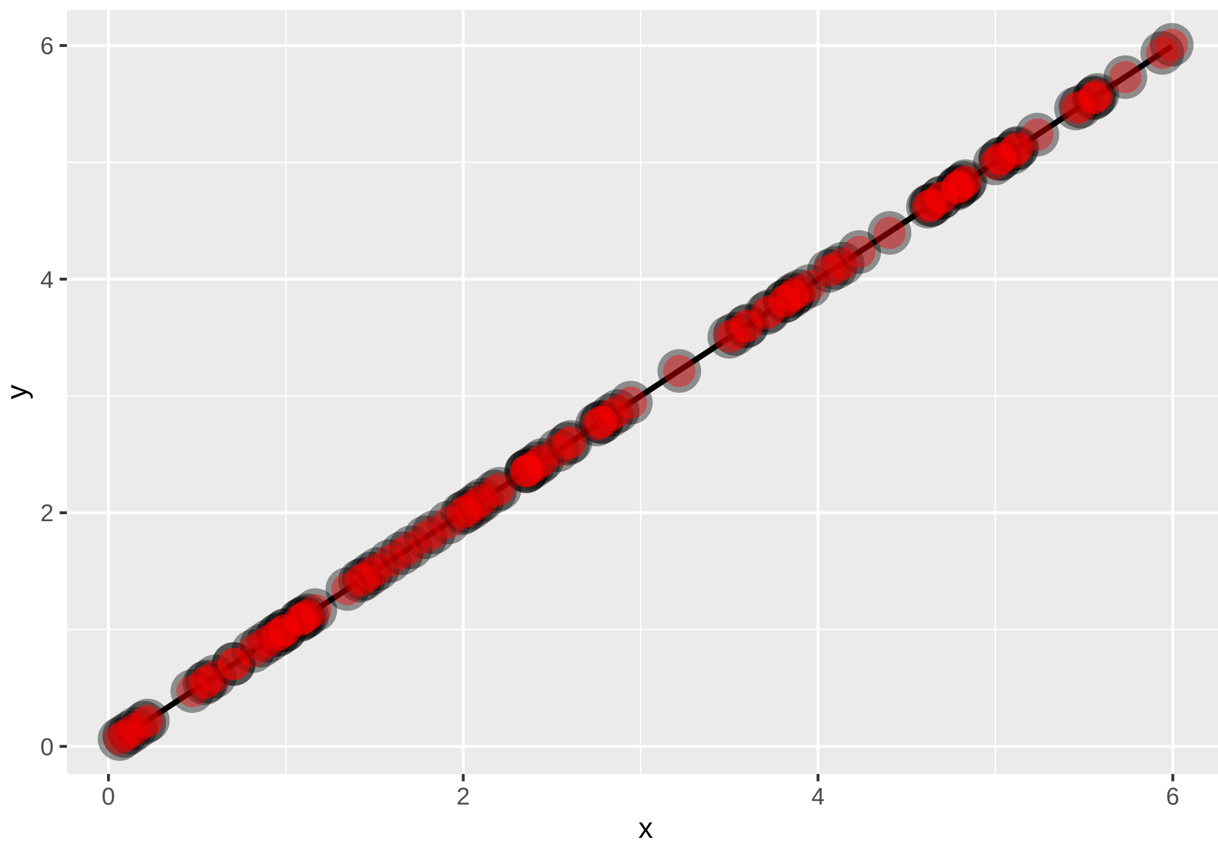


- A secondary pattern can be seen in the residual plot, indicating there might be a better model than a line.

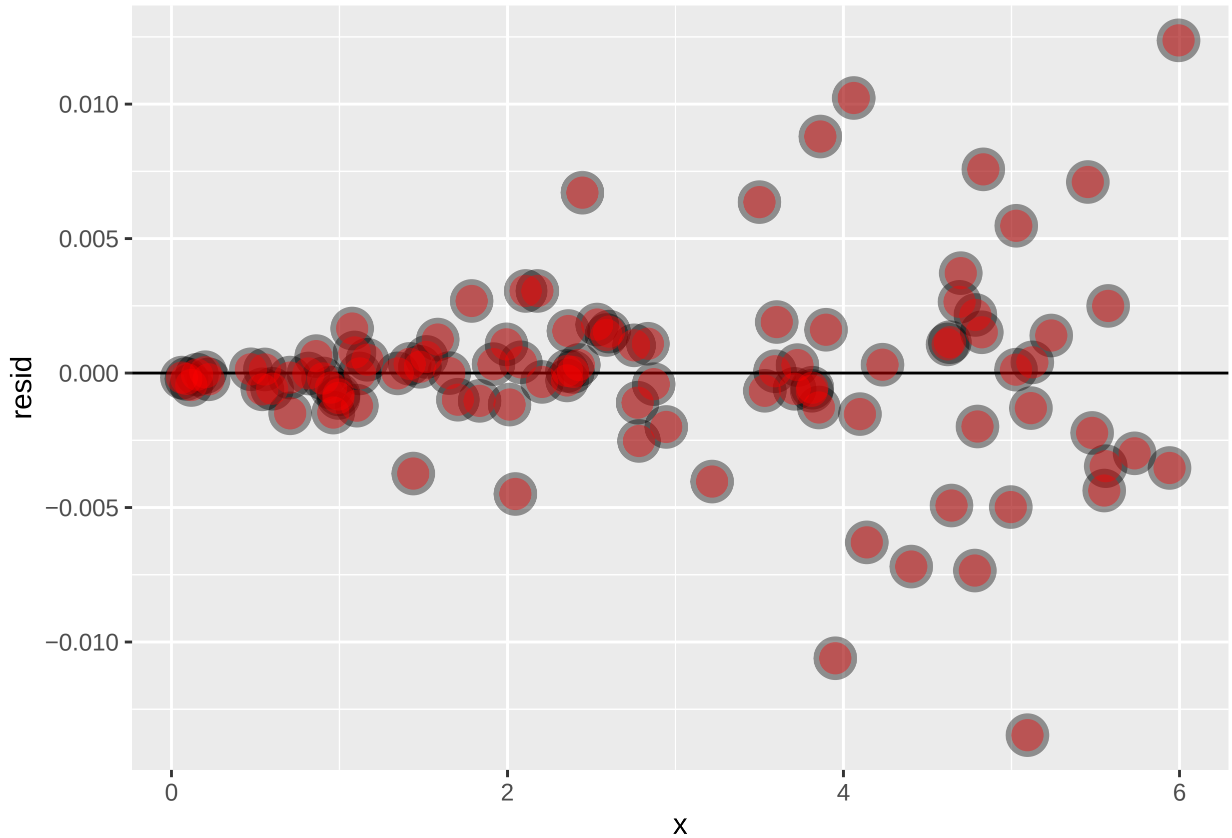
Detecting Heteroskedasticity with a Residual Plot

```
x <- runif(100, 0, 6)
y <- x + rnorm(100, mean = 0, sd = 0.001 * x)#sd increases as x increases
plot <- ggplot(data.frame(x = x, y = y), aes(x,y)) +
        geom_smooth(method = "lm", colour = "black") +
        geom_point(size = 7, colour = "#000000", alpha = 0.4) +
        geom_point(size = 5, colour = "#FF0000", alpha = 0.4)
residplot <- ggplot(data.frame(x = x, resid = resid(lm(y ~ x))),
                    aes(x,resid)) +
        geom_hline(yintercept = 0, colour = "#000000") +
        geom_point(size = 7, colour = "#000000", alpha = 0.4) +
        geom_point(size = 5, colour = "#FF0000", alpha = 0.4)
plot
```

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




```
residplot
```



* The plot looks linear, but plotting the residuals reveals an underlying pattern

Residual Variance

Estimating Residual Variaiton

- Model: $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$
- The mean linear estimate of σ^2 is $\frac{1}{n}\sum_{i=1}^n e_i^2$, the average squared residual
- Most people use:
  $\hat\sigma^2 = \frac{1}{n-2}\sum_{i=1}^n e_i^2$
  + with $n-2$ instead of n so that $E[\hat\sigma^2] = \sigma^2$

Diamond Example

```
y <- diamond$price
x <- diamond$carat
n <- length(y)

#Solving resid s.d. implicitly
sqrt(sum(resid(fit)^2) / (n - 2))
```

```
## [1] 31.84052
```

```r
#Getting resid deviation with functions
fit <- lm(y ~ x)
summary(fit)$sigma
```

```
## [1] 31.84052
```

```r
#You can see the value in the summary print out here:
summary(fit)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -85.159 -21.448  -0.869  18.972  79.370
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -259.63      17.32  -14.99   <2e-16 ***
## x            3721.02      81.79   45.50   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.84 on 46 degrees of freedom
## Multiple R-squared:  0.9783, Adjusted R-squared:  0.9778
## F-statistic:  2070 on 1 and 46 DF,  p-value: < 2.2e-16
```

Summarizing Variation

- **Total Variability** - the variability around an intercept (mean only regression) + $\sum_{i=1}^{n}(Y_i - \bar{Y})^2$
  + Sum of Regression & Error Variability

- **Regression Variability** - the variability that is explained by adding the predictor
  + $\sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2$

- **Error Variability** - what's leftover around the regression line
  + $\sum_{i=1}^{n}(Y_i - \hat{Y})^2$

R Squared, the Coefficent of Determination

- R squared is the percentage of the total variability that is explained by the linear relationship with the predictor
  $R^2 = \frac{\sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}$

- R^2 is the percentage of variation explained by the regression model

- $0 \leq R^2 \leq 1$

- R^2 is the sample correlation squared

- R^2 can bea misleading summary of model fit
  + Deleting data can inflate R^2
  + (For later,) Adding terms to a regression model always increases R^2

- Execute `example(anscombe)` to see the following data:
  + Basically same mean and variance of X and Y
  + Identical correlations (hence the same R^2 value)
  + Same linear regression relationship

Lesson with `swirl()`: Residual Variation

- `deviance` will calculate the sum of the squares of a `lm`

Inference in Regression

Inference in Regression

Recall Our Model and Fitted Values

- Model:
  + $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$
  + $\epsilon \sim N(0, \sigma^2)$, an error term
  + $\hat{\beta}_1 = Cor(Y, X)\frac{Sd(Y)}{Sd(X)}$
  + $\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X}$

- We assume that the true model is known for most of this course

Review Some Statistical Inference Concepts

- Statistics like $\frac{\hat{\theta}-\theta}{\hat{\sigma}_{\hat{\theta}}}$ often have the following properties:
  + Is normally distributed and ahs a finite sample Student's T distribution if the estimated variance is repalced with a sample estimate (under normality assumptions).
  + Can be used to test $H_0 : \theta = \theta_0$ versus $H_a : \theta >, <, \neq \theta_0$
  + Can be used to create a confidence interval for θ via $\hat{\theta} \pm Q_{1-\alpha/2}\hat{\sigma}_{\hat{\theta}}$ where $Q_{1-\alpha/2}$ is the relevant quantile from either a normal or T distribution

- In the case of regression with iid sampling assumptions and normal errors, out inferences will follow very similarily to what was discussed in the inference class.

- Under assumptions on the ways in which the X values are collected the iid sampling model, and mean model, the nromal results hold to create intervals and confidence intervals

Explanation

- Variance of our regression slope, $\sigma^2_{\hat\beta_1}$, tells both how variable points are around the regression line, σ^2, and how variable the points are from the mean
  $\sigma^2_{\hat\beta_1} = Var(\hat\beta_1) = \sigma^2 / \sum_{i=1}^n (X_i - \bar X)^2$
  + This implies spreaded out points will give a lower variance for a slope
  - Thus large cluster of points very far apart would give the best variance, although this lm would assume the uncollected data between the clusters is linear
- Variance of the intercept, $\sigma^2_{\hat\beta_0}$, is less informative but still can provide some information.
  $\sigma^2_{\hat\beta_0} = Var(\hat\beta_0 = (\frac{1}{n} + \frac{\bar X^2}{\sum_{i=1}^n (X_i - \bar X)^2})\sigma^2$
- In both these cases, in practice, σ is replaced by its estimate
- Under iid gaussian errors, $\frac{\hat\beta_j - \beta_j}{\sigma \hat{}\beta_j}$, follows a t distribution with $n-2$ degrees of freedom and a normal distribution for large n
  + This can be used to create confidence intervals and perform hypothesis tests.

Coding Example

- Showing R is calculating all these values as we have given

```
library(UsingR); data(diamond)
y <- diamond$price
x <- diamond$carat
n <- length(y)
beta1 <- cor(y, x) * sd(y) / sd(x) #Slope
beta0 <- mean(y) - beta1 * mean(x) #y-intercept
e <- y - (beta0 + beta1 * x) #resids
sigma <- sqrt(sum(e^2) / (n-2)) #est. sd for resids
ssx <- sum((x - mean(x))^2) #Numerator of variance calculation
seBeta0 <- sqrt((1 / n + mean(x)^2 / ssx)) * sigma #s.e. of intercept
seBeta1 <- sigma / sqrt(ssx) #s.e. of slope
tBeta0 <- beta0 / seBeta0 #t statistic for intercept; H_0: beta0=0
tBeta1 <- beta1 / seBeta1 # t statistic for slope
#Relevant p values
pBeta0 <-  2 * pt(abs(tBeta0), df = n - 2, lower.tail = FALSE)
pBeta1 <-  2 * pt(abs(tBeta1), df = n - 2, lower.tail = FALSE)
coefTable <- rbind(c(beta0, seBeta0, tBeta0, pBeta0), c(beta1, seBeta1, tBeta1, pBeta1))
colnames(coefTable) <- c("Estimate", "Std. Error", "t value", "P(>|t|)")
rownames(coefTable) <- c("(Intercept)", "x")
coefTable
```

```
##              Estimate Std. Error   t value     P(>|t|)
## (Intercept) -259.6259   17.31886 -14.99094 2.523271e-19
## x           3721.0249   81.78588  45.49715 6.751260e-40
```

```r
summary(lm(y ~ x))$coef
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) -259.6259   17.31886 -14.99094 2.523271e-19
## x           3721.0249   81.78588  45.49715 6.751260e-40
```

Generating Confidence Intervals

```r
fit <- lm(y ~ x)
sumCoef <- summary(fit)$coef

#Intercept
sumCoef[1, 1] + c(-1, 1) * qt(0.975, df = fit$df) * sumCoef[1, 2]
```

```
## [1] -294.4870 -224.7649
```

```r
#Slope; Change in x per 1 y unit
sumCoef[2, 1] + c(-1, 1) * qt(0.975, df = fit$df) * sumCoef[2, 2]
```

```
## [1] 3556.398 3885.651
```

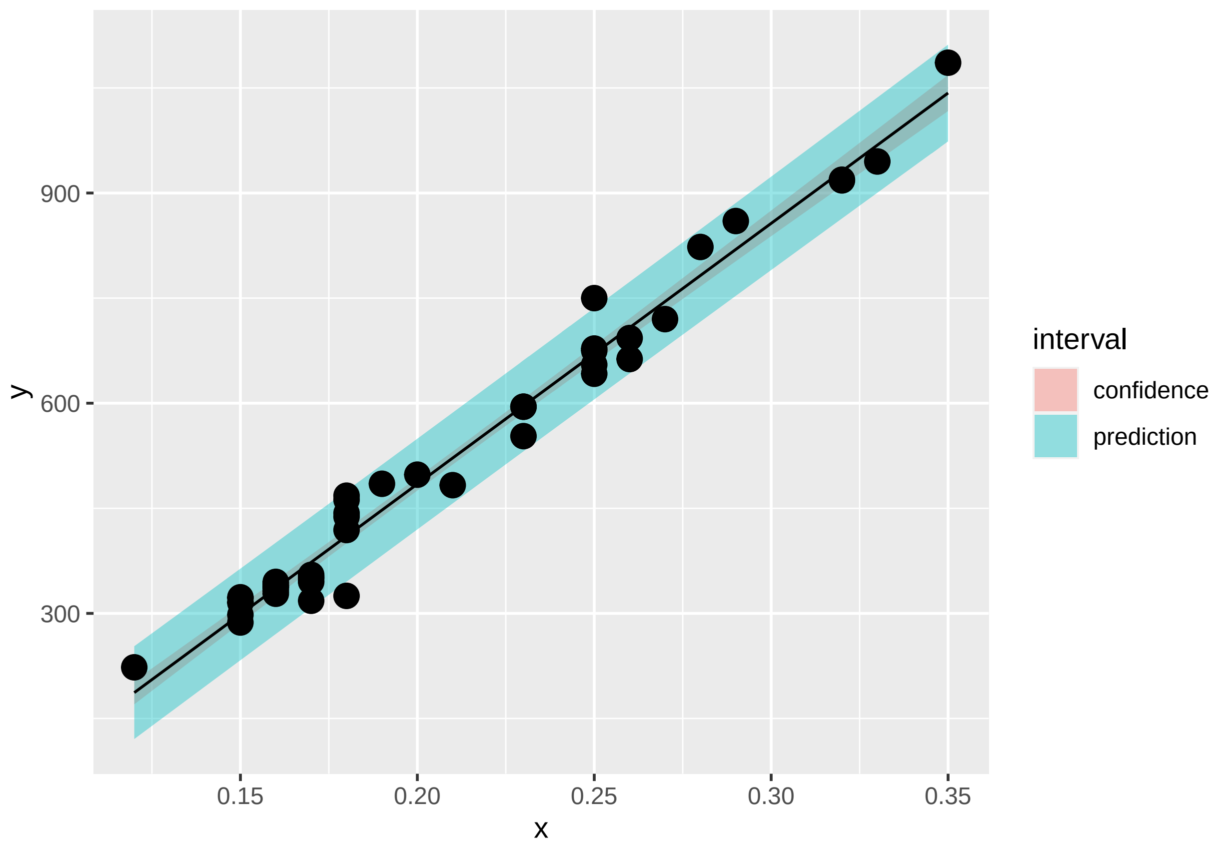
Prediction

- Consider predicting Y at a value of X
  + Predicting the price of a diamond given the carat
  + Predicting the height of a child given the height of the parents
- The obvious estimate for prediction at point x_0 is $\hat{\beta}_0 + \hat{\beta}_1 x_0$
- A standard error is needed to create a prediction interval
- There's a distinction between intervals for the regression line at points and the prediction of what a y would be at point x_0
- Line at x_0 std. error: $\hat{\sigma}\sqrt{\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (X_i - \bar{X})^2}}$
  + Variance will be the least when predicting the average of x
  + The denominator is how variable the 'x's are, so the more variability the less this error
- Prediction interval std. error at x_0: $\hat{\sigma}\sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (X_i - \bar{X})^2}}$

Generating Prediction Intervals in Diamond Data Set

```r
newx <- data.frame(x = seq(min(x), max(x), length = 100))
##Data Wranglin'
p1 <- data.frame(predict(fit, newdata = newx, interval = ("confidence")))
p2 <- data.frame(predict(fit, newdata = newx, interval = ("prediction")))
#p1 is giving confidence for each interval
```

```r
#p2 is giving the actual prediction at that data value

p1$interval = "confidence"
p2$interval = "prediction"
p1$x <- newx$x
p2$x <- newx$x
dat <- rbind(p1, p2)
names(dat)[1] <- "y"

##Plotting
plot <- ggplot(dat, aes(x, y)) +
        geom_ribbon(aes(ymin = lwr, ymax = upr, fill = interval),
                    alpha = 0.4) +
        geom_line() +
        geom_point(data = data.frame(x = x, y = y), size = 4)
plot
```



* Blue is prediction area, salmon color is preciting the line at each spot.
+ Both get narrower near middle since we're more confident as we are closer to the mean of x.

Lesson with `swirl()`: Introduction to Multivariable Regression

- Once we identify one regression line we can eliminate it to reduce the dimensions of data

- By subtracting the mean from each variable, the regression line goes through the orgin, hence its intercept is zero.
  + thus we eliminate one of the two regressors, the constant, leaving just the predicting variable
  + Subtracting the means is a special case of Gaussian Elimination
  - We pick one regressor and replace all other varaibles by the residuals of their regressions against that one
  + Subtracting the mean is equivalent to replacing a varaible by the residual of its regression against 1.
  - as such `lm(child ~ 1, galton)` will give an intercept of the mean, with a slope of 0.

Eliminate Variable Function

- First we want a function to regress the given variable on the given predictor, suppressing the intercept, and return the residual.

```
regressOneOnOne <- function(predictor, other, dataframe){
  # Point A. Create a formula such as Girth ~ Height -1
  formula <- paste0(other, " ~ ", predictor, " - 1")
  # Use the formula in a regression and return the residual.
  resid(lm(formula, dataframe))
}
```

- Using that function we can write another function to eliminate the specified predictor from the dataframe by regressing all other variables on that predictor and returning a data frame containing the residuals of those regressions.

```
eliminate <- function(predictor, dataframe){
  # Find the names of all columns except the predictor.
  others <- setdiff(names(dataframe), predictor)
  # Calculate the residuals of each when regressed against
  # the given predictor with the previous function
  temp <- sapply(others, function(other)regressOneOnOne(predictor, other, dataframe))
  # convert matrix of resids to a data frame and return.
  as.data.frame(temp)
}
```

- We could use eliminate multiple times to get rid of more and more variables, each time essentially using Gaussian elimination to re-express all the terms such that they are plotted with the mean's intercection as the orgin. This in turn replaces the outcome and all other regressors by their residuals against the chosen variable.

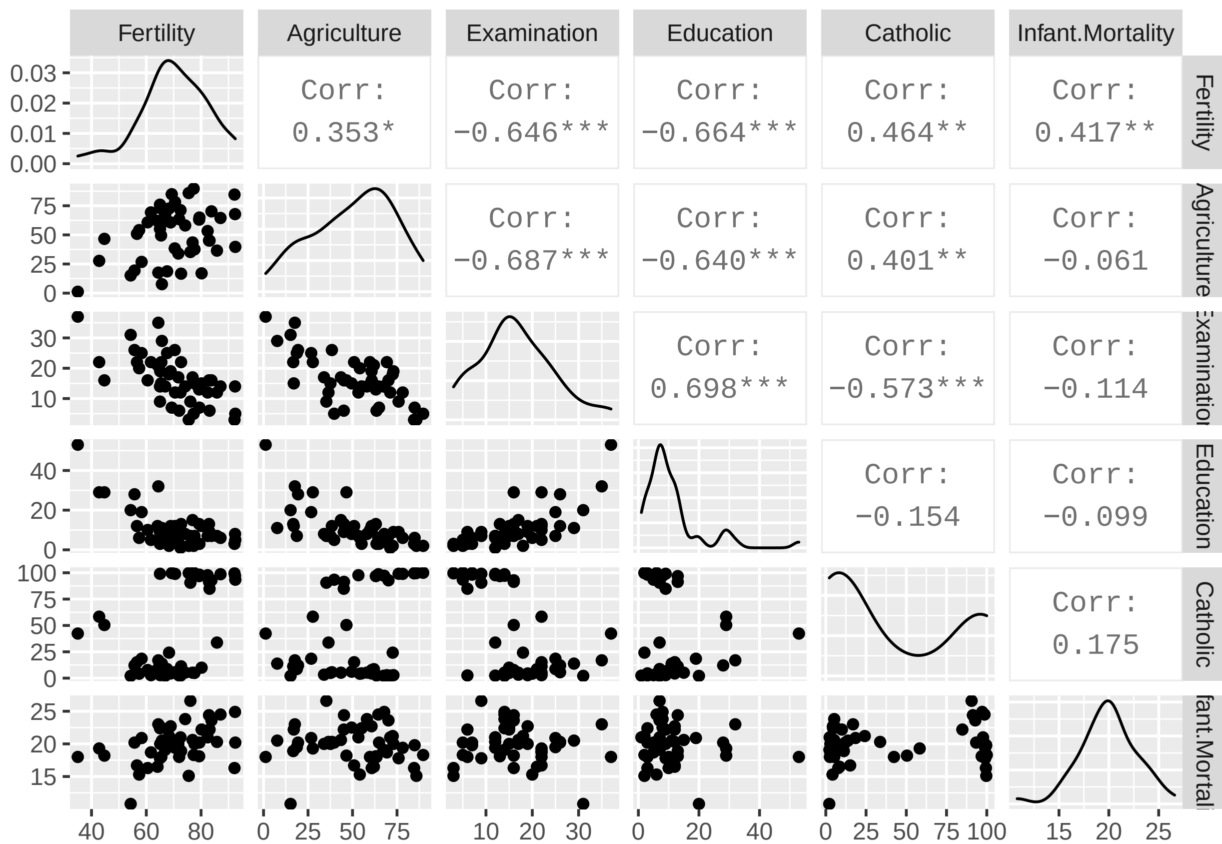
Lesson with `swirl()`: MultiVar Examples

- This data was gathered in 1888 in Switzerland, below are explaination fo the variables, all of which except fertility represent proportions of the population.

+ Fertility - a common standardized fertility measure
+ Agriculture - % of males involved in agriculture as occupation
+ Examination - % draftees receiving highest mark on army examination
+ Education - % education beyond primary school for draftees
+ Catholic - % catholic (as opposed to protestant)
+ Infant.Mortality - live births who live less than 1 year

- Check out this 6 by 6 array of scatterplots showing pairwise relations between the variables. + Lol, jk they just show the points plotted because I couldn't figure out `ggpairs`, I ought to use lattice for this task, but I'm just going to move on because I've spent enough time on it :(

```
data("swiss"); library(GGally)
ggpairs(swiss, lower = list(continous = "smooth"))
```



Reading Multiple Explanatory Variables

```
results <- summary(lm(Fertility ~ ., data = swiss))
results
```

```
##
## Call:
## lm(formula = Fertility ~ ., data = swiss)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -15.2743  -5.2617   0.5032   4.1198  15.3213
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      66.91518   10.70604   6.250 1.91e-07 ***
## Agriculture      -0.17211    0.07030  -2.448  0.01873 *
## Examination      -0.25801    0.25388  -1.016  0.31546
## Education        -0.87094    0.18303  -4.758 2.43e-05 ***
## Catholic          0.10412    0.03526   2.953  0.00519 **
## Infant.Mortality  1.07705    0.38172   2.822  0.00734 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.165 on 41 degrees of freedom
## Multiple R-squared:  0.7067, Adjusted R-squared:  0.671
## F-statistic: 19.76 on 5 and 41 DF,  p-value: 5.594e-10
```

- The `Coefficents` table states the Estimate/Slope for each explanatory variable to the dependent variable. For example:
  + For every 1% increase in males involved in argiculture as an accupation we expect a .17 decrease in fertility, if all other variables are held constant.
  + For every 1% increase in Catholisism we expect a .10 increase in fertility, if all other variables are held constant.
  + For every 1% increase in education we expect a .87 decrease in fertility, if all other variables are held constant. + Etc., etc.…
- The astrieks indicate what level of significance that explanatory variable has on the dependent variable, fertility. For example the alpha level of the t-test for Agriculture has one * as such it is significant at an alpha level of 0.05
- Hoever, if only Agriculture is listed as the independent variable we will see the coefficent change to positive, indicating that sometimes additional variables can affect the influence of an independent vairable on a dependent one.

```r
summary(lm(Fertility ~ Agriculture, swiss))$coef
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 60.3043752 4.25125562 14.185074 3.216304e-18
## Agriculture  0.1942017 0.07671176  2.531577 1.491720e-02
```

- One last note: Adding additional, repeated info to a lm won't change the result, for example…

```r
extra <- (swiss$Education + swiss$Agriculture)
extraLM <- lm(Fertility ~. + extra, swiss)$coef
extraLM
```

```
##      (Intercept)      Agriculture      Examination        Education
##       66.9151817       -0.1721140       -0.2580082       -0.8709401
##         Catholic Infant.Mortality            extra
##        0.1041153        1.0770481               NA
```

```r
lm(Fertility ~ ., swiss)$coef - extraLM
```

```
## Warning in lm(Fertility ~ ., swiss)$coef - extraLM: longer object length is not
## a multiple of shorter object length

##      (Intercept)      Agriculture      Examination        Education
##                0                0                0                0
##         Catholic Infant.Mortality            extra
##                0                0               NA
```

- The above code returns NA for extra because it gave no additional info to the linear model, and when substracting all the coefficients we can see there is no diffrence between the orginal and lm with `extra`

Quiz 2

1. Given...

```r
x <- c(0.61, 0.93, 0.83, 0.35, 0.54, 0.16, 0.91, 0.62, 0.62)
y <- c(0.67, 0.84, 0.6, 0.18, 0.85, 0.47, 1.1, 0.65, 0.36)
```

Give a P-value for the two sided hypothesis test of whether β_1 from a linear regression model is 0 or not

```r
results <- summary(lm(y ~ x))
results$coefficients
```

```
##              Estimate Std. Error   t value   Pr(>|t|)
## (Intercept) 0.1884572  0.2061290 0.9142681 0.39098029
## x           0.7224211  0.3106531 2.3254912 0.05296439
```

- 0.05296

2. Consider the previous problem, give the estimate of the residual standard deviation

```r
results$sigma
```

```
## [1] 0.2229981
```

3. In the `mtcars` data set, fit a linear regression model of weight (predictor) on mpg (outcome). Get a 95% confidence interval for the expected mpg at the average weight. What is the lower endpoint?

```r
fit <- lm(mpg ~ wt, mtcars)
results <- predict(fit, newdata = data.frame(wt = mean(mtcars$wt)),
        interval = "confidence")
results
```

```
##        fit      lwr      upr
## 1 20.09062 18.99098 21.19027
```

- 18.991

4. Refer to the help file for `mtcars`. What is the weight coefficient interpreted as?
   * Expected change in mpg/1,000 lb increase in weight.

5. Consider again the `mtcars` data set and a linear regression model with mpg as predicted by weight (1,000 lbs). A new car is coming weighing 3000 pounds. Construct a 95% **prediction** interval for its mpg. What is the upper endpoint?

```
results <- predict(fit, newdata = data.frame(wt = 3),
        interval = "prediction")
results
```

```
##        fit      lwr      upr
## 1 21.25171 14.92987 27.57355
```

```
* 27.57
```

6. Consider the `mtcars` data set again with mpg as predicted by weight. A "short" ton is defined as 2,000 lbs. Construct a 95% confidence interval for the expected change in mpg per 1 short ton increase in weight. Give the lower endpoint.

```
fit <- lm(mpg ~ I(wt * 1000/2000), mtcars)
coefs <- summary(fit)$coef
inter <- coefs[2,1]
slope <- coefs[2,2]
slopeInterval <- inter + c(-1, 1) * qt(0.975, df = fit$df) * slope
slopeInterval
```

```
## [1] -12.97262  -8.40527
```

7. If my X from a linear regression is measured in centimeters and I convert it to meters what would happen to the slope coefficient?

- Slope is *rise/run*, or *change in y/change in x* since we're changing the units of x we have to multiple by the conversion factor *100 cm/1 m*, which is to multiple the coefficent by 100.

8. I have an outcome, Y, and a predictor, X and fit a linear regression model with $Y = \beta_0 + \beta_1 X + \epsilon$ to obtain $\hat{\beta}_0$ and $\hat{\beta}_1$. What would be the consequence to the subsequent slope and itnercept if I were to refit the model with a new regressor, $X + c$ for some constant, c?

- $Y = \beta_0 + \beta_1 X + \epsilon = \beta_0 - c\beta_1 + c\beta_1 + \beta_1 X + \epsilon$
  $= \beta_0 - c\beta_1 + \beta_1(X + c) + \epsilon$
  \+ As such this new intercept is $\beta_0 - c\beta_1$

9. Refer back to the mtcars data set with mpg as an outcome and weight (wt) as the predictor. About what is the ratio of the the sum of the squared errors, $\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$ when comparing a model with just an intercept (denominator) to the model with the intercept and slope (numerator)?

```
results <- summary(lm(mpg~wt, mtcars))
1 - results$r.squared
```

```
## [1] 0.2471672
```

10. Do the residuals always have to sum to 0 in linear regression?

- Yes, if an intercept is included in the resid.s

Multivariable Regression, Residuals, & Diagnostics

Multivariable Regression

Multivariable Regression Part 1

Intro Scenerio Example

- If one were to present evidence of a relationship between breath mint useage (mints per day, X) and pulmonary function (lung health measurement in FEV (Forced Expiratory Volume)) + You may be skeptical, for smokers tend to use more breath mints than non-smokers, and smoking is related to a loss in pulmonary function.
+ To conteract this skepticism one may want to investigate if non-smoking breath mint users had lower lung function than non-smoking non-breath mint users and if smoking breath mint suers had lower lung function than smoking non-breath mint users.
+ In other words, to consider these results, we would have to demonstrate that the results hold when smoking status is held fixed.

The Linear Model

- How can one generalize Simple Linear Regression (SLR) to incorporate lots of regressors for the purpose of predictions?

- Are there consequences for adding lots fo regressors?
+ Omitting variables can cause incorrect predictions
+ Adding too many variables into a model will eventually give 0 residuals, even if adding in random garbage predictors. This is called **over-fitting**

- The general linear model extends SLR by adding terms linearly intot he model
$Y_i = \beta_1 X_{1i} + \beta_2 X_{2i} + ... + \beta_p X_{pi} + \epsilon_i = \sum_{k=1}^{p}(X_{ik}\beta_j) + \epsilon_i$
+ Here $X_{1i} = 1$ typically, so that an intercept is included.

- Least squares (and hence Middle of Least squares estimates under iid Gaussianity of the errors) minimizes:
$\sum_{i=1}^{n}(Y_i - \sum_{k=1}^{p} X_{ki}\beta_j)^2$ + Note that the important linearity is linearity in the coefficients, thus:
$Y_i = \beta_1 X_{1i}^2 + \beta_2 X_{2i}^2 + ... + \beta_p X_{pi}^2 + \epsilon_i$
+ The above equation is still a linear model, we've just squared the element of the predictors

Multivariable Regression Part 2

How to Get Estimates

- Recall that the Least Squares estimate for regression through the origin, $E[Y_i] = X_{1i}\beta_1$, was $\sum(X_iY_i)/\sum X_i^2$

- For two regressors we would have a regression line of $E[Y_i] = X_{1i}\beta_1 + X_{2i}\beta_2 = \mu_i$ and least squares would minimize $\sum_{i=1}^n (Y_i - X_{1i}\beta_1 - X_{2i}\beta_2)^2$

- With two independent variables, X_1, X_2, being used in a `lm` X_1 has to then be adjusted for X_2 being involved

- So for the equation: $E[Y_i] = X_{1i}\hat{\beta}_1 + X_{2i}\hat{\beta}_2$
  + $\hat{\beta}_1 = \frac{\sum_{i=1}^n (e_{i,Y|X_2} * e_{i,X_1|X_2})}{\sum_{i=1}^n e^2_{i,X_1|X_2}}$
  - Remember that e_i is just a col. vector of 0's except at the i^{th} position, where there is a 1.
  + Essentially the regression estimate for β_1 is the regression through the origin estimate having regressed X_2 out of both the response and the predictor.
  - Similarly, the regression estimate for β_2 is the regression through the orgin estimate having regressed X_1 out of both the response and the predictor.
  + Multivariate regression estimates are exactly those having removed the linear relationship of the other variables from both the regressor and response.

- $Y_i = \beta_1 X_{1i} + \beta_2 X_{2i}$ where $X_{2i} = 1$ is an intercept term.

- Notice the fitted coefficient of X_{2i} on Y_i is $\bar{Y}$
  + The residuals $e_{i,Y|X_2} = Y_i - \bar{Y}$

- Notice the fitted coefficient of X_{2i} on $X1i$ is $\bar{X}_1$
  + The residuals $e_{i,X_1|X_2} = X_{1i} - \bar{X}_1$

- Thus
  $\hat{\beta}_1 = \frac{\sum_{i=1}^n (e_{i,Y|X_2} * e_{i,X_1|X_2})}{\sum_{i=1}^n e^2_{i,X_1|X_2}}$
  $= \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^n (X_i - \bar{X})^2}$
  $= Cor(X, Y)\frac{Sd(Y)}{Sd(X)}$

- For multiple variables this case just keeps adding terms, the least square solutions have to minimize:
  $\sum_{i=1}^n (Y_i - X_{1i}\beta_1 - ... - X_{pi}\beta_p)^2$
  + The least squares estimate for the coefficeint of a multivariate regression model is exactly regression through the orgin with the linear relationships with the other regressors removed from both the regressor and outcome by taking residuals.
  + In this sense, multivariate regression "adjusts" a coefficient for the linear impact of the other variables.

Multivariable Regression Continued

Simulation

```
set.seed(1618033)
n <- 100
x1 <- rnorm(n)
x2 <- rnorm(n)
x3 <- rnorm(n)
y <- 1 + x1 + x2 + x3 +
        rnorm(n, sd = .1) #rand. noise as \epsilon term
ey <- resid(lm(y ~ x2 + x3))
ex <- resid(lm(x1 ~ x2 + x3))
sum(ey * ex) / sum(ex ^ 2) #Reg. through orgin estimate
```

```
## [1] 0.9889948
```

```
coef(lm(ey ~ ex - 1)) #Reg. est of ey to ex
```

```
##        ex
## 0.9889948
```

```
coef(lm(y ~ x1 + x2 + x3)) #Notice x1 is same as above
```

```
## (Intercept)          x1          x2          x3
##   1.0195807   0.9889948   1.0048448   1.0178638
```

Interpretation of the Coeficients

- Our model is $E[Y|X_1 = x_1, ..., X_p = x_p] = \sum_{k=1}^{p} x_k \beta_k$
  + Where β_k is the coeffient for each x_k

- The interpretation of a multivaraite regression coefficient is the expected change in the response per unit change in the regressor, holding all of the other regressors fixed. As such the diffrence between adding 1 to a regressor$_i$ and the orginal equation is just β_i, as seen below
  + Adding 1: $E[Y|X_1 = x_1 + 1, ..., X_p = x_p] = (x_1 + 1)\beta_1 + \sum_{k=2}^{p} x_k \beta_k$
  + Difference: $(x_1 + 1)\beta_1 + \sum_{k=2}^{p} x_k \beta_k - \sum_{k=1}^{p} x_k \beta_k = \beta_1$

Fitted Values, Residuals and Residual Variation

ALL of our Simple Linear Regression (SLR) quantites can be extended to linear models of multiple dimensions
* Model: $Y_i = \sum_{k=1}^{p} X_{ik} \beta_k + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$
* Fitted Responses: $\hat{Y}_i = \sum_{k=1}^{p} X_{ik} \hat{\beta}_k$
* Residuals: $e_i = Y_i - \hat{Y}_i$
* Variance ewstiamte $\hat{\sigma}^2 = \frac{1}{n-p} \sum_{i=1}^{n} e_i^2$
* To get predicted responses at new values, $x_1, ..., x_p$ simply plug them into the linear model $\sum_{k=1}^{p} x_k \hat{\beta}_k$
* Each coefficient has their own standard error, $\hat{\sigma}_{\hat{\beta}_k}$, and as such $\frac{\hat{\beta}_k - \beta_k}{\hat{\sigma}_{\hat{\beta}_k}}$ follows a T distribution with

$n - p$ degrees of freedom
* Predicted responses have standard errors and we can calculate predicted and expected rewsponse intervals.

Linear Models Summary

- Linear Models are the single most important applied statistical and amchine learning technique, *by far*.

- They can… + decompose a signal into its harmonics
  + flexibly fit complicated functions & curves.
  + fit factor variables as predictors
  + uncover complex multivariate relationships with the response
  + build accurate prediction models

Multivariable Regression Tips and Tricks

Multivariable Regression Examples Part 1

```r
library(datasets); data(swiss)
```

- `swiss` is a data frame with 47 observations on 6 variables

1) *Fertility* - Ig, a "common standardized fertility measure"

2) *Agriculture* - % of males invovled in agriculture as occupation

3) *Examination* - % of draftees receiving highest mark on army examination

4) *Education* - % education beyond primary school for draftees

5) *Catholic* - % catholic (as opposed to protestant)

6) *Infant.Mortality* - % of live births who live less than 1 year

- All variables but *Fertility* give proportions of the population

- These data are from Switzerland in 1888 from 47 French-speaking "provinces"

```r
all <- summary(lm(Fertility ~ . , data = swiss))
all$coefficients
```

```
##                    Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept)      66.9151817 10.70603759  6.250229 1.906051e-07
## Agriculture      -0.1721140  0.07030392 -2.448142 1.872715e-02
## Examination      -0.2580082  0.25387820 -1.016268 3.154617e-01
```

```
## Education        -0.8709401  0.18302860 -4.758492 2.430605e-05
## Catholic          0.1041153  0.03525785  2.952969 5.190079e-03
## Infant.Mortality  1.0770481  0.38171965  2.821568 7.335715e-03
```

- Each estimate would be interpreted as: Our model estimates an expected `Estimate` Increase/Decrease in standardized fertility for every 1% increase in percentage of `Explanatory Variable`

- The `Std. Error` describes how precise the `Estimate` is

- The t-test for $H_0 : \beta_{Agri} = 0$ versus $H_a : \beta_{Agri} \neq 0$ is significant.
  - R gives the `t value` for this test and the `Pr(>|t|)`, P-value
- Having only one predictor will change to coefficents

```
onlyAgri <- summary(lm(Fertility ~ Agriculture, data = swiss))
onlyAgri$coefficients
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 60.3043752 4.25125562 14.185074 3.216304e-18
## Agriculture  0.1942017 0.07671176  2.531577 1.491720e-02
```

```
all$coefficients[1:2,]
```

```
##              Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept) 66.915182 10.70603759  6.250229 1.906051e-07
## Agriculture -0.172114  0.07030392 -2.448142 1.872715e-02
```

- This diffrence when including new factors is a version of Simpson's paradox

Simulation

```
set.seed(1618033)
n <- 100
x2 <- 1:n
x1 <- 0.01 * x2 + runif(n, -0.1, 0.1)
y <- -x1 + x2 + rnorm(n, sd = 0.01)
summary(lm(y ~ x1))$coef
```

```
##               Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)  0.6769049   1.188377  0.5696045 5.702494e-01
## x1          96.1246184   2.014714 47.7112934 1.208965e-69
```

- Only looking at `x1` doesn't show the underneath pattern generated from `x2` and gives a very large slope, however adding in `x2` ...

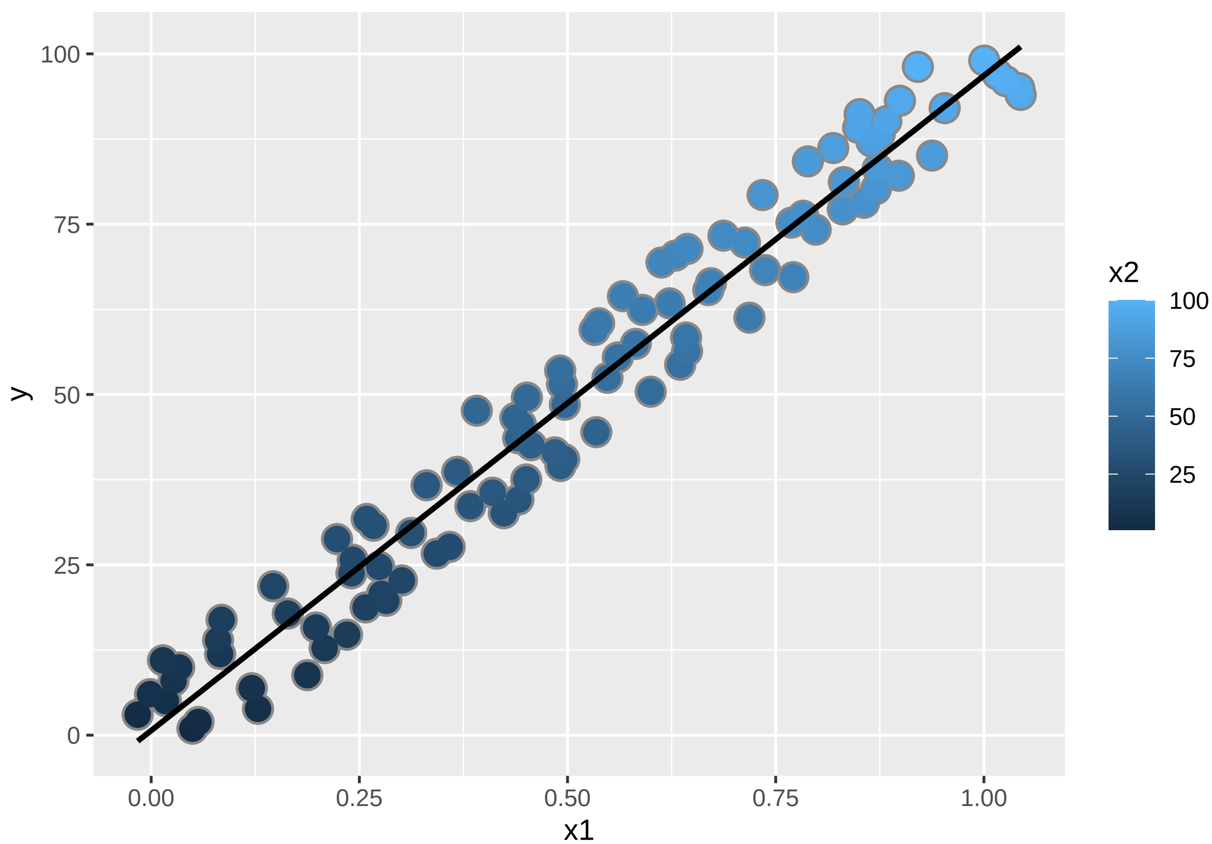
```
summary(lm(y ~ x1 + x2))$coef
```

```
##                  Estimate   Std. Error      t value      Pr(>|t|)
## (Intercept)  0.0003514301 0.0019009607    0.1848697  8.537172e-01
## x1          -1.0257893318 0.0159992072  -64.1150103  3.195073e-81
## x2           1.0002757089 0.0001613649 6198.8421136 2.580392e-273
```

- … gives a more accurate representation of the true coefficients for `y`.

- Let's do some plotting:

```
library(tidyverse)
dat <- data.frame(y = y, x1 = x1, x2 = x2,
                  ey = resid(lm(y ~ x2)), ex1 = resid(lm(x1 ~ x2)))
plot <- ggplot(dat, aes(x1, y, colour = x2)) +
  geom_point(colour = "#888888", size = 5) +
  geom_point(size = 4) +
  geom_smooth(method = lm, se = FALSE, colour = "#000000")
plot
```

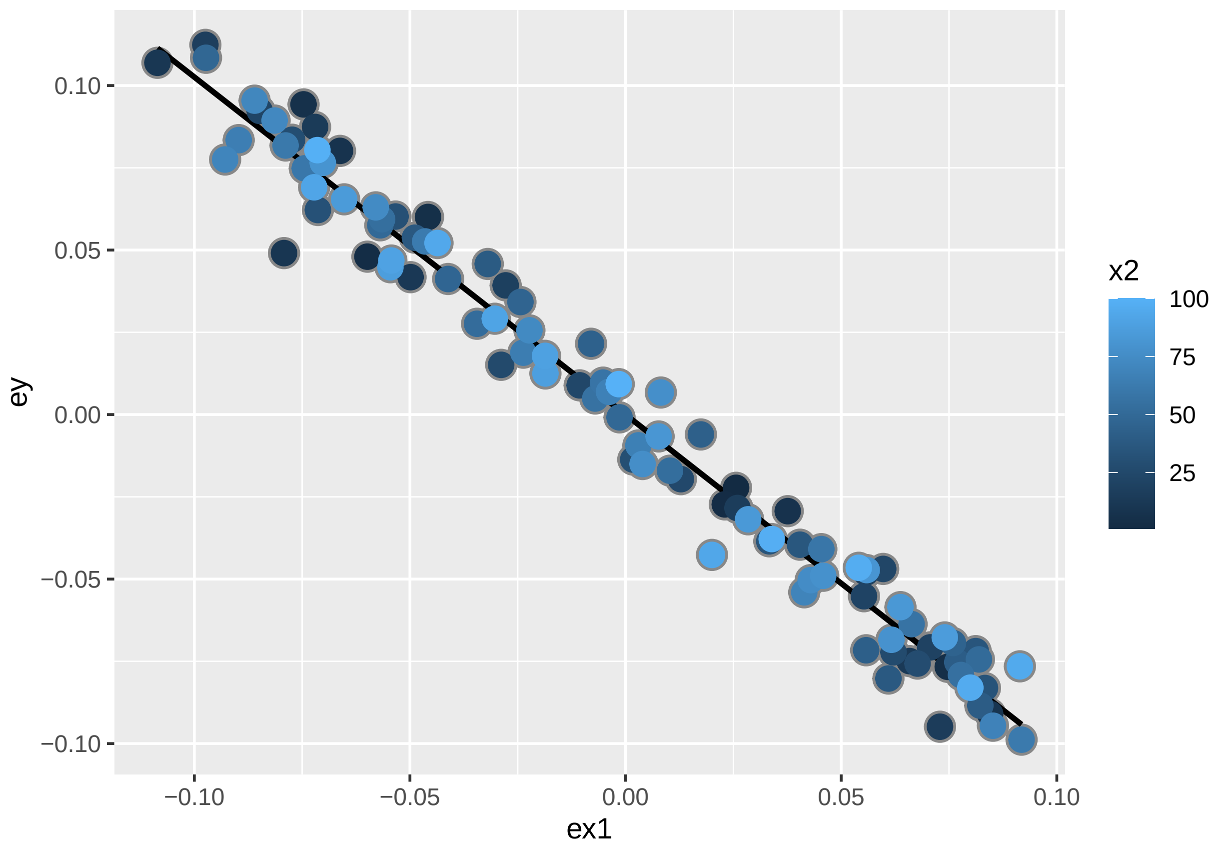


- Although `y` grows with `x1` it can also be seen that `x2` grows as `x1` grows

- Now let's look at the residuals of `y ~ x2` versus `x1 ~ x2`

```
residplot <- ggplot(dat, aes(ex1, ey, colour = x2)) +
  geom_point(colour = "#888888", size = 5) +
  geom_smooth(method = lm, se = FALSE, colour = "#000000") +
  geom_point(size = 4)
residplot
```

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



- This smoother has the slope of the `estimate` for `x1` above

Back to `swiss` Data Set

- The sign reverses itself witht the inclusion of Examinaiton and Education

- The percent of males in the province working in agriculture is negatively related to educational attainment (correlation of -0.6395) and Educationa nd Examination (correlation of 0.6984) are obviously measuring similar things.
  - Is the positive marginal an artifact for not having accounted for the Education level? (Education does have a strong effect, btw)

- At the minimum, anyone claiming that provinces that focus more on agriculture have higher fertility rates would immediately be open to criticism due to these other cofactors present in the data.

Multivariable Regression Examples Part 2

- Consider the linear model
  $Y_i = \beta_0 + X_{i1}\beta_i + \epsilon_i$
  - Where each X_{i1} is binary such that the value is 1 if measurement, i, is in a group and 0 otherwise.

- Then for people in the group, $E[Y_i] = \beta_0 + \beta_1$
- And for people not in the group, $E[Y_i] = \beta_0$
  - As such the mean of the treated group is $\hat{\beta}_0 + \hat{\beta}_1$
  - Likewise, the mean for the control group is just $\hat{\beta}_0$
- β_1 is interpretted as the increase or decrease in the mean of the treated group
  - This also gives the inference for the two groups, the same value a 2-sample t-test would give you

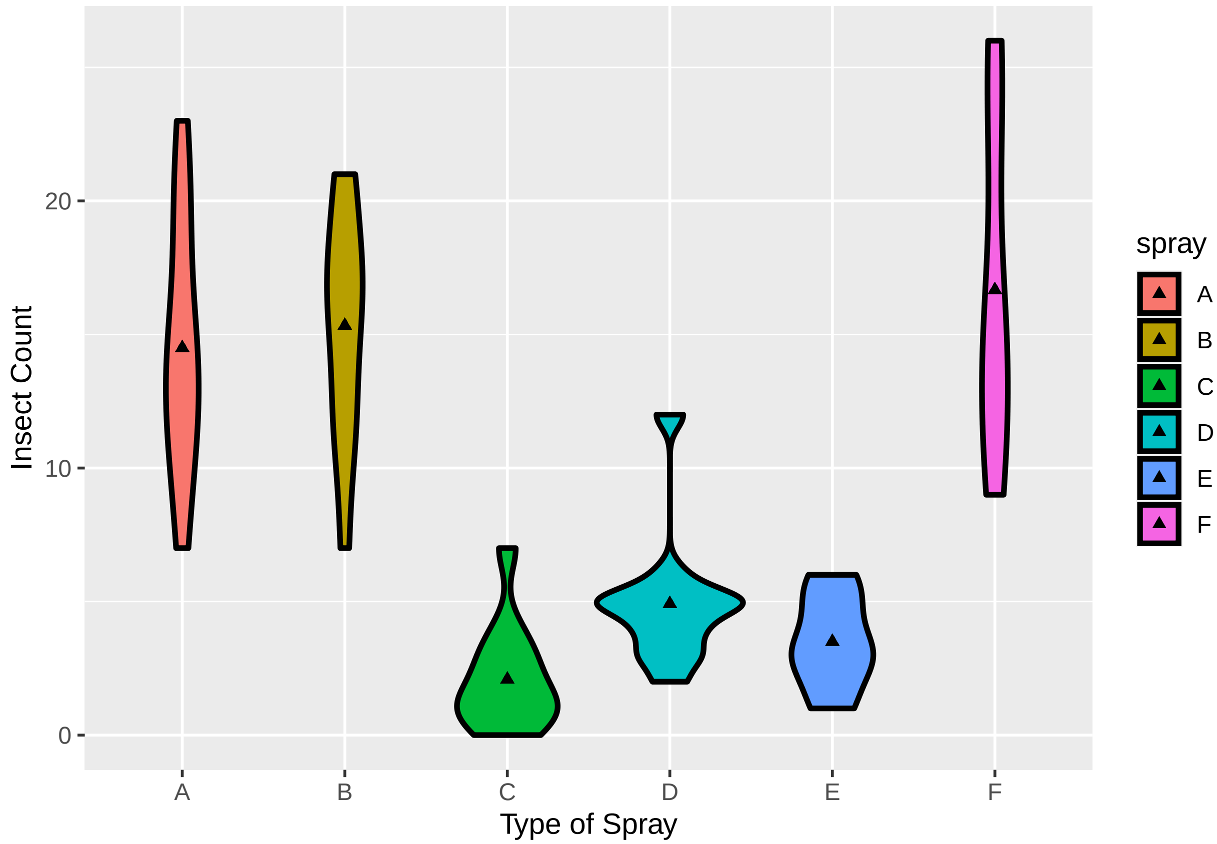
Comparing More Than 2 Levels

- Since the value of β_1 gives the value of a 2-sample t-test, you can extend this to perform t-tests across multiple variables
- Consider a multilevel factor level. For example a three level factor such as the US political party affiliation: Republican, Democrat, Independent.
- $Y_i = \beta_0 + X_{i1}\beta_1 + X_{i2}\beta_2 + \epsilon_i$
  - X_{i1} is 1 for Republicans and 0 otherwise
  - X_{i2} is 1 for Democrats and 0 otherwise
- If i is **...** then $E[Y_i] = ...$
  - **Republican** $= \beta_0 + \beta_1$
  - **Democrat** $= \beta_0 + \beta_2$
  - **Independent** $= \beta_0$
- β_1 compares Republicans to Independents
- β_2 compares Democrats to Independents
- $\beta_1 - \beta_2$ compares Republicans to Democrats
- (Choice of reference category changes the interpretation.)

Example in R with InsectSprays

```
data(InsectSprays); library(tidyverse)
mus <- InsectSprays %>% group_by(spray) %>% summarise(mean = mean(count))
dat <- merge(InsectSprays, mus, by = "spray")
plot <- ggplot(dat, aes(spray, count, fill = spray)) +
  geom_violin(colour = "#000000", size = 1) +
```

```r
  labs(x = "Type of Spray", y = "Insect Count") +
  geom_point(aes(y=mean), shape = 17)
plot
```



- Fitting `count` as a response to `spray`, displaying mean with a triangle

```r
res <- lm(count ~ spray, InsectSprays)
summary(res)$coef
```

```
##               Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)  14.5000000   1.132156 12.8074279 1.470512e-19
## sprayB        0.8333333   1.601110  0.5204724 6.044761e-01
## sprayC      -12.4166667   1.601110 -7.7550382 7.266893e-11
## sprayD       -9.5833333   1.601110 -5.9854322 9.816910e-08
## sprayE      -11.0000000   1.601110 -6.8702352 2.753922e-09
## sprayF        2.1666667   1.601110  1.3532281 1.805998e-01
```

- It can be seen that `sprayA` is missing, this is because all of the other sprays are in comparision to `sprayA`, so the estimate values are indicating the change of each spray relative to spray A

- The average count of `sprayA` is just the `Estimate` of the `(Intercept)`

- The average count of `sprayC` would be the `Estimate` of the `(Intercept)` plus the `Estimate` of `sprayC`, as seen below

```r
c(summary(res)$coef[1,1], mus$mean[1])
```

```
## [1] 14.5 14.5
```

```r
c(summary(res)$coef[1,1] + summary(res)$coef[3,1], mus$mean[3])
```

```
## [1] 2.083333 2.083333
```

Hard Coding the Dummy Variables

```r
summary(lm(count ~
             I(1 * (spray == 'B')) + I(1 * (spray == 'C')) +
             I(1 * (spray == 'D')) + I(1 * (spray == 'E')) +
             I(1 * (spray == 'F')),
           data = InsectSprays))$coef
```

```
##                          Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)            14.5000000   1.132156 12.8074279 1.470512e-19
## I(1 * (spray == "B"))   0.8333333   1.601110  0.5204724 6.044761e-01
## I(1 * (spray == "C")) -12.4166667   1.601110 -7.7550382 7.266893e-11
## I(1 * (spray == "D"))  -9.5833333   1.601110 -5.9854322 9.816910e-08
## I(1 * (spray == "E")) -11.0000000   1.601110 -6.8702352 2.753922e-09
## I(1 * (spray == "F"))   2.1666667   1.601110  1.3532281 1.805998e-01
```

```r
summary(res)$coef
```

```
##               Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)  14.5000000   1.132156 12.8074279 1.470512e-19
## sprayB        0.8333333   1.601110  0.5204724 6.044761e-01
## sprayC      -12.4166667   1.601110 -7.7550382 7.266893e-11
## sprayD       -9.5833333   1.601110 -5.9854322 9.816910e-08
## sprayE      -11.0000000   1.601110 -6.8702352 2.753922e-09
## sprayF        2.1666667   1.601110  1.3532281 1.805998e-01
```

- As such we can change what the β_0 spray is, aka the reference level

```r
summary(lm(count ~
             I(1 * (spray == 'A')) + I(1 * (spray == 'B')) +
             I(1 * (spray == 'D')) + I(1 * (spray == 'E')) +
             I(1 * (spray == 'F')),
           data = InsectSprays))$coef
```

```
##                        Estimate Std. Error  t value     Pr(>|t|)
## (Intercept)            2.083333   1.132156 1.840148 7.024334e-02
## I(1 * (spray == "A")) 12.416667   1.601110 7.755038 7.266893e-11
## I(1 * (spray == "B")) 13.250000   1.601110 8.275511 8.509776e-12
## I(1 * (spray == "D"))  2.833333   1.601110 1.769606 8.141205e-02
## I(1 * (spray == "E"))  1.416667   1.601110 0.884803 3.794750e-01
## I(1 * (spray == "F")) 14.583333   1.601110 9.108266 2.794343e-13
```

- Instead of typing all the factors one can just `relevel` the factors

```r
spray_ <- relevel(InsectSprays$spray, "C")
summary(lm(count ~ spray_, data = InsectSprays))$coef
```

```
##              Estimate Std. Error  t value     Pr(>|t|)
## (Intercept)  2.083333   1.132156 1.840148 7.024334e-02
## spray_A     12.416667   1.601110 7.755038 7.266893e-11
## spray_B     13.250000   1.601110 8.275511 8.509776e-12
## spray_D      2.833333   1.601110 1.769606 8.141205e-02
## spray_E      1.416667   1.601110 0.884803 3.794750e-01
## spray_F     14.583333   1.601110 9.108266 2.794343e-13
```

- Including all the parameters will just return `NA` for one of the parameters (In this case it would be for `sprayA`)

- We can remove the intercept to get a set of all the levels

```r
summary(lm(count ~ 0 + spray, data = InsectSprays))$coef
```

```
##         Estimate Std. Error   t value     Pr(>|t|)
## sprayA 14.500000   1.132156 12.807428 1.470512e-19
## sprayB 15.333333   1.132156 13.543487 1.001994e-20
## sprayC  2.083333   1.132156  1.840148 7.024334e-02
## sprayD  4.916667   1.132156  4.342749 4.953047e-05
## sprayE  3.500000   1.132156  3.091448 2.916794e-03
## sprayF 16.666667   1.132156 14.721181 1.573471e-22
```

Summary

- If we treat `spray` as a factor, R includes an intercept and omits the alphabetically first level of the factor.
  - All t-tests are for comparisons of Sprays versus Spray A

  - Emprirical mean for A is the `(Intercept)`

  - Other group means are the intercept plus their coefficient

- If we omit an intercept, then it includes terms for all levels of the factor.
  - Group means are the coeffcients

  - Tests are tests of whether the groups are different than zero.
- If we want comparisions between, say between Spray B and C, we could refit the model with C (or B) as the reference level.

Additional Tid-bits on this data
* Counts are counded from below by 0, which violates the assumption of normality of the errors.
+ Also there are counts near 0, so both the actual assumption and the intent of the assumption are violated.
* Variance does not appear to be constant
* Perhaps taking logs of the counts would help.

+ There are 0 counts, so perhaps `log(count + 1)`
* Poisson GLMs for fitting count data will be covered later in this course.
* Because of these issues our means are correct, but our inference would not be.

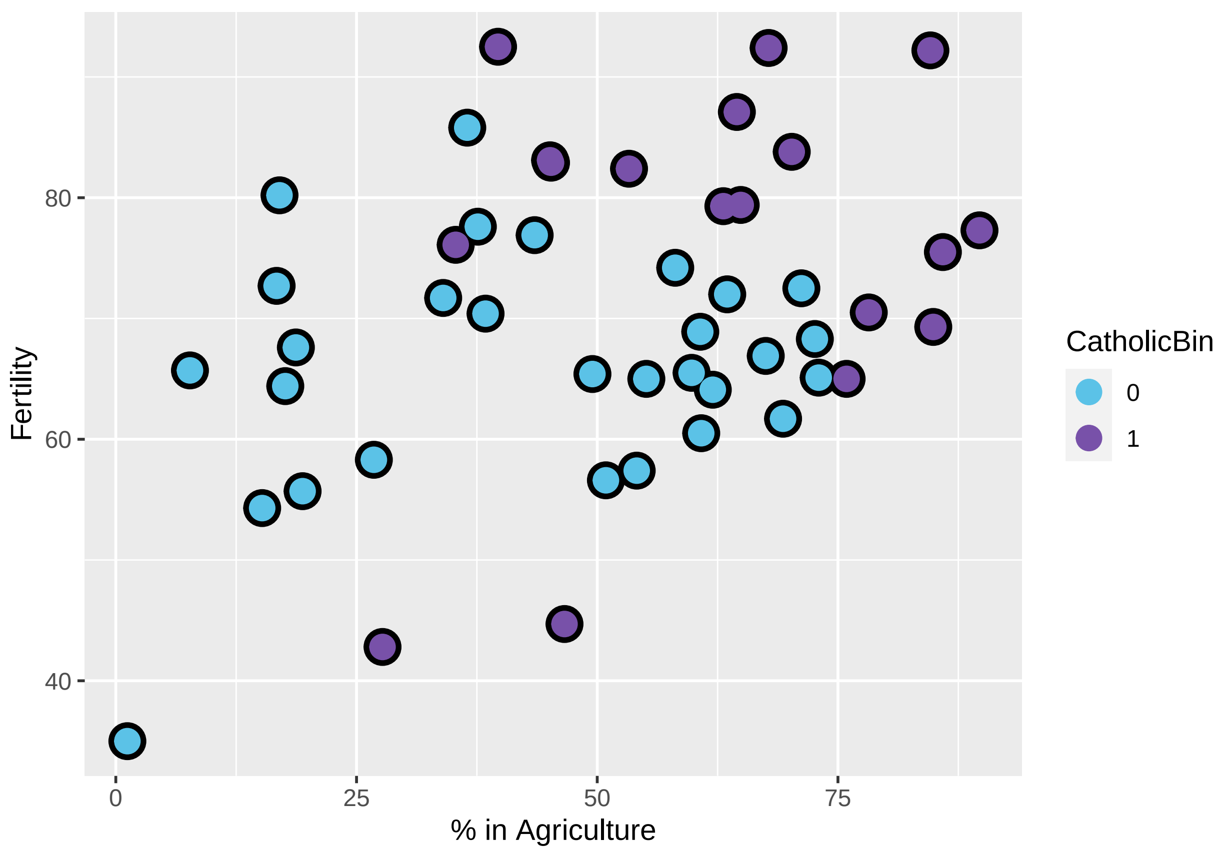
Multivariable Regression Examples Part 3

We'll be looking at the `swiss` data set to see how we can translate a proportion to a binary variable

```
library(datasets); data(swiss)
##Create binary var from Catholic, 1 if > 50%, 0 o/w
dat <- swiss %>% mutate(CatholicBin = factor(1 * (Catholic > 50))) %>%
  select(Agriculture, Fertility, CatholicBin)
```

Now we can plot by subsets of variables

```
pal <- c("#5BC2E7", "#7851A9")
plot <- ggplot(dat, aes(Agriculture, Fertility, colour = CatholicBin)) +
  scale_colour_manual(values = pal) +
  geom_point(size = 6, colour = "#000000") + geom_point(size = 4) +
  labs(x = "% in Agriculture", y = "Fertility")
plot
```



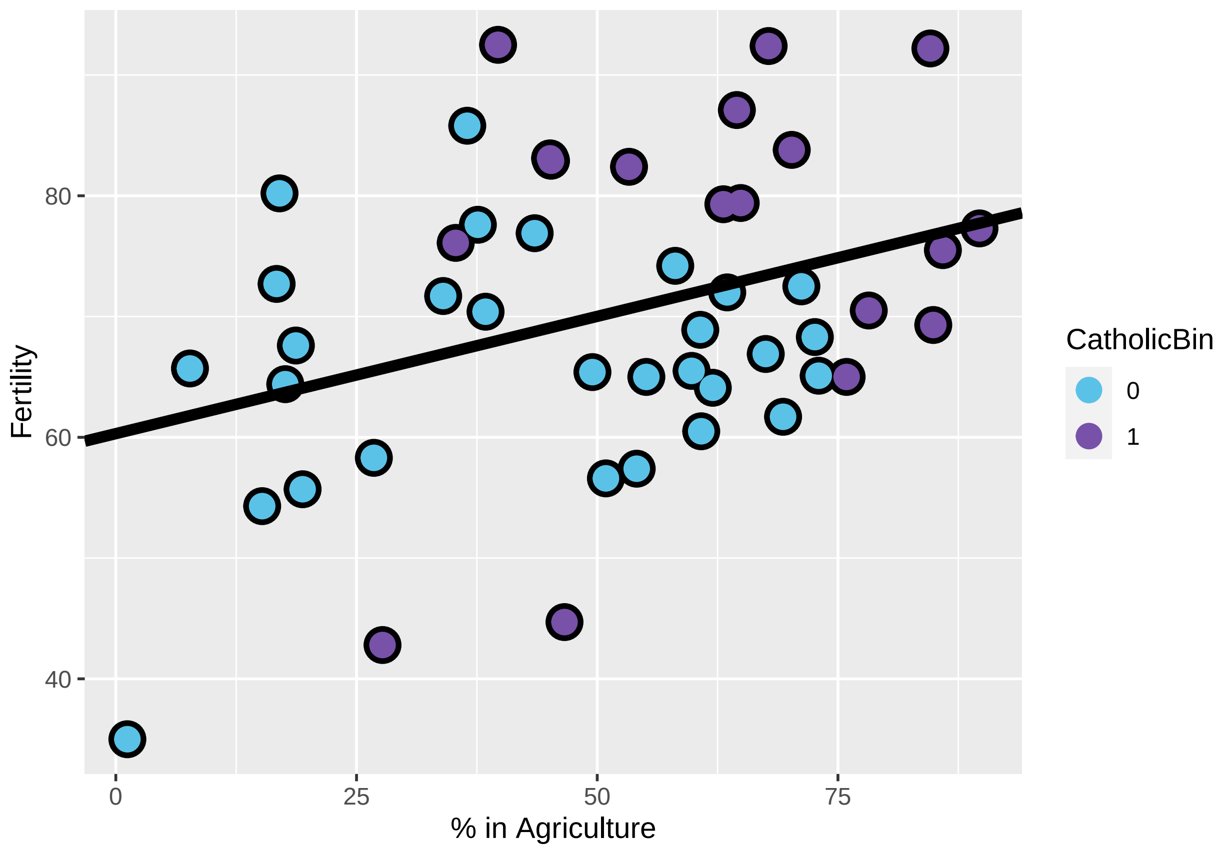
Fitting the Model

- $y = Fertility$
- $x_1 = Agriculture$
- $x_2 = 1$ if over .50 Catholic, 0 otherwise
  Then our model could be:
- $E[y|x_1 x_2] = \beta_0 + \beta_1 x_1 + \beta_2 x_2$
  Which would give us the expected values of:
- $E[y|x_2 = 0] = \beta_0 + \beta_1 x_1$
- $E[y|x_2 = 1] = \beta_0 + \beta_2 + \beta_1 x_1$
  These two models have the same slope, β_1, but different intercepts.
  To make a model that has diffrent slopes we could fit it as such:
- $E[y|x_1 x_2] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$
  We then have the following expected values:
- $E[y|x_2 = 0] = \beta_0 + \beta_1 x_1$
- $E[y|x_2 = 1] = \beta_0 + \beta_1 x_1 + \beta_2 + \beta_3 x_1$
  $= \beta_0 + \beta_2 + (\beta_1 + \beta_3) x_1$
  These models now have both different intercepts and slopes because we included an **interaction term**, $\beta_3 x_1 x_2$

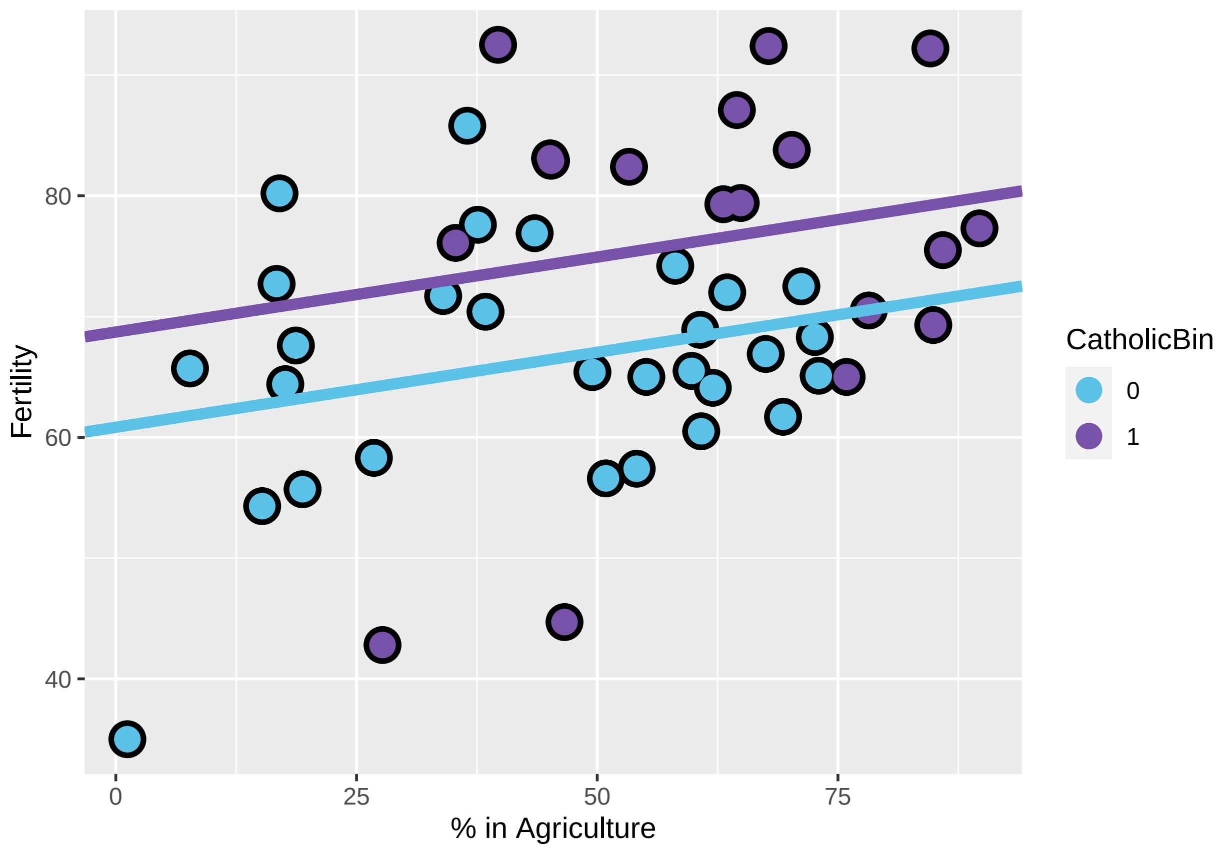
Multivariable Regression Examples Part 4

We'll now look an example of a model with an **interaction term**

```
##Fitting without CatholicBin
fafit <- lm(Fertility ~ Agriculture, data = dat)
plot + geom_abline(intercept = coef(fafit)[1],
                   slope = coef(fafit)[2], size = 2)
```




```r
##Fitting first proposed model
facfit <- lm(Fertility ~ Agriculture + CatholicBin, data = dat)
plot + geom_abline(intercept = coef(facfit)[1], slope = coef(facfit)[2],
                   size = 2, colour = pal[1]) +
  geom_abline(intercept = coef(facfit)[1] + coef(facfit)[3],
              slope = coef(facfit)[2], size = 2, colour = pal[2])
```




```
##Fitting model with diffrent slopes
interactfit <- lm(Fertility ~ Agriculture * CatholicBin, dat)
##Let's look at the coef.s before tossing them all in
summary(interactfit)$coef
```

```
##                              Estimate  Std. Error    t value     Pr(>|t|)
## (Intercept)               62.04993019  4.78915566 12.9563402 1.919379e-16
## Agriculture                0.09611572  0.09881204  0.9727127 3.361364e-01
## CatholicBin1               2.85770359 10.62644275  0.2689238 7.892745e-01
## Agriculture:CatholicBin1   0.08913512  0.17610660  0.5061430 6.153416e-01
```

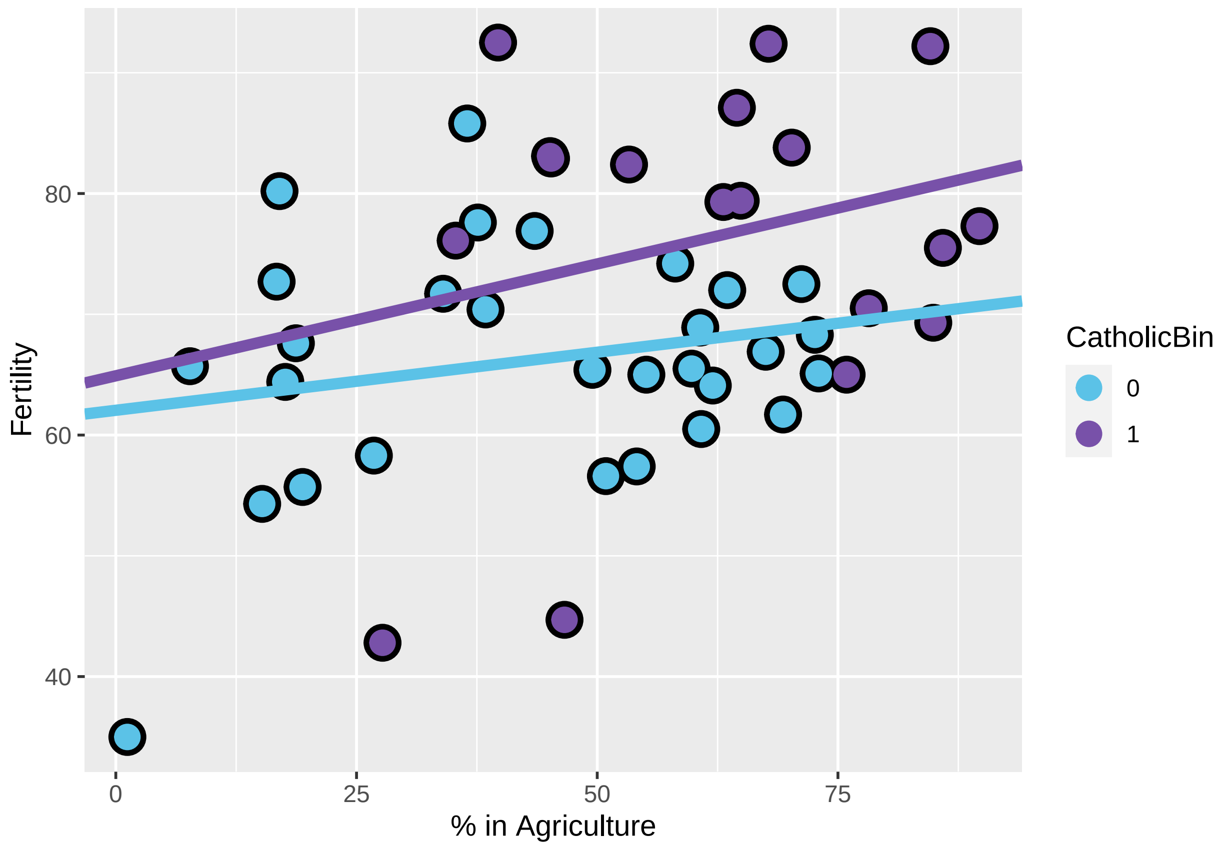
```
coef(interactfit)
```

```
##              (Intercept)              Agriculture             CatholicBin1
##              62.04993019               0.09611572               2.85770359
## Agriculture:CatholicBin1
##               0.08913512
```

- We can see here that `R` has automaticly made 3 variables: x_1, Agriculture; x_2, CatholicBin; and x_3, Agriculture*CatholicBin.

```
plot + geom_abline(intercept = coef(interactfit)[1],
                   slope = coef(interactfit)[2],
                   size = 2, colour = pal[1]) +
  geom_abline(intercept = coef(interactfit)[1] + coef(interactfit)[3],#Int.Var
```

```
            slope = coef(interactfit)[2] + coef(interactfit)[4],
            size = 2, colour = pal[2])
```



Lesson with swirl(): MultiVar Examples2

(No new content, review of `InsectSprays` example)

Lesson with swirl(): MultiVar Examples3

This lesson looks at WHO data on hunger but essentially analyzes it the same as the `swiss` dataset.

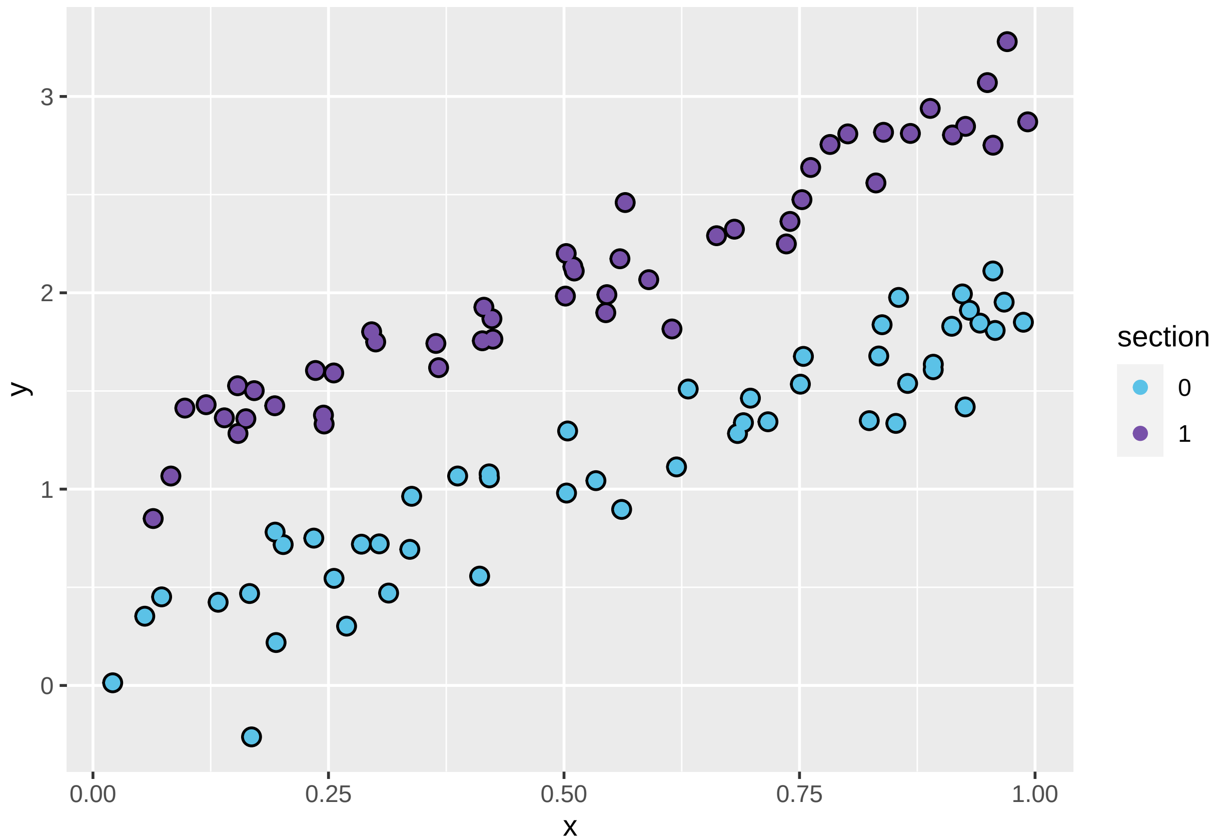
Adjustment

Adjustment Examples

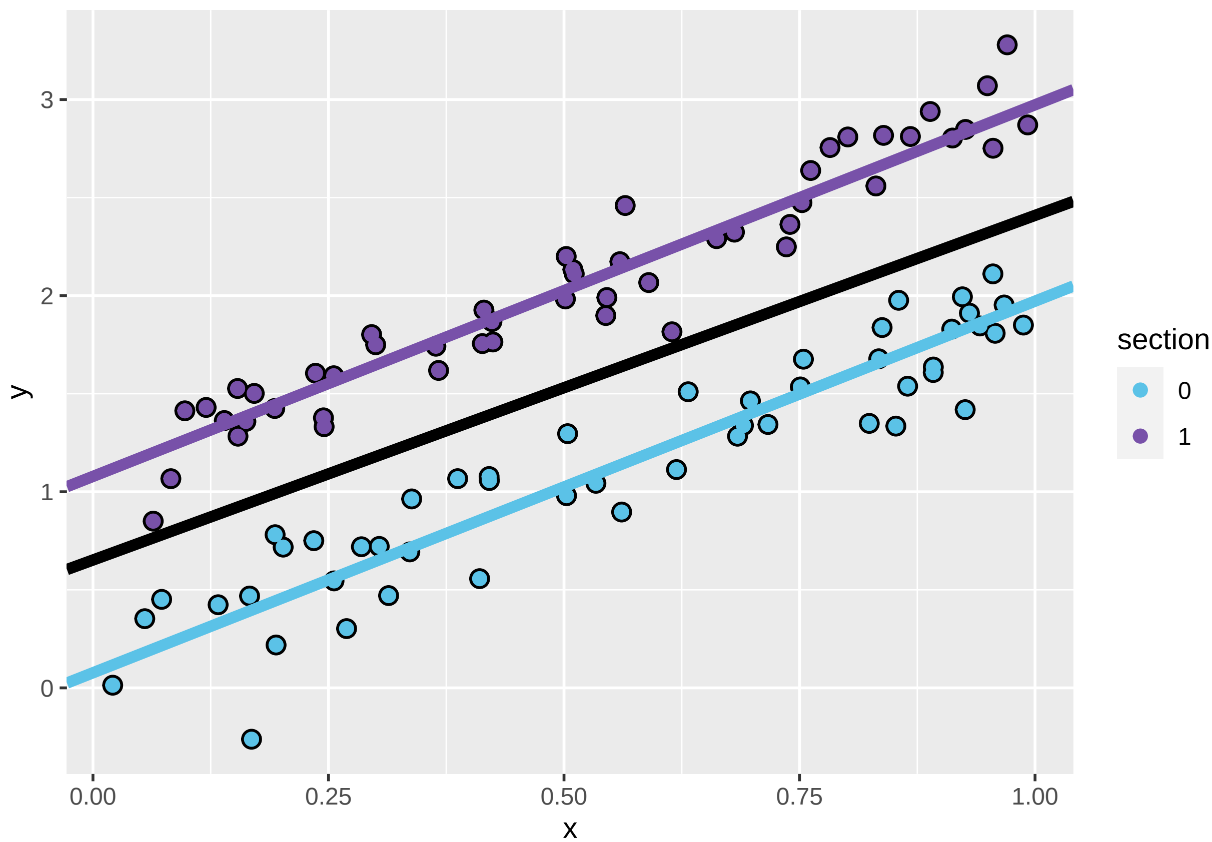
Simulation 1

```
set.seed(1618033)
n <- 100
t <- rep(c(0,1), each = n/2)
x <- c(runif(n/2), runif(n/2))
```

```r
beta0 <- 0
beta1 <- 2
tau <- 1
sigma <- 0.2
y <- beta0 + x * beta1 + t * tau +
  rnorm(n, sd = sigma) #For noise
plot <- ggplot(data.frame(x = x, y = y, section = factor(t)),
               aes(x,y, colour = section)) +
  scale_colour_manual(values = pal) +
  geom_point(size = 3, colour = "#000000") +
  geom_point(size = 2)
plot
```




```r
#Fiting a model to the points
xfit <- lm(y ~ x)
xtfit <- lm(y ~ x + t)
plot + geom_abline(intercept = xfit$coef[1], slope = xfit$coef[2],
                   colour = "#000000", lwd = 2) + # All points, no t
  #With resepect to t
  geom_abline(intercept = xtfit$coef[1], slope = xtfit$coef[2],
              colour = pal[1], lwd = 2) +
  geom_abline(intercept = xtfit$coef[1] + xtfit$coef[3], slope = xtfit$coef[2],
              colour = pal[2], lwd = 2)
```



- The difference in the coloured and black lines is 0.1377 d
- Some things to note about this simulation
  - The X cariable is unrelated to group status, t

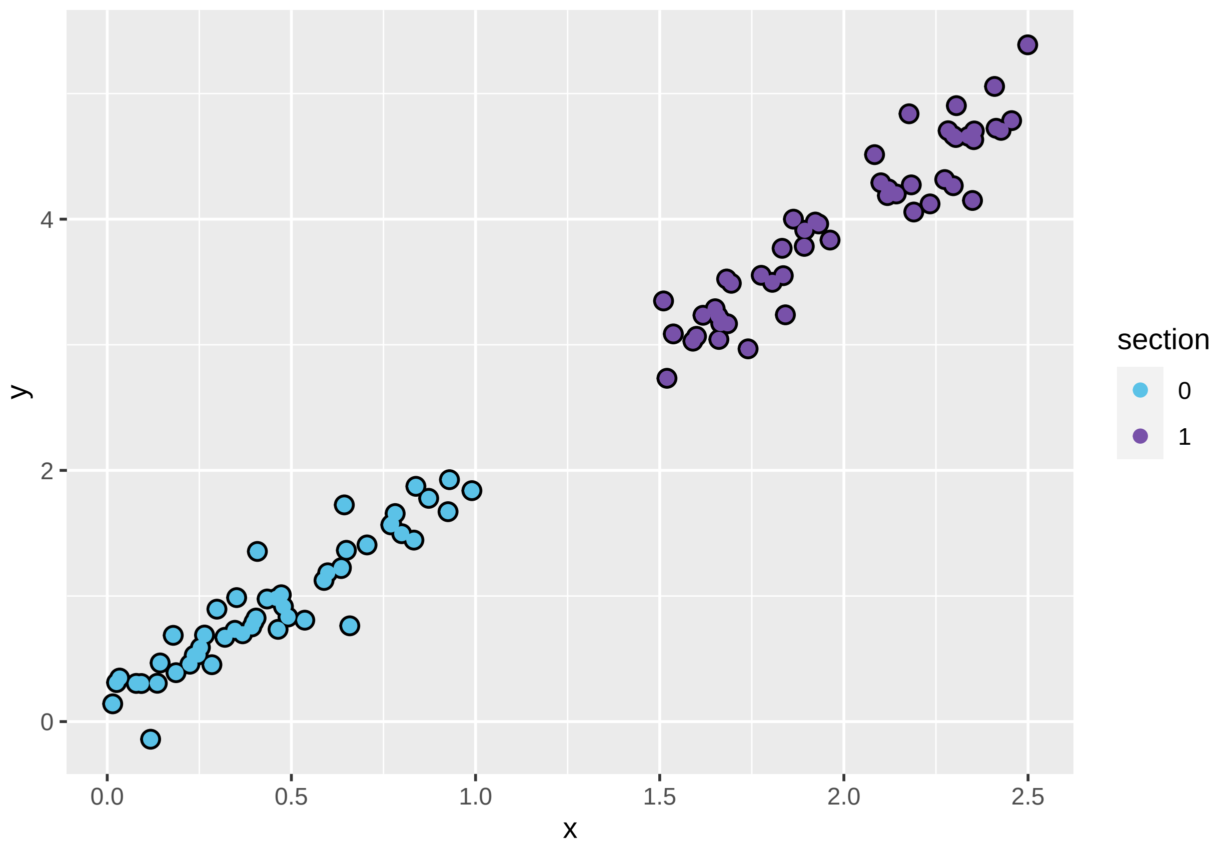
  - The X variable is related to Y, but the intercept depends on group status

  - The group variable is related to Y
    - The relationship between group status and Y is constant depending on X

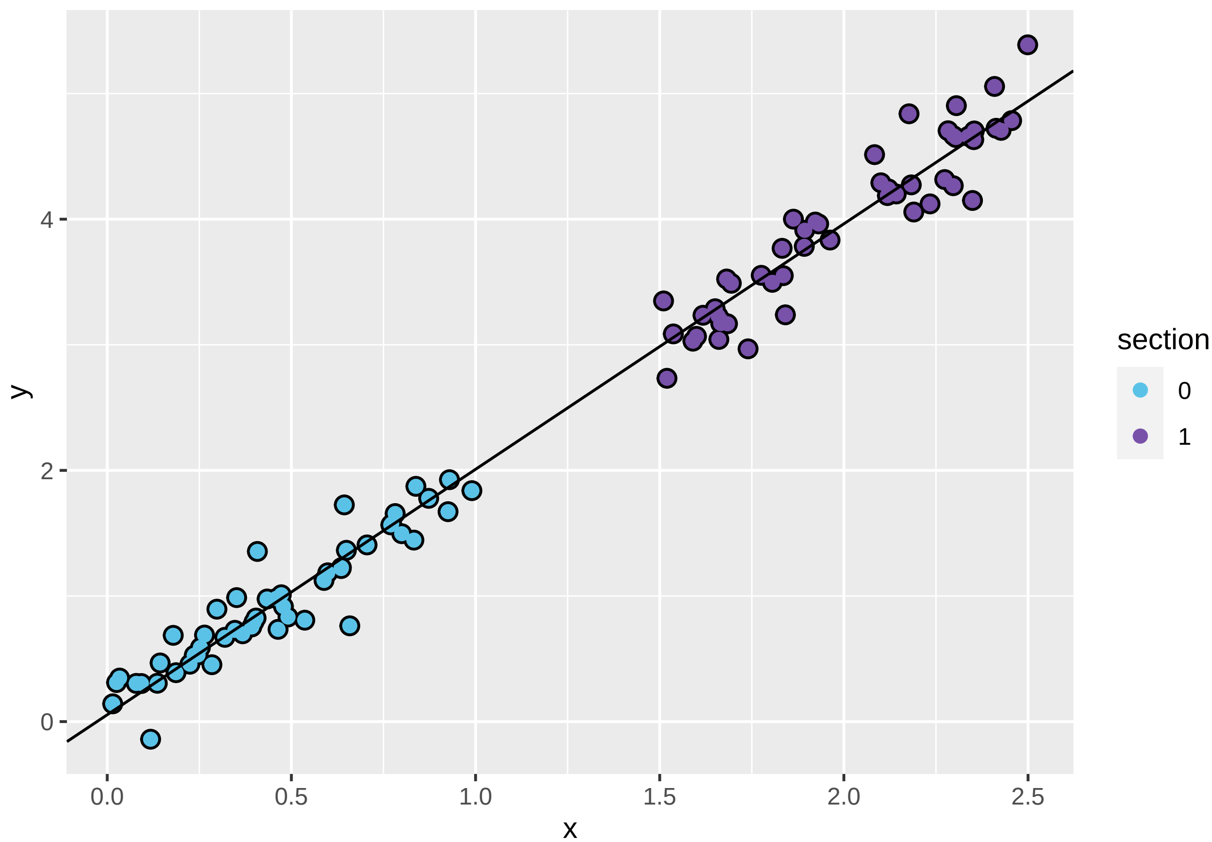
    - The relationship between group and Y disregarding X is about the same as holding X constant

Simulation 2

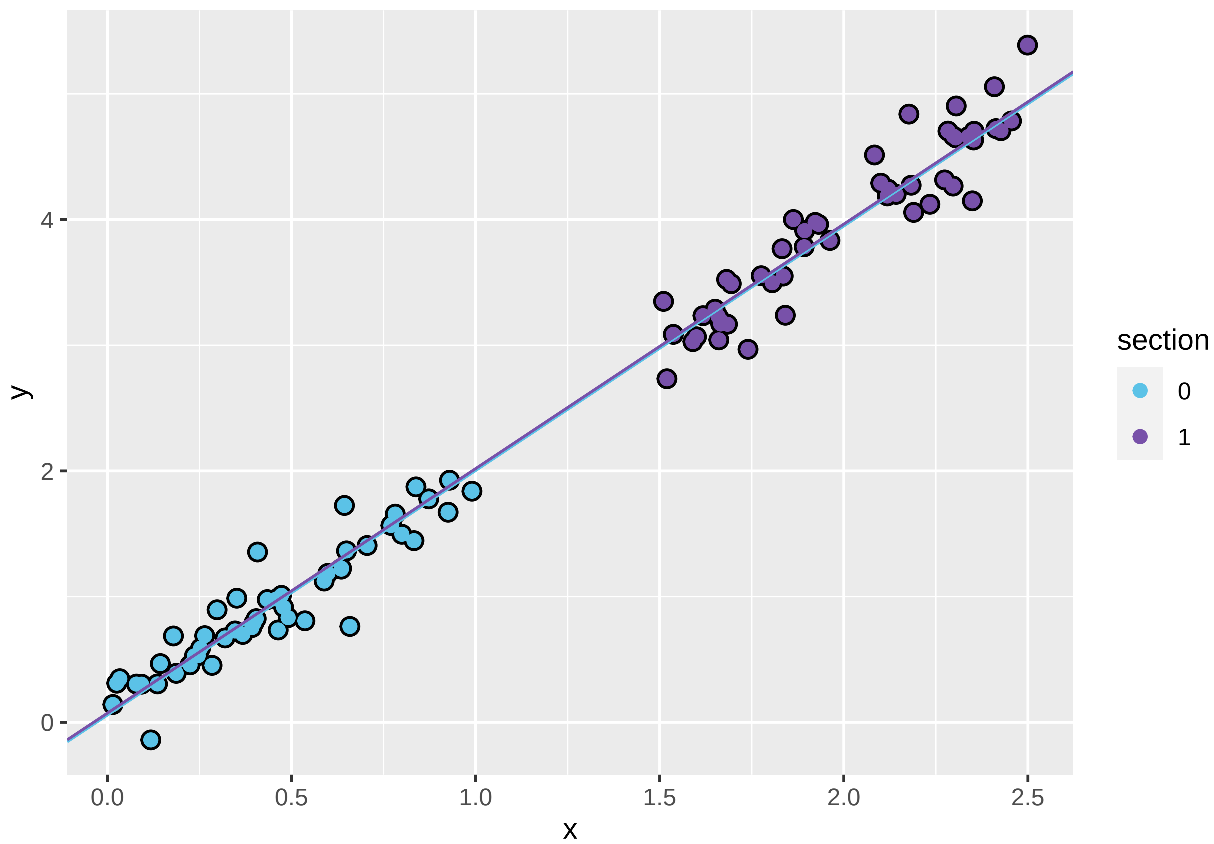
```
x <- c(runif(n/2), 1.5 + runif(n/2))
tau <- 0
y <- beta0 + x * beta1 + t * tau + rnorm(n, sd = sigma)
plot <- ggplot(data.frame(x = x, y = y, section = factor(t)),
               aes(x,y, colour = section)) +
  scale_colour_manual(values = pal) +
  geom_point(size = 3, colour = "#000000") +
  geom_point(size = 2)
plot
```




```
xfit <- lm(y ~ x)
xtfit <- lm(y ~ x + t)
plot + geom_abline(intercept = xfit$coef[1], slope = xfit$coef[2])
```




```r
plot +
  geom_abline(intercept = xtfit$coef[1], slope = xtfit$coef[2],
              colour = pal[1]) +
  geom_abline(intercept = xtfit$coef[1] + xtfit$coef[3],
              slope = xtfit$coef[2], colour = pal[2])
```




```r
xfit$coefficients
```

```
## (Intercept)           x
##  0.05432115  1.95436653
```

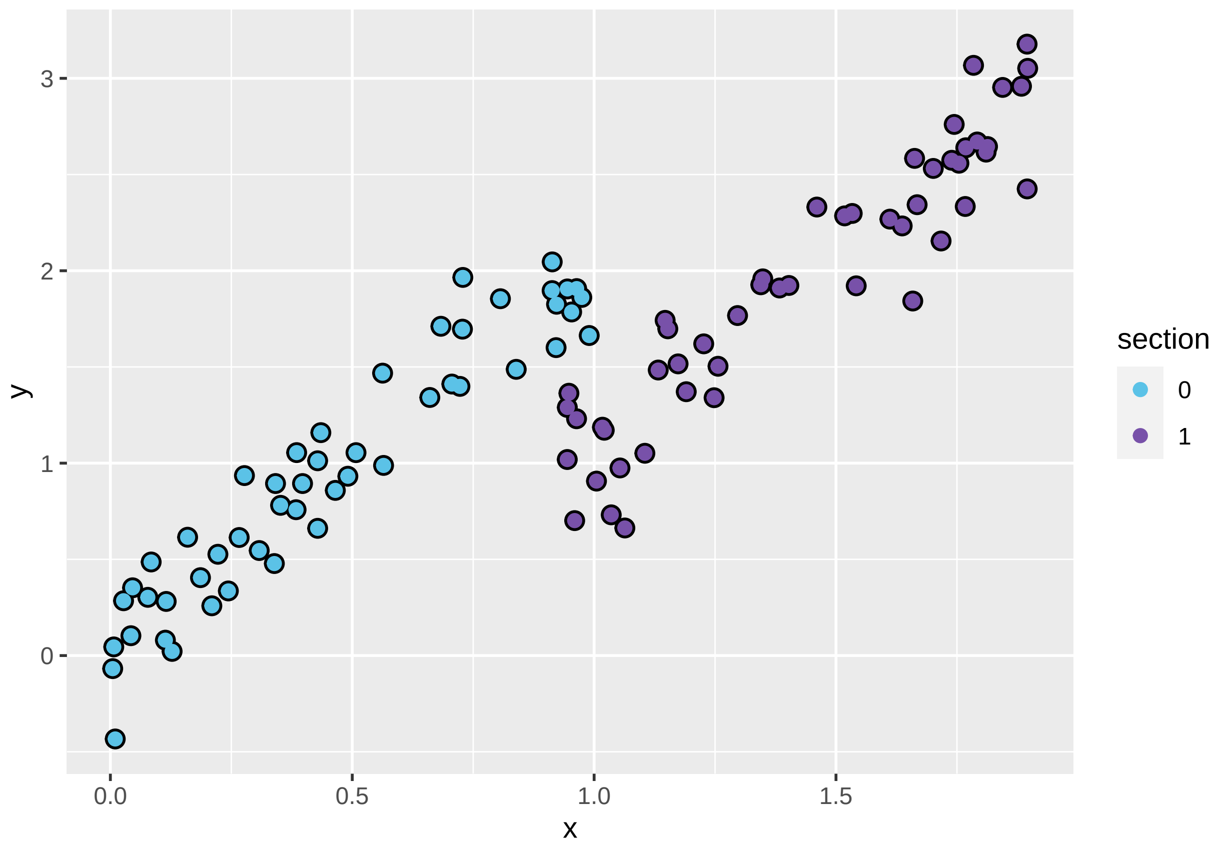
```r
xtfit$coefficients
```

```
## (Intercept)           x           t
##  0.05743077  1.94549763  0.01551007
```

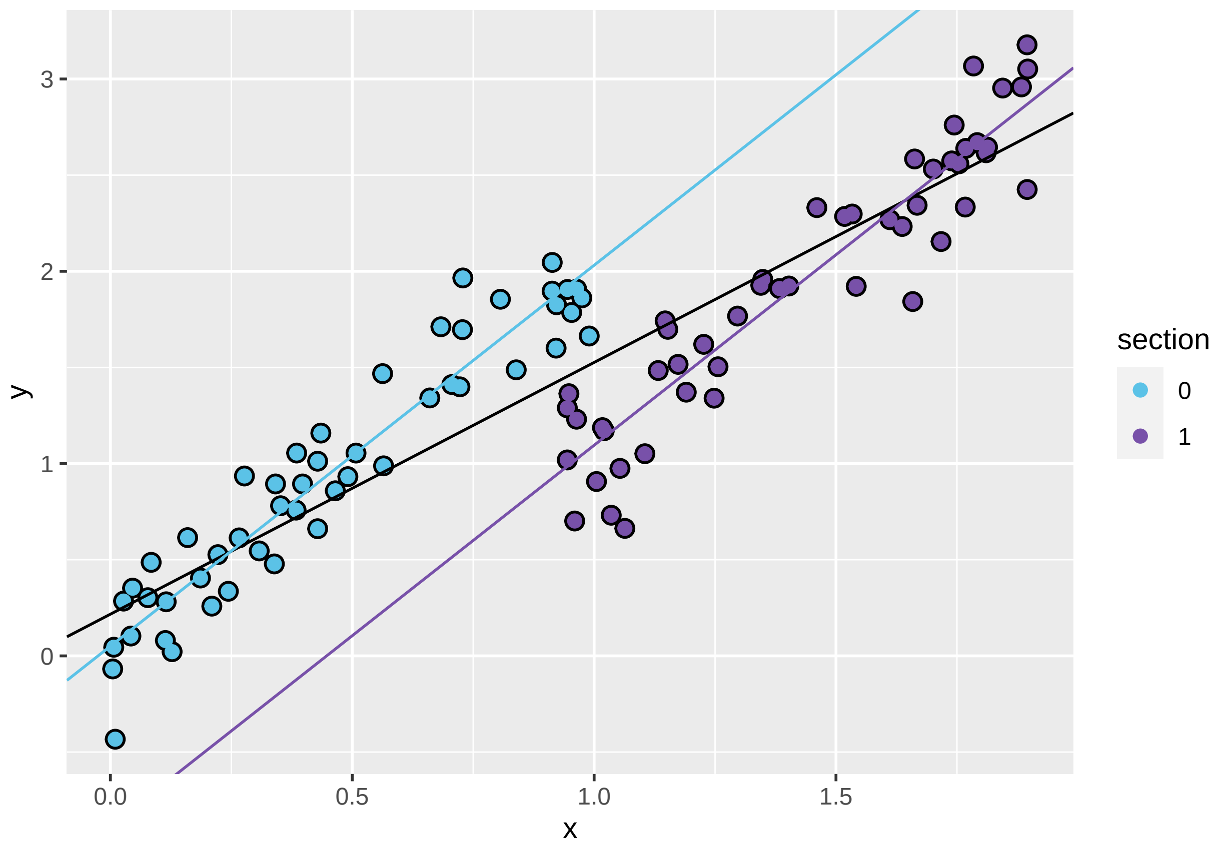
- These lines look nearly the same with only a slightly different intercept. However, the X variable is highly related to group status, below 1 is section `0` and above 1.5 is section `1`.
- The X variable is realted to Y, and the intercept does not depend on the group varaible
  - The X variable remains related to Y holding group status constant
- The group variable is marginally related to Y disregarding X
- The model would estimate no adjusted effect due to group.
  - There isn't any data to inform the relationship between group and Y.
  - This conclusion is entirely based on the model
- This sort of model may occur if the x-variable is something related to being in the given section.

Simulation 3

```
x <- c(runif(n/2), 0.9 + runif(n/2))
tau <- -1
y <- beta0 + x * beta1 + t * tau + rnorm(n, sd = sigma)
plot <- ggplot(data.frame(x = x, y = y, section = factor(t)),
               aes(x,y, colour = section)) +
  scale_colour_manual(values = pal) +
  geom_point(size = 3, colour = "#000000") +
  geom_point(size = 2)
plot
```




```
##Fitting a model
xfit <- lm(y ~ x)
xtfit <- lm(y ~ x + t)
plot + geom_abline(intercept = xfit$coef[1], slope = xfit$coef[2]) +
  geom_abline(intercept = xtfit$coef[1], slope = xtfit$coef[2],
              colour = pal[1]) +
  geom_abline(intercept = xtfit$coef[1] + xtfit$coef[3],
              slope = xtfit$coef[2], colour = pal[2])
```

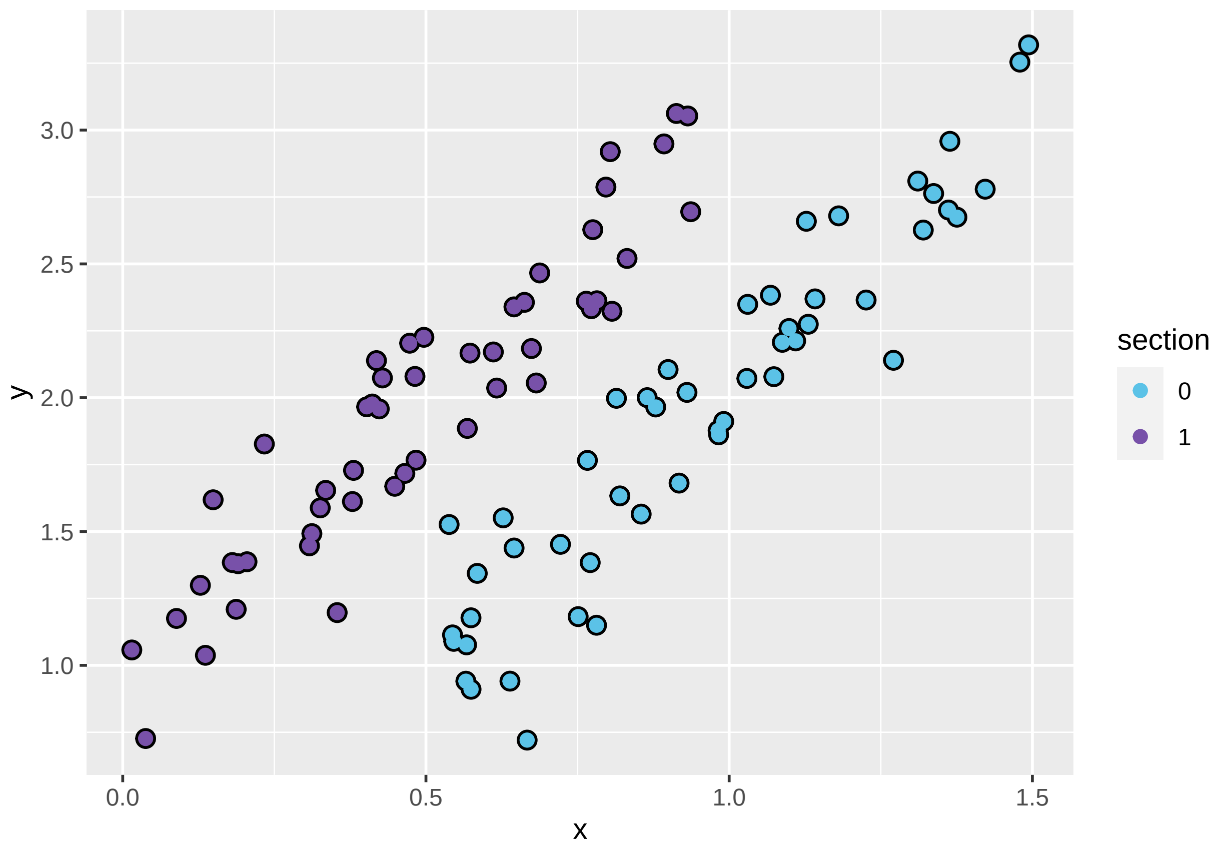


- Here we can see subsetting by group gives a clearly diffrent slope and still has some shared x values between groups. They also have diffrent intercepts

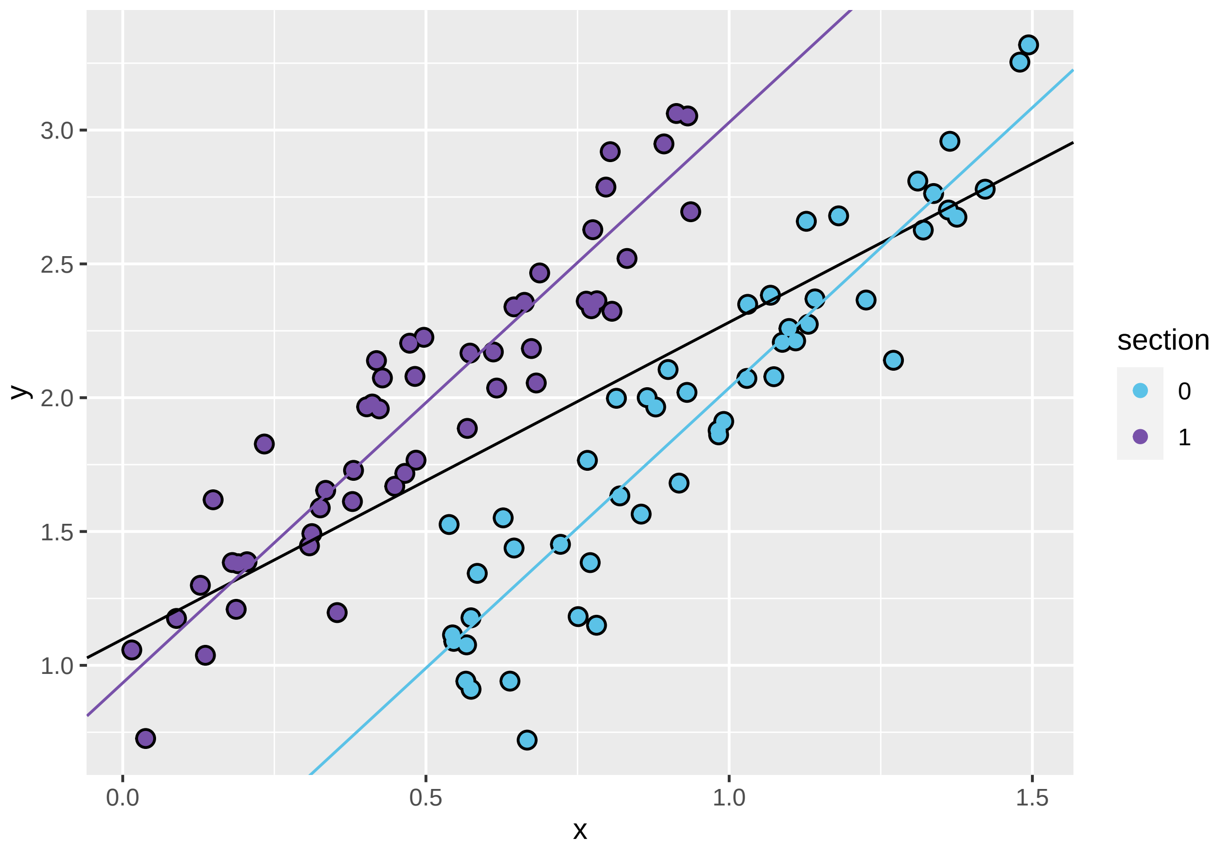
- This is an example of Simpson's "paradox"

Simulation 4

```
x <- c(0.5 + runif(n/2), runif(n/2))
tau <- 1
y <- beta0 + x * beta1 + t * tau + rnorm(n, sd = sigma)
plot <- ggplot(data.frame(x = x, y = y, section = factor(t)),
               aes(x,y, colour = section)) +
  scale_colour_manual(values = pal) +
  geom_point(size = 3, colour = "#000000") +
  geom_point(size = 2)
plot
```



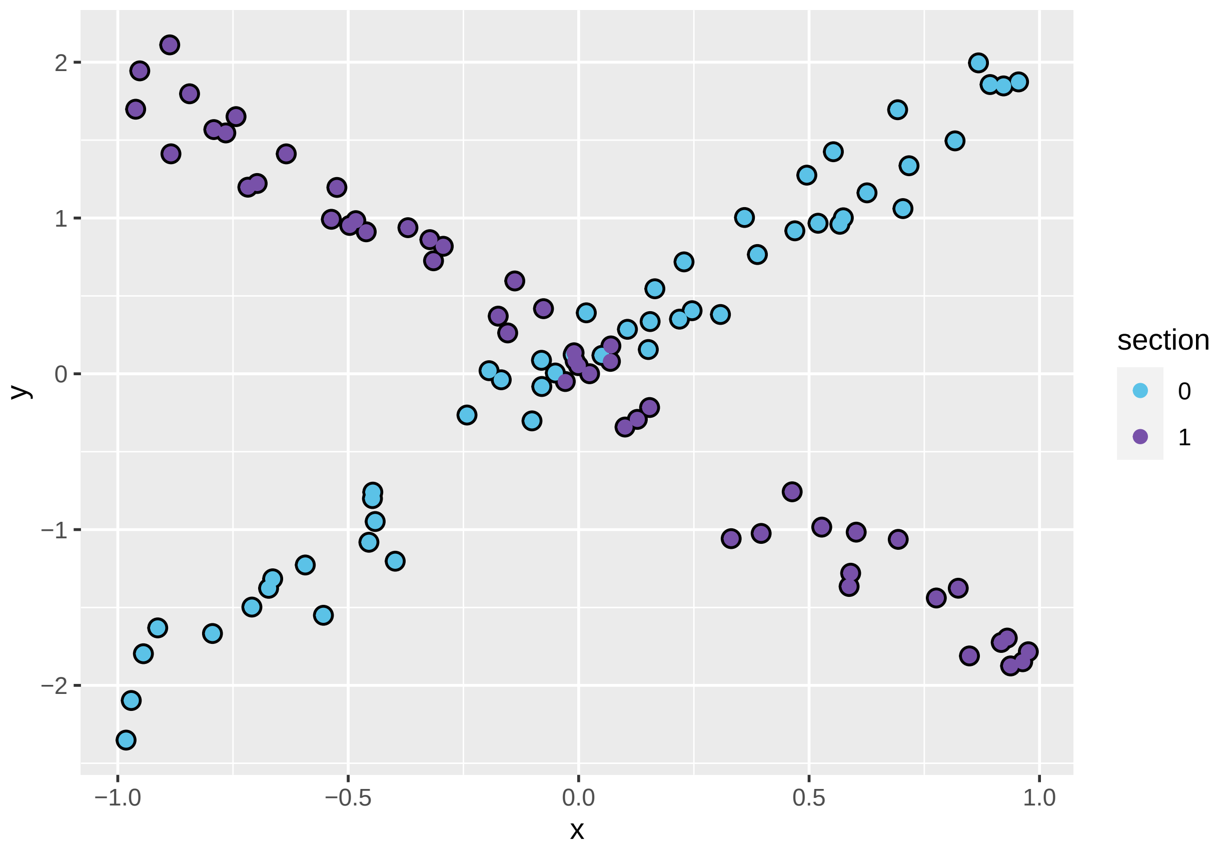

```
##Fitting a model
xfit <- lm(y ~ x)
xtfit <- lm(y ~ x + t)
plot + geom_abline(intercept = xfit$coef[1], slope = xfit$coef[2]) +
  geom_abline(intercept = xtfit$coef[1], slope = xtfit$coef[2],
              colour = pal[1]) +
  geom_abline(intercept = xtfit$coef[1] + xtfit$coef[3],
              slope = xtfit$coef[2], colour = pal[2])
```



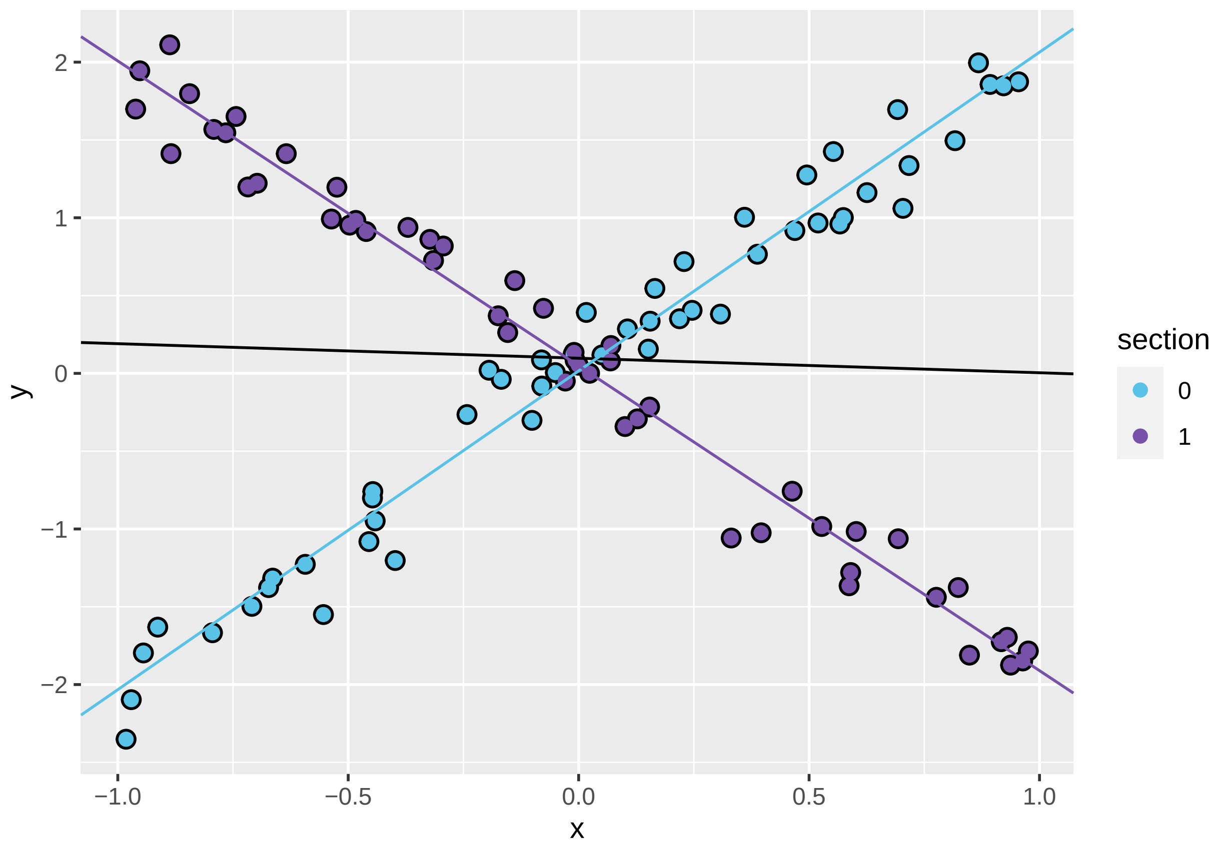
- In this example there is a large affect when adjusted for x, however the marginal affect is not great with mean(section==0) = 1.9586 and mean(section==1) = 1.8103. Group status is not realted to X much with all the overlap.

Simulation 5

```
x <- c(runif(n/2, -1, 1), runif(n/2, -1, 1))
tau <- 0
tau1 <- -4
y <- beta0 + x * beta1 + t * tau + t * x * tau1 + rnorm(n, sd = sigma)
plot <- ggplot(data.frame(x = x, y = y, section = factor(t)),
               aes(x,y, colour = section)) +
  scale_colour_manual(values = pal) +
  geom_point(size = 3, colour = "#000000") +
  geom_point(size = 2)
plot
```




```
##Fitting a model
xfit <- lm(y ~ x)
xtfit <- lm(y ~ x + t + I(x * t))
plot + geom_abline(intercept = xfit$coef[1], slope = xfit$coef[2]) +
  geom_abline(intercept = xtfit$coef[1], slope = xtfit$coef[2],
              colour = pal[1]) +
  geom_abline(intercept = xtfit$coef[1] + xtfit$coef[3],
              slope = xtfit$coef[2] + xtfit$coef[4],
              colour = pal[2])
```




```
c(mean(y[1:(n/2 - 1)]), mean(y[n/2:n]))
```

```
## [1] 0.07857856 0.28650081
```

- Here the marginal difference is quite small, however looking at the trend by group shows that section `1` has a decrease in Y as X increases. However the intercept is quite high. As such one couldn't advise for a treatment if an individual had a low x.

Some final thoughts

- Modeling multivariate relationships is difficult
- Play around with simulations to see how the inclusion or exclusion of another variable can change analyses
- The results of these analyses deal witht eh impact of variables on associations
  - Ascertaining mechanisms or cause are difficult subjects to be added on top of difficulty in understanding multivariate associations

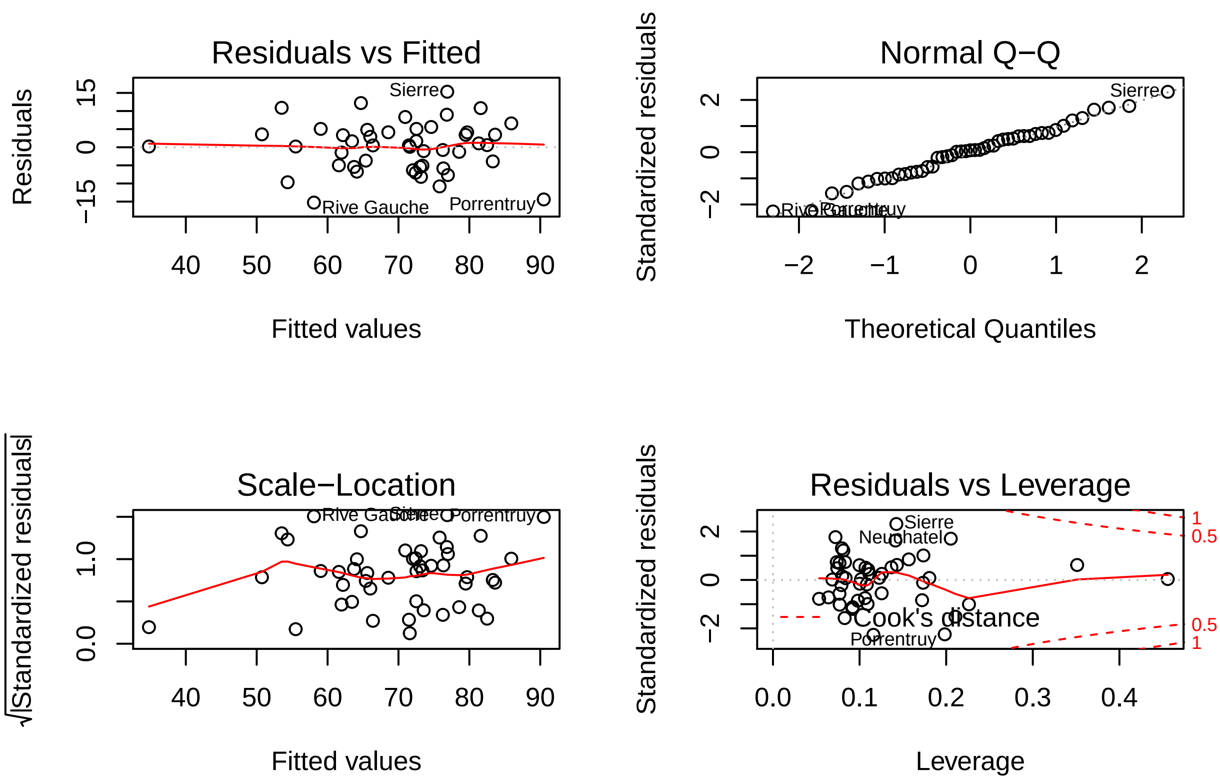
Residuals Again

Residuals and Diagnostics Part 1

The Linear Model

- Specified as $Y_i = \sum_{k=1}^{p} X_{ik}\beta_j + \epsilon_i$
- We'll also assume here that $\epsilon_i \sim^{iid} N(0, \sigma^2)$
- The residuals are defined as $e_i = Y_i - \hat{Y}_i = Y_i - \sum_{k=1}^{p} X_{ik}\hat{\beta}_j$
- Our estimate of residual variation is $\hat{\sigma}^2 = \frac{\sum_{i=1}^{n} e_i^2}{n-p}$, the $n - p$ is such that $E[\hat{\sigma}^2] = \sigma^2$

```
data(swiss); par(mfrow = c(2,2))
fit <- lm(Fertility ~ . , data = swiss)
#R will splot the fit with a series of residual and diagnostic plots
plot(fit)
```



- Residuals vs. Fitted
  - Plots residuals against the predicted (fitted) values.
- Normal Q-Q
  - Tests for normallity of error terms, plots true z-scores versus normalized z-scores

- Scale-Location
  - Plots standardized residuals against fitted values, scaled so they are like a t-statistic

- Residuals vs Leverage
  - Plots standardized residuals against their leverage

Leverage

- Points that are far from the center of data have a lot of leverage, simular to a weight on the end of a lever having more leverage than if it's near the fulcrum

Residuals and Diagnostics Part 2

- Calling a point an outlier is vague.
  - They can be the result of real or spurious processes.
    * If they real we don't want to just get rid of it

    * If they are spurious and their is evidence for this we can safely remove it from the model

- Outliers can conform to the regression relationship (being marginally outlying in X or Y, but not outlying given the regression relationship)

- There are various ways we can investigate outliers to ascertain how much influence they have.

- A list of default diagnostic measures can be obtained by executing `?incluence.measures`
  - `rstandard` - standardized residuals, (residuals divided by their standard deviations)

  - `rstudent` - standardized residuals, residuals divided by their standard deviations, where the ith data point was deleted in the calculation of the standard deviation for the residual to follow a t distribution

  - `hatvalues` - measures of leverage
    * Useful in finding data entry errors

  - `dffits` - change in the predicted response when the i^{th} point is deleted in fitting the model. Helps measure for influence

  - `dfbetas` - change in individual coefficients when the i^{th} point is deleted in fitting the model. Helps measure for influence
    * Returns a matrix of two values, slope & intercept

  - `cooks.distance` - overall change in the coefficients when the i^{th} point is deleted.
    * In a sense, summarizes `dfbetas`

  - `resid` - returns the ordinary residuals

  - `resid(fit) / (1 - hatvalues(fit))` where `fit` is the linear model fit returns the PRESS residuals, i.e. the leave one out cross validation residuals - the difference in the response and the predicted response at data point i, where it was not included in the model fitting.

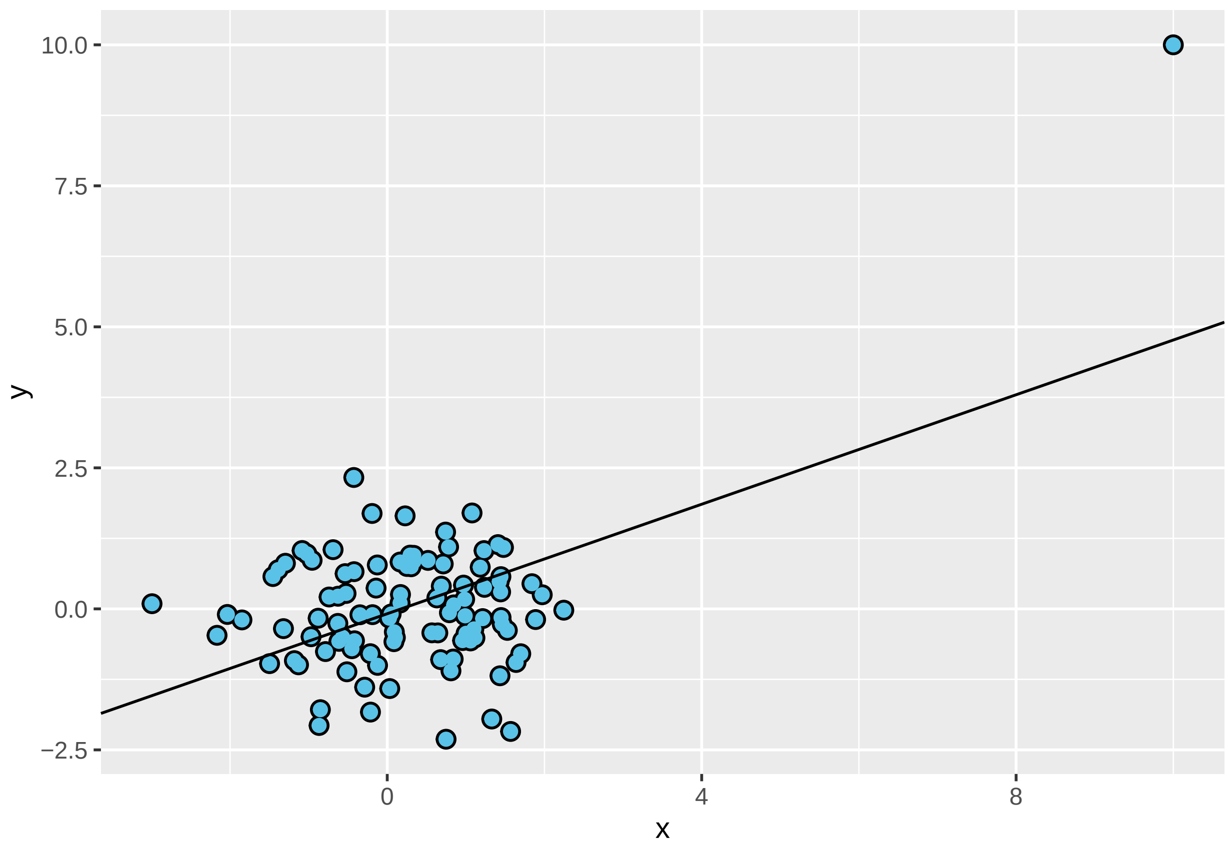
- One can not just set a residual threshold and exclude points outside of it as that is akin to P-hacking
  - However one can declare points outliers from the residual, utilizing `rstandard` or `rstudent`

  - Although, it is better to look at them with respect to the overall cloud of data rather than just setting strict thresholds
- Linear algebra exploits are used such that the model doesn't have to be refit when using these models

Residuals and Diagnostics Part 3

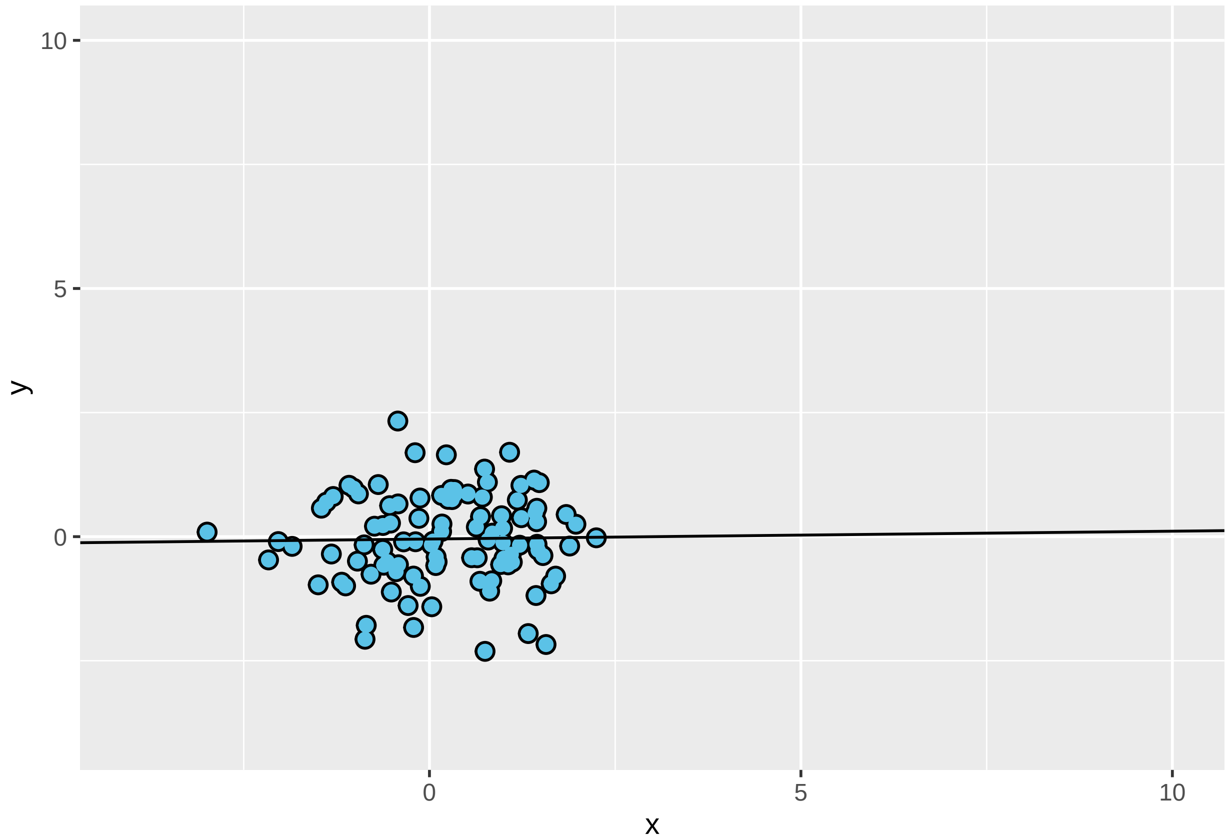
Simulation 1

```
set.seed(1618033)
n <- 100
x <- c(10, rnorm(n))
y <- c(10, rnorm(n))
plot <- ggplot(data.frame(x = x, y = y), aes(x,y)) +
  geom_point(size = 3, colour = "#000000") +
  geom_point(size = 2, colour = "#5BC2E7")
fit <- lm(y ~ x)
plot + geom_abline(intercept = fit$coef[1], slope = fit$coef[2])
```



- The point `(10,10)` has created a strong regression relationship where there shouldn't be one

```
explot <- ggplot(data.frame(x = x[-1], y = y[-1]), aes(x,y)) +
  geom_point(size = 3, colour = "#000000") +
  geom_point(size = 2, colour = "#5BC2E7") +
  xlim(c(-4,10)) + ylim(c(-4,10))
exfit <- lm(y[-1]~x[-1])
explot + geom_abline(intercept = exfit$coef[1], slope = exfit$coef[2])
```



Diagnostic values:

```
round(dfbetas(fit)[1:10,2], 3) #Slope dif.
```

```
##      1      2      3      4      5      6      7      8      9     10
##  7.404  0.010  0.011  0.012  0.004 -0.125 -0.142 -0.008 -0.003 -0.148
```

```
round(hatvalues(fit)[1:10], 3) #
```

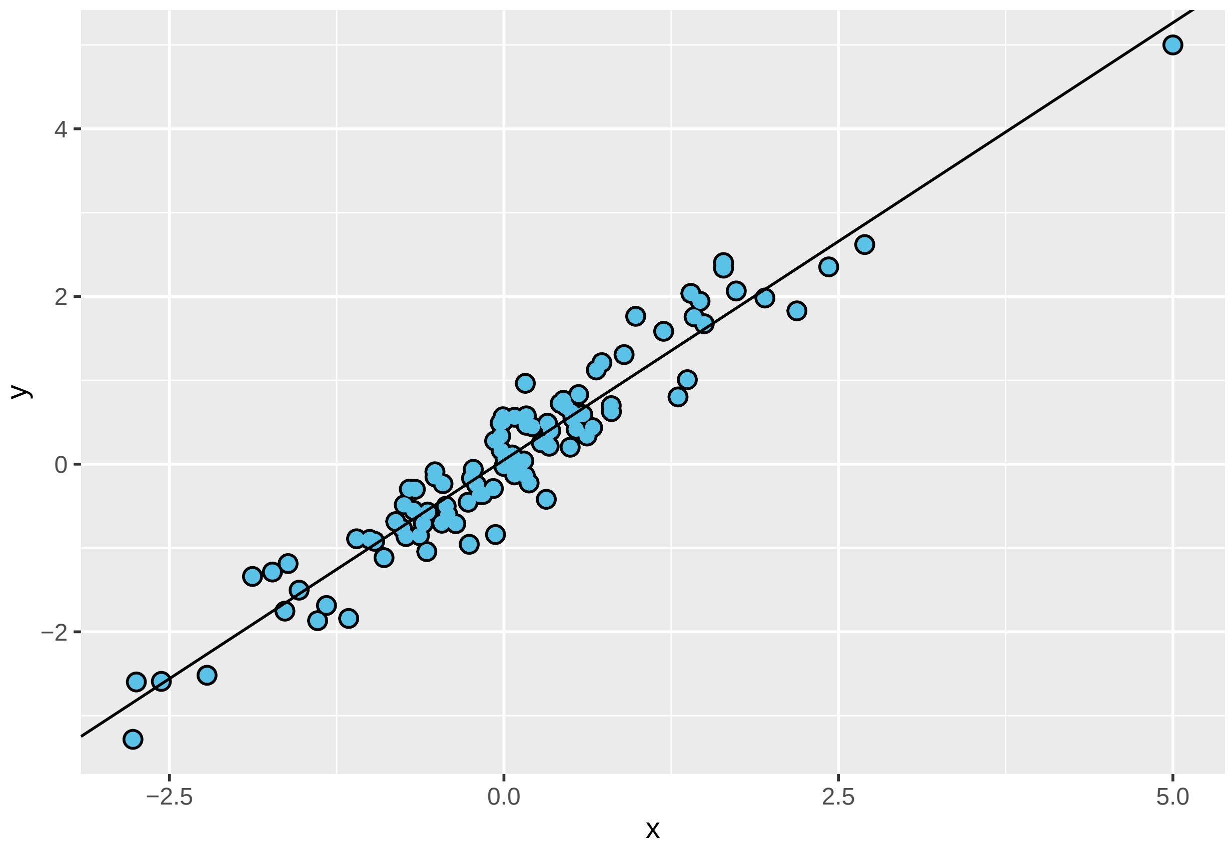
```
##     1     2     3     4     5     6     7     8     9    10
## 0.471 0.010 0.014 0.012 0.011 0.029 0.036 0.014 0.016 0.025
```

- Notice the first point in both instances are larger than the rest, indicating the first point's influence

Simulation 2

```
x <- rnorm(n)
y <- x + rnorm(n, sd = 0.3)
x <- c(5, x)
y <- c(5, y)
plot <- ggplot(data.frame(x = x, y = y), aes(x,y)) +
  geom_point(size = 3, colour = "#000000") +
  geom_point(size = 2, colour = "#5BC2E7")
fit <- lm(y ~ x)
```

```
plot + geom_abline(intercept = fit$coef[1], slope = fit$coef[2])
```



- Here the outlier has a lot of leverage but adheres nicely to our model
  Diagnostic values:

```
round(dfbetas(fit)[1:10,2], 3) #Slope dif.
```

```
##      1      2      3      4      5      6      7      8      9     10
## -0.410 -0.057 -0.018 -0.075  0.011  0.242 -0.002 -0.057 -0.014 -0.290
```

```
round(hatvalues(fit)[1:10], 3) #
```

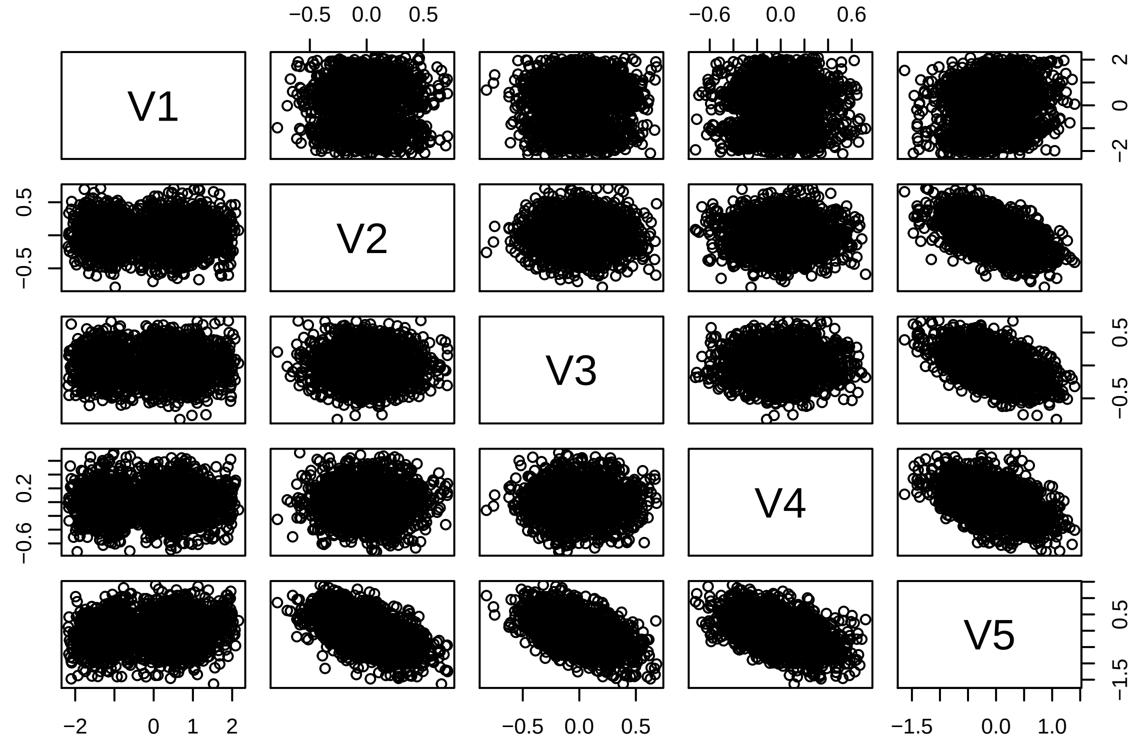
```
##     1     2     3     4     5     6     7     8     9    10
## 0.194 0.012 0.010 0.014 0.010 0.029 0.010 0.010 0.011 0.038
```

- Here the first value is still large but not as much as previous

- The hatvalue is larger since its outside of the x valeus but still adheres to the realtionship

Using Residuals to See Underlying Patterns

```
dat <- read.table('http://www4.stat.ncsu.edu/~stefanski/NSF_Supported/Hidden_Images/orly_owl_fi
pairs(dat) #Not much detail due to overplotting
```

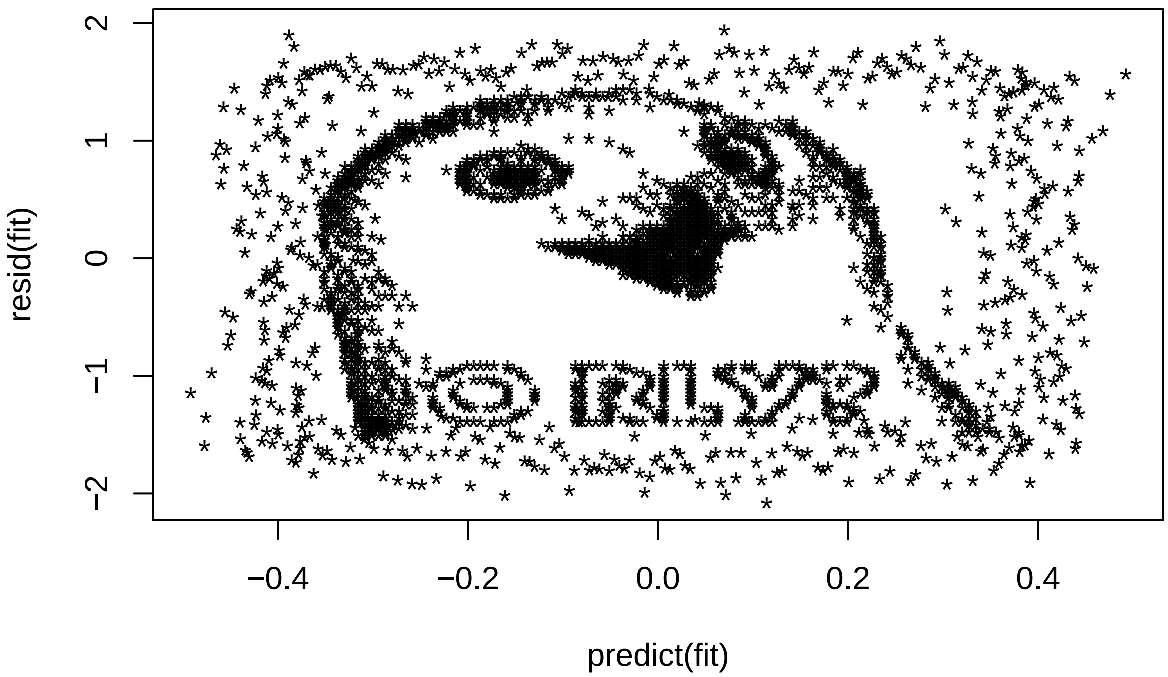



```r
#Looking at P-values
fit <- lm(V1 ~ . - 1, dat)
summary(fit)$coef
```

```
##     Estimate Std. Error   t value     Pr(>|t|)
## V2 0.9856157 0.12798121  7.701253 1.989126e-14
## V3 0.9714707 0.12663829  7.671225 2.500259e-14
## V4 0.8606368 0.11958267  7.197003 8.301184e-13
## V5 0.9266981 0.08328434 11.126919 4.778110e-28
```

The fit shows that all the variables are significant, but we should still look at the residuals to see if there is an underlying pattern

```r
plot(predict(fit), resid(fit), pch = '*')
```



And by plotting the residuals we can see that although the data is significant there is an underlying pattern

Lesson with `swirl()`: Residuals Diagnostics and Variation

(No new content)

Model Selection

How do we chose what variables to include in a regression model?
* It depends, but that's what this section aims to look at

Model Selection Part 1

Multivariable Regression

- The next course is on prediction and machine learning, here we're focusing on modeling
  - Prediction has a different set of criteria, needs for interpretability are less

  - In modeling, out interest lies in parsimonious, as simple as possible but no simplier. Interpretable representations enhance our understanding of the phenomena beneath the data

  - "A model is a lense through which to look at your data" - Scott Zeger
    * Under this philosophy, what's the right model? Whatever model connects the data to a true, parsimonious statement about what you're studying

- There are nearly uncountable ways that a model can be wrong, this can be adjusted with varaible inclusion and exclusion

The Rumsfeldian Triplet

"*There are known knowns. These are things we know that we know. There are known unknowns. That is to say, there are things that we know we don't know. But there are also unknowns. There are things we don't know we don't know.*" -Donald Rumsfeld
In terms of regression:
* (Known knowns) Regressors that we know we should check to include in the model and we have.
* (Known Unknowns) Regressors that we would like to include in the model, but simply don't have.
* (Unknown Unknowns) Regressors that we don't even know about that we should have included in the model.

General Rules

- Omitting variables results in bias in the coefficients of interest - unless their regressors are uncorrelated with the omitted ones.
  - This is why we randomize treatments, it attempts to uncorrelate our treatment indicator with variables that we don't have access to to put in the model

  - (If there's too many unobserved confounding variables, even randomization won't help you.)

- Including variables that we shouldn't have increases standard errors of the regression variables
  - Including any new variables increases (actual, not estimated) standard errors of other regressors. So we don't want to idly throw variables into the model.

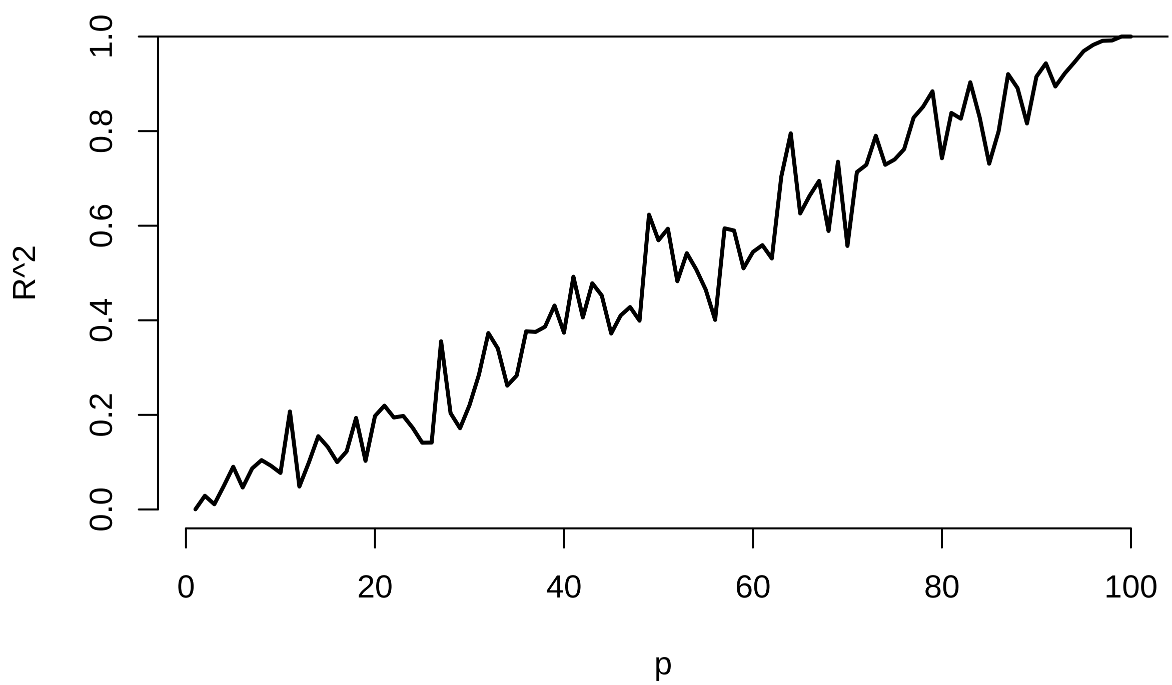
- The model must tend toward perfect fit as the number of non-redundant regressors approaches n

- R^2 increases monotonically as more regressors are included

- The mean square error decreases monotonically as more regressors are included
  Below is a plot of R^2 versus n where there is no true correlation underlying

```
set.seed(1618033)
n <- 100
plot(c(1, n), 0:1, type = "n", frame = FALSE, xlab = "p", ylab = "R^2")
r <- sapply(1:n, function(p)
      {
        y <- rnorm(n); x <- matrix(rnorm(n * p), n, p)
        summary(lm(y ~ x))$r.squared
      }
    )
```

```r
lines(1:n, r, lwd = 2)
abline(h = 1)
```



As we can see it trends towards 1, even though it's just noise being added

Model Selection Part 2

Variance Inflation

```r
set.seed(1618033)
n <- 100
nosim <- 1000
x1 <- rnorm(n); x2 <- rnorm(n); x3 <- rnorm(n);
betas <- sapply(1:nosim, function(i){
  y <- x1 + rnorm(n, sd = .3)
  ##Looking at variable for each added variable
  c(coef(lm(y ~ x1))[2],
    coef(lm(y ~ x1 + x2))[2],
    coef(lm(y ~ x1 + x2 + x3))[2])
})
round(apply(betas, 1, sd), 5) #Look at sd of regression models
```

```
##      x1      x1      x1
```

```
## 0.02898 0.02899 0.02899
```

- It can be seen that regardless of how many regressors we include the variance of the first coeficient does not increase
  Let's look at a case where each varaible relies on `x1`

```
x1 <- rnorm(n)
x2 <- x1/sqrt(2) + rnorm(n)/sqrt(2) #Slightly depends on x1
x3 <- x1 * 0.95 + rnorm(n) * sqrt(1 - 0.95^2);  #Heavily depends on x1
betas <- sapply(1:nosim, function(i){
  y <- x1 + rnorm(n, sd = .3)
  c(coef(lm(y ~ x1))[2],
    coef(lm(y ~ x1 + x2))[2],
    coef(lm(y ~ x1 + x2 + x3))[2])
})
round(apply(betas, 1, sd), 5)
```

```
##      x1      x1      x1
## 0.03076 0.04443 0.10256
```

- Here we can see an increase in the variance of the coefficients as we include more variables that are all dependent on x1
  - If the variable you include is highly correlated to what you're investigating you'll get a higher standard error
    - If you can avoid it that's ideal but if you have to map both variables (such as height and weight) you'll get this drawback of a higher standard error

Variance Inflation Factors

- Notice variance inflation was much worse when we included a variable that was highly related to `x1`.

- We don't know σ, so we can only estimate the increase int he actual standard error of the coefficients for including a regressor

- However, σ drops out of the relative standard erorr, If one sequentially adds variables, one cna check the variance inflation for including each one.

- When the other regressors are actually orthogonal to the regressor of interest, then there is no variance inflation

- The **variance inflation factor (VIF)** is the increase in the variance for the i^{th} regressor compared to the ideal setting where it is orthogonal to the other regressors
  - $\sqrt{VIF}$ = the increase in the sd

- Remember that variance inflation is only part of the picture. We want to include certain variables, even if they dramatically inflate our variance.

Variance Inflation with Swiss Data

```r
library(car)#Companion to Applied Regression
data(swiss)
```

```r
fit <- lm(Fertility ~ ., swiss)
res <- data.frame(rbind(vif(fit), sqrt(vif(fit))))
rownames(res) <- c("Variance", "SD")
res
```

```
##          Agriculture Examination Education Catholic Infant.Mortality
## Variance    2.284129    3.675420  2.774943 1.937160         1.107542
## SD          1.511334    1.917138  1.665816 1.391819         1.052398
```

- `[1,1]` indicates that the Standard Error (SSE) of the Agriculture is double than what it would be if it were orthoginal to all the other regressors
- Infant.Mortality is low because it likely is already orthoginal to the other factors

Model Selection Part 3

What About Residual Variance Estimation?

- Assuming that the model is linear with additive iid errors (with finite variance), we can mathematically describe the impact of omitting necessary varaibles or including unnecessary ones.
  - If we underfit the model, the variance estimate is biased due to that missing factor
  - If we correctly or overfit the model, including all necessary covariates and/or unnecessary covariates, the variance estiamte is unbiased
    - However, the variance of the variance is larger if we include unnecessary variables
- Principal componetns or factor analytic models on covariates are often udeful for reducing complex covariate spaces.
- Good design can often eliminate the need for complex model searches at analyses; though often contorl over the design is limited.
- If the models of interest are nested and without lots of parameters differentiating them, it's fairly uncontroversial to use nested likelihood ratio tests.

Nested Model Testing

```r
fit1 <- lm(Fertility ~ Agriculture, swiss)
fit3 <- update(fit, Fertility ~ Agriculture + Examination + Education)
fit5 <- update(fit, Fertility ~ Agriculture + Examination + Education + Catholic + Infant.Morta
##Once can use ANOVA to perform an ANalyis Of the VAriance
anova(fit1, fit3, fit5)
```

```
## Analysis of Variance Table
```

```
##
## Model 1: Fertility ~ Agriculture
## Model 2: Fertility ~ Agriculture + Examination + Education
## Model 3: Fertility ~ Agriculture + Examination + Education + Catholic +
##     Infant.Mortality
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     45 6283.1
## 2     43 3180.9  2    3102.2 30.211 8.638e-09 ***
## 3     41 2105.0  2    1075.9 10.477 0.0002111 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- `Res.Df` - Number of Observations $-$ Number of Parameters
- `RSS` - Residual Sum of Squares
- `Sum of Sq` - How many addtional Df were added from previous model
- `F` - F-statistic
- `P...` - If significant it implies that the additional inclusion appears to be nessecary

Practice Exercise in Regression Modeling

My practice exercise can be found on GitHub

Quiz 3

1. Consider the mtcars data set. Fit a model with mpg as the outcome that includes number of cylinders as a factor variable and weight as confounder. Give the adjusted estimate for the expected change in mpg comparing 8 cylinders to 4.

```r
data(mtcars)
fit <- lm(mpg ~ factor(cyl) + wt, mtcars)
summary(fit)$coef
```

```
##                Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)  33.990794  1.8877934 18.005569 6.257246e-17
## factor(cyl)6 -4.255582  1.3860728 -3.070244 4.717834e-03
## factor(cyl)8 -6.070860  1.6522878 -3.674214 9.991893e-04
## wt           -3.205613  0.7538957 -4.252065 2.130435e-04
```

- -6.071

2. Consider the mtcars data set. Fit a model with mpg as the outcome that includes number of cylinders as a factor variable and weight as a possible confounding variable. Compare the effect of 8 versus 4 cylinders on mpg for the adjusted and unadjusted by weight models. Here, adjusted means including the weight variable as a term in the regression model and unadjusted

means the model without weight included. What can be said about the effect comparing 8 and 4 cylinders after looking at models with and without weight included?

```r
cylfit <- lm(mpg ~ factor(cyl), mtcars)
summary(fit)$coef
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)  33.990794  1.8877934 18.005569 6.257246e-17
## factor(cyl)6 -4.255582  1.3860728 -3.070244 4.717834e-03
## factor(cyl)8 -6.070860  1.6522878 -3.674214 9.991893e-04
## wt           -3.205613  0.7538957 -4.252065 2.130435e-04
```

```r
summary(cylfit)$coef
```

```
##                Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)   26.663636  0.9718008 27.437347 2.688358e-22
## factor(cyl)6  -6.920779  1.5583482 -4.441099 1.194696e-04
## factor(cyl)8 -11.563636  1.2986235 -8.904534 8.568209e-10
```

3. Consider the mtcars data set. Fit a model with mpg as the outcome that considers number of cylinders as a factor variable and weight as confounder. Now fit a second model with mpg as the outcome model that considers the interaction between number of cylinders (as a factor variable) and weight. Give the P-value for the likelihood ratio test comparing the two models and suggest a model using 0.05 as a type I error rate significance benchmark.

```r
intfit <- lm(mpg ~ factor(cyl) * wt, mtcars)
summary(fit)$coef
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)  33.990794  1.8877934 18.005569 6.257246e-17
## factor(cyl)6 -4.255582  1.3860728 -3.070244 4.717834e-03
## factor(cyl)8 -6.070860  1.6522878 -3.674214 9.991893e-04
## wt           -3.205613  0.7538957 -4.252065 2.130435e-04
```

```r
summary(intfit)$coef
```

```
##                   Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)      39.571196   3.193940 12.3894599 2.058359e-12
## factor(cyl)6    -11.162351   9.355346 -1.1931522 2.435843e-01
## factor(cyl)8    -15.703167   4.839464 -3.2448150 3.223216e-03
## wt               -5.647025   1.359498 -4.1537586 3.127578e-04
## factor(cyl)6:wt   2.866919   3.117330  0.9196716 3.661987e-01
## factor(cyl)8:wt   3.454587   1.627261  2.1229458 4.344037e-02
```

```r
anova(fit, intfit)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ factor(cyl) + wt
## Model 2: mpg ~ factor(cyl) * wt
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     28 183.06
```

```
## 2     26 155.89  2     27.17 2.2658 0.1239
```

4. Consider the mtcarsmtcarsmtcars data set. Fit a model with mpg as the outcome that includes number of cylinders as a factor variable and weight inlcuded in the model as:

```r
weightedfit <- lm(mpg ~ I(wt * 0.5) + factor(cyl), mtcars)
```

How is the wt coefficient interpretted?

```r
summary(weightedfit)$coef
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)  33.990794   1.887793 18.005569 6.257246e-17
## I(wt * 0.5)  -6.411227   1.507791 -4.252065 2.130435e-04
## factor(cyl)6 -4.255582   1.386073 -3.070244 4.717834e-03
## factor(cyl)8 -6.070860   1.652288 -3.674214 9.991893e-04
```

- Change in mpg per 1 ton increase in weight while holding cyl constant (Since wt is originally in 1000 lbs)

5. Given:

```r
x <- c(0.586, 0.166, -0.042, -0.614, 11.72)
y <- c(0.549, -0.026, -0.127, -0.751, 1.344)
```

Give the hat diagonal for the most influential point

```r
fit <- lm(y ~ x)
hatvalues(fit)
```

```
##         1         2         3         4         5
## 0.2286650 0.2438146 0.2525027 0.2804443 0.9945734
```

6. With the same dataset as question 5, give the slope dfbeta for the point with the highest hat value.

```r
dfbetas(fit)
```

```
##   (Intercept)             x
## 1  1.06212391   -0.37811633
## 2  0.06748037   -0.02861769
## 3 -0.01735756    0.00791512
## 4 -1.24958248    0.67253246
## 5  0.20432010 -133.82261293
```

7. Consider a regression relationship between Y and X with and without adjustment for a third variable Z. Which of the following is true about comparing the regression coefficient between Y and X with and without adjustment for Z.

- It is possible to have a scenerio of Simpson's paradox where the coefficient can be positive and strongly significant before adjusting for Z, then negative and strongly significant after the adjustment.

Logistic Regression and Poisson Regression

GLMs

Intro

- Generalized linear models (GLMs) framework came from **Nelder and Wedderburn in 1972** in the Journal of the Royal Statistical Society.

- GLM is a family of models that includes linear models.
  - Extending the family handles many of the issues with linear models, but at the expense of some complexity and loss of some of the mathematical "tidiness"

- A GLM involves three componets
  - An *exponential family* model for the response

  - A systematic component via a linear predictor

  - A link function that connects the means of the response to the linear predictor

- The three most famous cases of GLMs are:
  - Linear models

  - Binomial and binary regression

  - Poisson regression

limitations of Linear Models

- Additive response models don't make much sense if the response is discrete, or strictly positive.

- Additive error models often don't make sense, for example if the outcome has to be positive.

- Transformations are often hard to interpret
  - There's value in modeling the data on the scale that it was collected.

  - Particularly interpetable transformations, natural logarithms in specific, aren't applicable for negative or zero values.

Example, Linear Models

- Assume that $Y_i \sim N(\mu_i, \sigma^2)$ (the Gaussian distribution is an exponential family distribution.)

- Define the linear predictor to be $\eta_i = \sum_{k=1}^{p} X_{ik}\beta_k$.

- The link function, g, is defined such that $g(\mu) = \eta$.
  - For linear models $g(\mu) = \mu$ as such $\mu_i = \eta_i$

- The difference here is that we state Y_i is normally distributed as opposed to saying the error is normally distributed. We then connect the mean to the linear predictor.
  - The reasoning for this will be more clear when we look at different settings

Example, Logistic Regression

- Assume that $Y_i \sim Bernoulli(\mu_i)$ (Binomial) such that $E[Y_i] = \mu_i$ where $0 \leq \mu_i \leq 1$.

- Linear predictor $\eta_i = \sum_{k=1}^{p} X_{ik}\beta_k$

- Link function $g(\mu) = \eta = log(\frac{\mu}{1-\mu})$
  - g is the (natural) log odds, referred to as the **logit**
    - log is assumed to be base e

- The parameter estimates are obtained by maximizing the following liklihood function: $\prod_{i=1}^{n} \mu_i^{y_i}(1-\mu_i)^{1-y_i} = \exp(\sum_{i=1}^{n} y_i\eta_i)\prod_{i=1}^{n}(1+\eta_i)^{-1}$

Example, Poisson Regression

- Assume that $Y_i \sim Poisson(\mu_i)$ such that $E[Y_i] = \mu_i$ where $0 \leq \mu_i$

- Linear predictor (is still) $\eta_i = \sum_{k=1}^{p} X_{ik}\beta_k$

- Link function $g(\mu) = \eta = log(\mu)$

Some Things to Note

- In each case, the only way in which the likelihood depends on the data is through: $\sum_{i=1}^{n} y_i\eta_i = \sum_{i=1}^{n} y_i \times \sum_{k=1}^{p} X_{ik}\beta_k = \sum_{k=1}^{p} \beta_k \times \sum_{i=1}^{n} X_{ik}y_i$

- Thus if we don't need the full data, only $\sum_{i=1}^{n} X_{ik}y_i$.
  - This simplification is a consequence of chosing so-called "canonical" link functions

- All models achieve their maximum likelihood at the root of the so called normal equations $0 = \sum_{i=1}^{n} \frac{(Y_i-\mu_i)}{Var(Y_i)}W_i$
  - Where W_i are the derivative of the inverse of the link fucntion.

About Variances

- For the linear model $Var(Y_i) = \sigma^2$ is constant.

- For the Bernoulli model $Var(Y_i) = \mu_i(1-\mu_i)$

    - As such the variance depends on which observation you're looking at

- For the Poisson model $Var(Y_i) = \mu_i$
    - This variance also depends on the observation you're looking at

- There are `quasi-` versions of the non-constant functions that can be used when the data doesn't follow the variance assumtions of the GLMs.

Odds & Ends

- The normal equations ahve to be solved iteratively. Resulting in $\hat{\beta}_k$ and, if included, $\hat{\phi}$.

- Preddicted linear predictor responses can be obtained as $\hat{\eta} = \sum_{k=1}^{p} X_k \hat{\beta}_k$

- Predicted mean responses as $\hat{\mu} = g^{-1}(\hat{\eta})$

- Coefficients are interpretted as
  $g(E[Y|X_k = x_k + 1, X_{\sim k} = x_{\sim k}]) - g(E[Y|X_k = x_k, X_{\sim k} = x_{\sim k}]) = \beta_k$

- This is interpreted as the change in the link function of the expected response per unit change in X_k holding other regressors constant.
    - Variations on Newon/Raphson's algorithm are used to do it.
    - Asyptotics are used for inference usually, which means they require larger sample sizes for the P-values to be informative.
    - Many of the ideas from linear models can be brought over to GLMs

Logistic Regression

- This section coverers logistic regression for binary outcomes

Logistic Regression Part 1

Key Ideas

- Frequently we care about outcomes that ahve two values
    - Alive/Dead
    - Win/loss
    - Success/Failure
    - etc.

- Called binary, Bernoulli or 0/1 outcomes

- Collection of exchangeable binary outcomes for the same covariate data are called binomial outcomes.

Example Baltimore Ravens win/loss

```
url <- "https://github.com/bcaffo/courses/raw/master/07_RegressionModels/03_02_binaryOutcomes/d
loc <- "./data/ravensData.rda"
download.file(url, destfile = loc)
load(loc)
head(ravensData)
```

```
##   ravenWinNum ravenWin ravenScore opponentScore
## 1           1        W         24             9
## 2           1        W         38            35
## 3           1        W         28            13
## 4           1        W         34            31
## 5           1        W         44            13
## 6           0        L         23            24
```

- This data contains a 1/0 for W/L which are the first two columns.

- The other two columns are the Raven's score and their opponent's score for the game

Linear Regression

$RW_i = b_0 + b_1 RS_i + e_i$
* RW_i - 1 fi a Ravens win, 0 if not
* RS_i - Number of points Ravens scored
* b_0 - probability of a Ravens win if they score 0 points
* b_1 - increase in probability of a Ravens win for each additional point
* e_i - residual varaition

```
lmRavens <- lm(ravensData$ravenWinNum ~ ravensData$ravenScore)
res <- summary(lmRavens)$coef
res
```

```
##                         Estimate  Std. Error  t value   Pr(>|t|)
## (Intercept)           0.28503172 0.256643165 1.110615 0.28135043
## ravensData$ravenScore 0.01589917 0.009058997 1.755069 0.09625261
```

```
res[1,]
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##  0.2850317  0.2566432  1.1106149  0.2813504
```

- From this model we can acertain:
  - $b_0 = 0.285$

  - $b_1 = 0.0159$

Odds

- Binary Outcome 0/1
  RW_i

- Probability $(0,1)$
  $Pr(RW_i|RS_i, b_0, b_1) =$

  p

- Odds $(0,\infty)$
  $\frac{Pr(RW_i|RS_i, b_0, b_1)}{1-Pr(RW_i|RS_i, b_0, b_1)} =$

  $\frac{p}{1-p}$
  - Probability can be derived from the odds: $p = \frac{O}{1+O}$

- Log odds $(-\infty,\infty)$ (**logit**)
  $\log(\frac{Pr(RW_i|RS_i, b_0, b_1)}{1-Pr(RW_i|RS_i, b_0, b_1)}) =$

  $\log(\frac{p}{1-p})$

Linear vs. Logistic Regression

- Linear definitions
  - $RW_i = b_0 + b_1 RS_i + e_i$

  - $E[RW_i|RS_i, b_0, b_1] = b_0 + b_1 RS_i$

- Logistic definitions
  - $Pr(RW_i|RS_i, b_0, b_1) = \frac{exp(b_0 + b_1 RS_i)}{1+exp(b_0 + b_1 RS_i)}$

  - $\log(\frac{Pr(RW_i|RS_i, b_0, b_1)}{1-Pr(RW_i|RS_i, b_0, b_1)}) = b_0 + b_1 RS_i$

Interpreting Logistic Regression

$\log(\frac{Pr(RW_i|RS_i, b_0, b_1)}{1-Pr(RW_i|RS_i, b_0, b_1)}) = b_0 + b_1 RS_i$
* b_0 - Log odds of a Ravens win if they score 0 points
* b_1 - Log odds ratio of win probability for each point scored (compared to zero points)
* $\exp(b_1)$ - Odds ratio of win probability for each point scored (compared to zero points)

Odds

- Imagine that you are playing a game where you have a probability of success of p (Perhaps drawing cards or something).

- If you win you gain X dollars, if you lose you lose Y dollars.

- What should we set X and Y for the game to be fair?
  $E[earnings] = Xp - Y(1-p) = 0$
- Which implies...
  $Xp = Y(1-p)$
  $\frac{Xp}{1-p} = Y$
  $\frac{p}{1-p} = \frac{Y}{X}$

- The odds can be said as "How much should you be willing to pay for a p probability of winning a dollar?"
  - If $p > 0.5$ you have to pay more if you lose than you get if you win.

  - If $p < 0.5$ you have to pay less if you lose than you get if you win.

Logistic Regression Part 2

Visualizing Fitting Logistic Regression Curves

```
set.seed(1618033)
library(manipulate)
x <- seq(-10, 10, length = 1000)
## manipulate can only be executed in an R Studio session
manipulate(
  plot(x, exp(beta0 + beta1 * x) / (1 + exp(beta0 + beta1 * x)),
       type = "l", lwd = 3, frame = FALSE),
  beta1 = slider(-2, 2, step = 0.1, initial = 2),
  beta0 = slider(-2, 2, step = 0.1, initial = 0)
  )
```

- This is a unit step function with a large enough `beta1`
  - `beta1` adjusts the steepness and direction of the slope

  - `beta0` determines at what x value the function changes from 0 to 1 (or 1 to 0 if `beta1` is negative)

- This model is: $\frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}}$

Logistic Regression Part 3

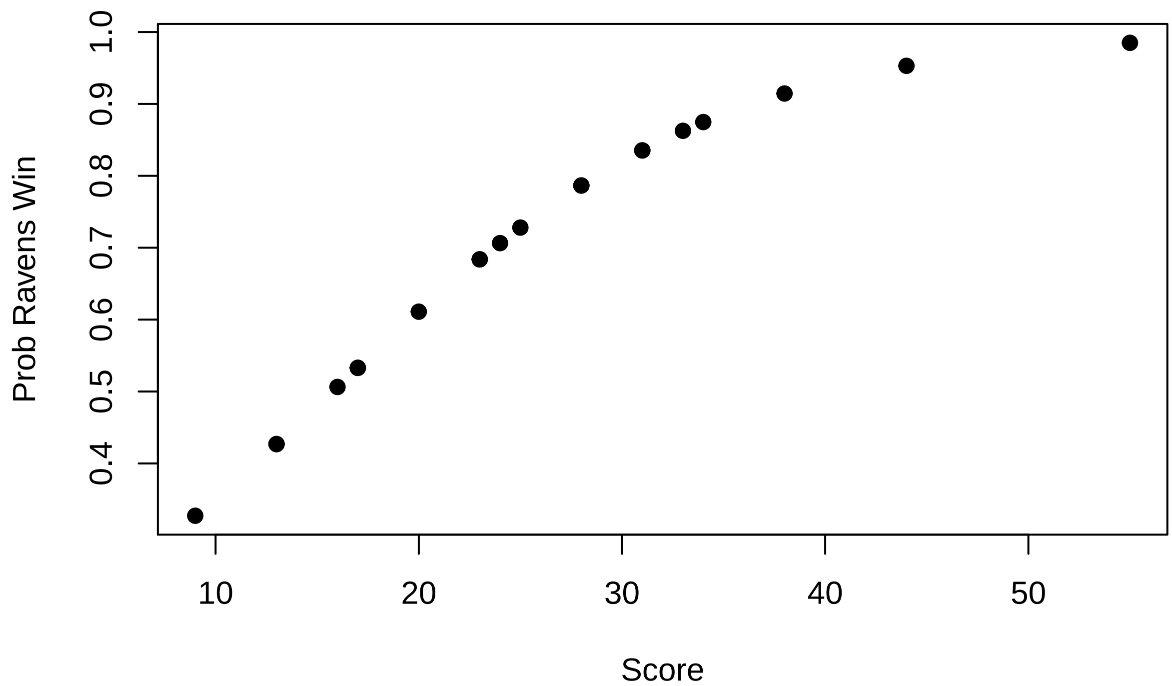
Ravens Logistic Regression

```
#If we had bounded count data we'd have to give a sample size
# Assumes you want the logistic link function, which we do in this case
logRegRavens <- glm(ravensData$ravenWinNum ~ ravensData$ravenScore,
                    family = "binomial")
summary(logRegRavens)
```

```
##
## Call:
## glm(formula = ravensData$ravenWinNum ~ ravensData$ravenScore,
##     family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7575  -1.0999   0.5305   0.8060   1.4947
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -1.68001    1.55412  -1.081     0.28
## ravensData$ravenScore  0.10658    0.06674   1.597     0.11
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 24.435  on 19  degrees of freedom
## Residual deviance: 20.895  on 18  degrees of freedom
## AIC: 24.895
##
## Number of Fisher Scoring iterations: 5
```

- Fitted Values

```r
plot(ravensData$ravenScore, logRegRavens$fitted,
     pch = 19, xlab = "Score", ylab = "Prob Ravens Win")
```



- These fitted values are only showing part of the curve since no data existed with a low enough score (i.e. 0)

Odds Ratios & Confidence Intervals

```
#Suggests a [1,2] increase in p(winning) for each point scored
exp(logRegRavens$coeff)
```

```
##         (Intercept) ravensData$ravenScore
##           0.1863724             1.1124694
```

```
#Looking on expenetial scale
exp(confint(logRegRavens))
```

```
## Waiting for profiling to be done...
```

```
##                              2.5 %   97.5 %
## (Intercept)            0.005674966 3.106384
## ravensData$ravenScore 0.996229662 1.303304
```

ANOVA for Logistic Regression

```
#Additional models could be added, or factor variables could be considered
anova(logRegRavens, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: ravensData$ravenWinNum
##
## Terms added sequentially (first to last)
##
##
##                      Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                                    19     24.435
## ravensData$ravenScore  1   3.5398        18     20.895  0.05991 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpreting Odds Ratios

- Not probabilities
- Odds ratio of 1 = no difference in odds
- Log odds ratio of 0 = no difference in odds
- Odds ratio < 0.5 or > 2 is commonly refered to as a "moderate effect"
  - Depends ont he context
- Relative risk, $\frac{Pr(RW_i|RS_i=10)}{Pr(RW_i|RS_i=0)}$, is often easier to interpret, but harder to estimate. It is just the ratio of two probabilities.
- For small probabilities Relative Risk $\approx$ Odds Ratio but **they are not the same!**

Further Resources

- **Wikipedia on Logistic Regression**
- **Logistic regression and glms in R**
  PDFs:
- Brian Caffo's lecture notes on: **Simpson's paradox**, **Case-control studies**
- **Open Intro Chapter on Logistic Regression**

Lesson with `swirl()`: Variance Inflation Factors

- Variance Inflation is when you include any new variables to a model, it will increase the standard errors of the other regressors

- This lesson looks at `vifSims.R`, which is has some simulations of this in effect
  - `rgp1()` has constant variance since each variable is uncorrelated

  - `rgp2()` has diffrent variances since `x2` & `x3` are correlated with `x1` * This lesson also looks at the `swiss` data set again, which was previously covered.

Lesson with `swirl()`: Overfitting and Underfitting

- This lesson has largely already been covered, however it does use `fitting.R` to simulate fitting diffrent models

- The `shapiro.test()` function will test a parameter for normality, with a null-hypothesis that it is normally distributed
  - This can be used to test residuals, if they aren't normally distributed it suggest there may be an underlying pattern

Reminder to commit (13) delete this line *AFTER* committing

Poisson Regression

Poisson Regression Part 1

Poisson Regression Part 2

Lesson with `swirl()`: Binary Outcomes

Lesson with `swirl()`: Count Outcomes

Reminder to commit (14) delete this line *AFTER* committing

Hodgepodge

Mishmash

Hodgepodge

Reminder to commit (15) delete this line *AFTER* committing

Quiz 4

Reminder to commit (S4) delete this line *AFTER* committing

Course Project

Reminder to commit (P1) delete this line *BEFORE* committing