Lab 10. Linear Regression

# Introduction

Linear regression is the next step up from correlation! In this lab you will learn how to visualize and run linear regressions using fake data, and then you will get to analyze my favorite weird dataset that I have ever found: the cow fight club data set.



# Learning Outcomes

By the end of today’s class you should be able to do the following in R:

* Use the lm() and summary() functions to perform a linear regression
* Use the melt() and facet\_grid() functions to examine multiple data sets at once
* Use the geom\_smooth() function to create visualizations of trends
* Interpret linear regression output

# Part 0: Libraries

Make sure to load the ggplot2 and reshape2 packages before you get started.

# Part 1: Fake Data

## Part 1.1 Make the Fake Data

Just like in week 9, we need data to play around with that has a relationship. So in this case, we’re going to make an x variable, then a few different y variables:

* One that has a positive relationship with x (**y1**)
* One that has a negative relationship with x (**y2**
* One that has no relationship with x (**y3**)
* One that has a complicated relationship with x (**y4**)

x <- rnorm(n = 50, mean = 25, sd = 5)  
y1 <- jitter(x, factor = 2000)  
y2 <- -1\*jitter(x, factor = 2000)  
y3 <- rnorm(n = 50, mean = 25, sd = 5)  
y4 <- jitter(x/2, factor = 2000) + 22

Now, let’s put them all together into a single data frame, which since I keep describing them and their relationship to x feels a little like a reality TV show featuring a lot of emotionally damaged contestants shoved into a space, so we will officially be calling this data set after a certain trashy tv show that I will never, ever publicly admit to watching.

AYTO <- data.frame(x,y1,y2,y3,y4)

## Part 1.2 Visualize

We’re going to use ggplot2 to visualize this data, but that means we need to have the data set in a longer format first using melt() from ggplot2(). Unlike in previous labs, if you don’t specify which column is which, the melt() function gets very confused, so make sure to use the *id.vars* argument.

AYT1 <- melt(AYTO, id.vars="x")

Now we can visualize this using ggplot - and this is where ggplot really begins to shine, because we can do a lot of different types of visualizations on different variables very quickly. To start with, here is a basic scatterplot with color-coded dots.

ggplot(AYT1, aes(x = x, y = value, col = variable)) + geom\_point()

However, you can do a lot more than just this. Use the following modifying code chunks, and add them to your ggplot code to see what they do, then answer the questions below.

geom\_smooth()  
geom\_smooth(method="lm")  
geom\_smooth(se=FALSE)  
facet\_grid(~variable)

### QUESTION 1: Which chunk of code adds a linear regression model with a ribbon of uncertainty?

A. geom\_smooth()  
B. geom\_smooth(method="lm")  
C. geom\_smooth(se=FALSE)  
D. facet\_grid(~variable)

### QUESTION 2: Looking at your data, what is the difference between y1 and y4?

1. Different intercept
2. One has a negative slope
3. One has no relationship with x
4. There is no real difference

## Part 1.3 Linear Regression Models

Now that we have looked at this fake data and made some hypotheses, we can proceed to building our linear regressions. We will use two different functions for this, the lm() function to build the model and the summary() function to interpret it.

The lm() function is very simple. It returns two variables, the **intercept** and the **slope**, which fit well into the formula y = mx + b. In the lm() function, you put the y variable first, then a tilde (~) and then the x variable, like so:

lm(y1~x)

##   
## Call:  
## lm(formula = y1 ~ x)  
##   
## Coefficients:  
## (Intercept) x   
## -0.127 1.013

In this case, the intercept is -0.127 and the slope is 1.013 - for some reason, R likes to label the slope as *x*, though it is more traditionally labeled as *b*.

### QUESTION 3: For both y1 and y2, your slope (x) should be close to 1 or -1. Why?

1. Because the average of the independent variable is 1
2. Because the intercept is 1
3. Because we just jittered x, and didn’t multiple it by anything that increased the slope
4. Because the maximum slope is always 1

### QUESTION 4: For y3, your slope should be close to 0. Why?

1. Because as x increases, y3 does nothing
2. Because x and y3 are very strongly correlated
3. Because y3 crosses the y axis at 0

### QUESTION 5: Which chunk of code would get you a linear regression of y4 versus x?

A. lm(AYT1$value~AYT1$x)  
B. lm(AYTO$value~AYTO$x)  
C. lm(AYTO$y4~AYTO$x)  
D. lm(AYT1$y4~AYT1$x)

## Part 1.4 Linear Model Significance

Though the lm() function builds a linear model, it doesn’t interpret how good of a fit the linear model is, nor does it tell you if the fit is statistically significant. To do that, you need to use the summary() function. You can run this as a nested function, like so:

summary(lm(y1~x))

##   
## Call:  
## lm(formula = y1 ~ x)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7.5807 -3.7611 0.1693 3.2781 7.0250   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.1270 3.1976 -0.040 0.968   
## x 1.0130 0.1215 8.336 6.87e-11 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.357 on 48 degrees of freedom  
## Multiple R-squared: 0.5915, Adjusted R-squared: 0.5829   
## F-statistic: 69.49 on 1 and 48 DF, p-value: 6.87e-11

There’s a lot of information here, but some bits are the same. You can see that the **intercept** and **slope** are repeated here, under a table where their heading is referred to as **estimate**. And if you scan over to the right, you’ll see a column that says **Pr(>|t|)** - this is your *p* value for each part of your equation!

### QUESTION 6: The intercept here is not statistically significant (p = 0.968). What is it not significantly different from? (go back to the lecture if you need help!)

1. The slope
2. The average x value
3. An intercept of 0
4. An intercept of 1

### QUESTION 7: The slope in this equation is statistically significant (p <0.001). What is it significantly different from? (go back to the lecture if you need help!)

1. The intercept
2. The average x value
3. A slope of 0
4. A slope of 1

### QUESTION 8: Find the significance of a linear regression between x and y4. Are the slope and intercept statistically significant? Show your code, and use your *p* values for each to explain your answer.

## Part 1.5 Linear Model Fit

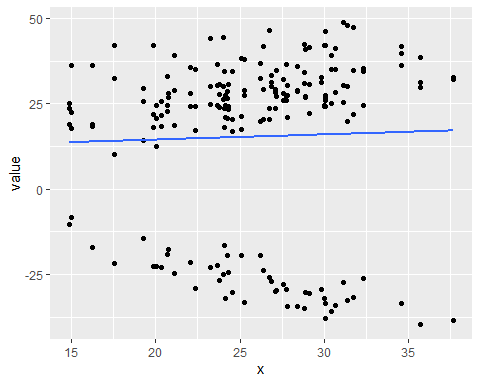
Statistical significance is important, but how good of a fit is the data in the first place? For that, you need to look at the R2 value. This is also output by the summary() function - but there are two! For now, just use the Multiple R-squared.

In the table in part 1.4, the R2 value is 0.591.

### QUESTION 9: Look at the models for y2, y3, and y4. Which one has the worst R2 value?

1. y2, because it is a negative relationship
2. y3, because it has little to no relationship
3. y4, because it has a different intercept

### QUESTION 10: Run a model where all y values are combined, the equivalent of the following graph. Why does this have such a low R2 value?

*do not just say because they have no relationship - why isn’t there one?* 

# Part 2: Cow Fight Club

Today you are going to be analyzing my favorite dataset I have ever found: the Cow Fight Club Data set.

OK, that’s not what it’s called in real life - the paper was Sartori, Cristina, et al. “Evolution of increased competitiveness in cows trades off with reduced milk yield, fertility and more masculine morphology.” Evolution 69.8 (2015): 2235-2245.

It’s a paper where the authors tried to determine if there was an evolutionary tradeoff between fighting prowess in cows and their other features (i.e. milk yield). So, essentially… cow fight club.

This dataset is full of continuous variables:

* **Animal** - a unique identifier for each cow, which is lame because they could have named each cow, missed opportunity
* **Fertility** - a continuous measurement of fertility, derived from the number of calves had each year, and the speed with which the cow became pregnant
* **Fighting\_ability** - how badass the cow is, scale from not to epic
* **Milk** - how much milk the cow produced, no idea what units let’s say liters
* **Fat** - how much fat was in the cow’s milk, let’s say grams
* **Protein** - how much protein was in the cow’s milk, let’s say milligrams
* **Weight\_category\_WC** - a categorical data set of weight class
* **Body\_weight** - the approximate weight of the cow
* **Calves** - the number of calves had by the cow
* **Age\_class** - approximate age in years
* **FightingLeague** - a categorical variable derived from Fighting Ability

Your job will be to formulate a hypothesis as to which of these variables best predicts fighting ability. You can pick any x variable other than **Fighting\_ability** or **Animal** (it’s basically a name, not a predictor). You can run all of them if you want, but you don’t have to find the best predictor - you just have to make a guess, test your guess, and explain why you were right or wrong.

### QUESTION 11: Which x variable do you plan on using? Explain why you think this one is (or isn’t) a good variable.

### QUESTION 12: Use an if-then style to write out a testable null hypothesis with your variables. Specify the p value of your slope.

### QUESTION 13: Is your variable a good predictor of cow fighting ability? Defend your answer using your *p* values and your R2 values.

### QUESTION 14: Are you rejecting your null hypothesis or not? Explain your answer.