# Regression

# Recap

- Correlations are their own effect size
- On a scale of -1 to 1
- Useful for depicting relationships

# **Today**

#### Regression

- What is it? Why is it useful
- Nuts and bolts
  - Equation
  - Ordinary least squares
  - Interpretation

## Regression

- Regression is an umbrella term -- lots of things fall under "regression"
- This system can handle a variety of forms of relations, although all forms have to be specified in a *linear* way.

The output of regression includes both effect sizes and statistical significance. We can also incorporate multiple influences (IVs) and account for their intercorrelations.



### Regression

- **Scientific** use: explaining the influence of one or more variables on some outcome.
  - Does this intervention affect reaction time?
  - Does self-esteem predict relationship quality?
- **Prediction** use: We can develop models based on what's happened in the past to predict what will happen in the figure.
  - Insurance premiums
  - Graduate school... success?
- Adjustment: Statistically control for known effects
  - If everyone had the same level of SES, would abuse still be associated with criminal behavior?

## How does Y vary with X?

- The regression of Y (DV) on X (IV) corresponds to the line that gives the mean value of Y corresponding to each possible value of X
- "Our best guess" regardless of whether our model includes categories or continuous predictor variables

## **Regression Equation**

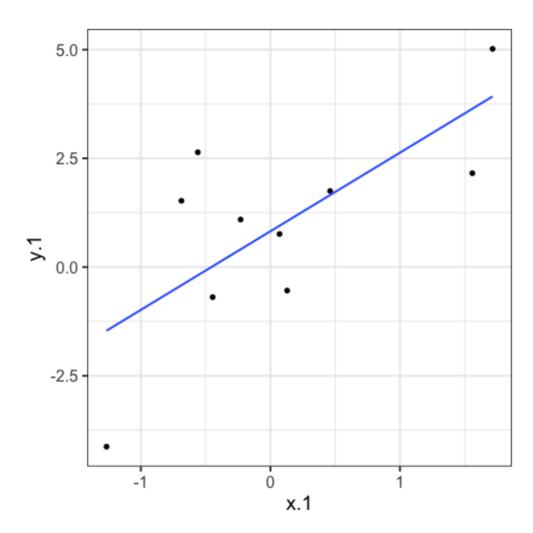
$$Y = b_0 + b_1 X + e$$
  $\hat{Y} = b_0 + b_1 X$ 

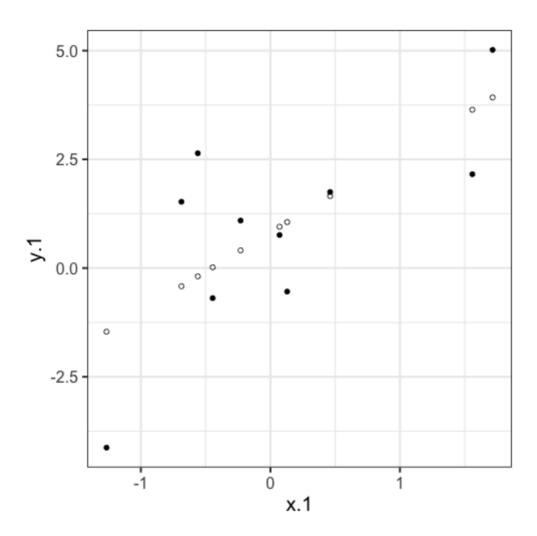
## **OLS**

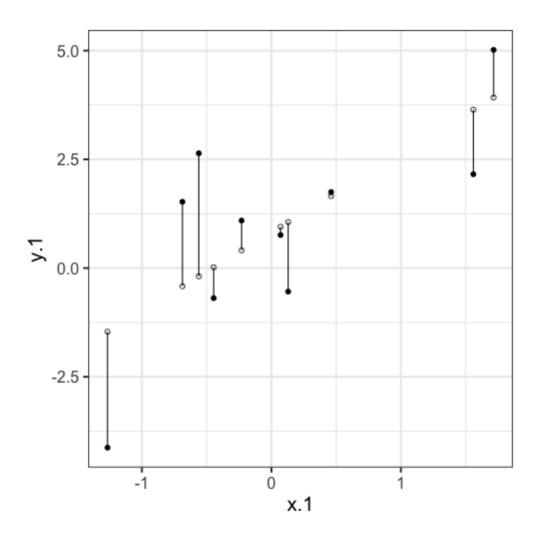
- How do we find the regression estimates?
- Ordinary Least Squares (OLS) estimation
- Minimizes deviations

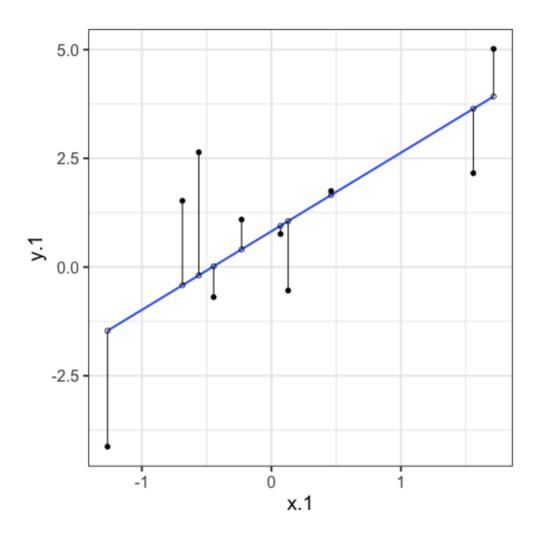
$$min\sum (Y_i - \hat{Y})^2$$

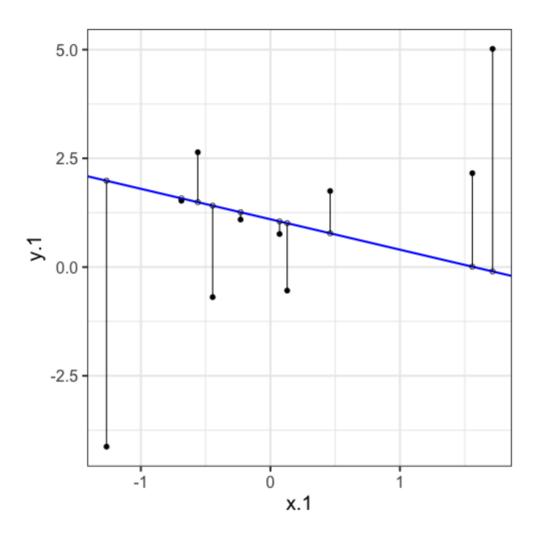
• Other estimation procedures possible (and necessary in some cases)











# compare to bad fit

$$Y=b_0+b_1X+e$$
  $\hat{Y}=b_0+b_1X$   $Y_i=\hat{Y}_i+e_i$   $e_i=Y_i-\hat{Y}_i$ 

### **OLS**

The line that yields the smallest sum of squared deviations

$$egin{aligned} \Sigma(Y_i - \hat{Y}_i)^2 \ &= \Sigma(Y_i - (b_0 + b_1 X_i))^2 \ &= \Sigma(e_i)^2 \end{aligned}$$

In order to find the OLS solution, you could try many different coefficients  $(b_0 \text{ and } b_1)$  until you find the one with the smallest sum squared deviation. Luckily, there are simple calculations that will yield the OLS solution every time.

## In R

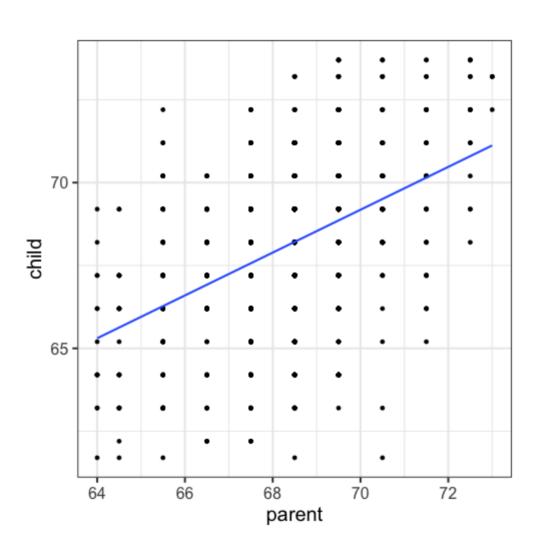
#### What if we regress parent height onto child height?

```
fit.1 <- lm(child ~ parent, data = galton.data)
 summary(fit.1)
##
## Call:
## lm(formula = child ~ parent, data = galton.data)
## Residuals:
      Min
               10 Median
                              3Q
## -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 '**' 0.05 '.' 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
```

A 1-unit change in X predicts a *b* change in Y

A 1-standard deviation change in X predicts a *b* standard deviation change in Y

## Child Height Predicted By Parent Height



## Data, predicted, and residuals

```
library(broom)
model info = augment(fit.1)
head(model_info)
## # A tibble: 6 x 8
##
    child parent .fitted .resid .std.resid .hat .sigma .cooksd
    <dbl>
          <dbl>
                 <dbl> <dbl>
                                <dbl>
                                       <dbl>
                                              <dbl>
                                                     <dbl>
##
## 1
    61.7 70.5 69.5 -7.81 -3.49 0.00270 2.22 0.0165
## 2 61.7 68.5 68.2 -6.51 -2.91 0.00109 2.23 0.00462
## 3 61.7 65.5 66.3 -4.57 -2.05 0.00374 2.23 0.00787
## 4 61.7 64.5 65.6 -3.93 -1.76 0.00597 2.24 0.00931
## 5 61.7 64 65.3 -3.60
                                -1.62 0.00735 2.24 0.00966
## 6 62.2
          67.5 67.6 -5.37
                                -2.40 0.00130 2.23 0.00374
describe(model_info)
```

```
##
                              sd median trimmed
                                                                       skew
                                                mad
                                                      min
                                                            max range
                       mean
             vars
##
  child
                                         68.12 2.97 61.70 73.70 12.00 -0.09
                1 928 68.09 2.52
                                  68.20
## parent
                2 928 68.31 1.79
                                 68.50 68.32 1.48 64.00 73.00 9.00 -0.04
##
  .fitted
                3 928 68.09 1.16
                                 68.21
                                         68.10 0.96 65.30 71.12 5.82 -0.04
  .resid
                                 0.05
                                       0.06 2.26 -7.81 5.93 13.73 -0.24
##
                4 928
                      0.00 2.24
  .std.resid
##
                5 928
                       0.00 1.00
                                 0.02
                                          0.03 \ 1.01 \ -3.49
                                                           2.65
```

## Are we doing a good job?

- The way the world is = our model + error
- How good is our model? Does it "fit" the data well?

To assess how well our model fits the data, we simply take all the variability in our outcome and partition it into different categories. For now, we will partition it into two categories: the variability that is predicted by (explained by) our model, and variability that is not.

To the extent that we can generate different predicted values of Y *based on the different values of X*, we are doing well with our model.

## $R^2$

- $R^2$  is the amount of variance in Y that is explained by X (aka by your model)
- measure of model fit; more variance explained, better your model

## $R^2$

```
fit.1 = lm(child ~ parent, data = galton.data)
summary(fit.1)
##
## Call:
## lm(formula = child ~ parent, data = galton.data)
##
## Residuals:
      Min
          10 Median 30
##
                                    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 '**' 0.05 '.' 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
```

## **Residual Standard Error**

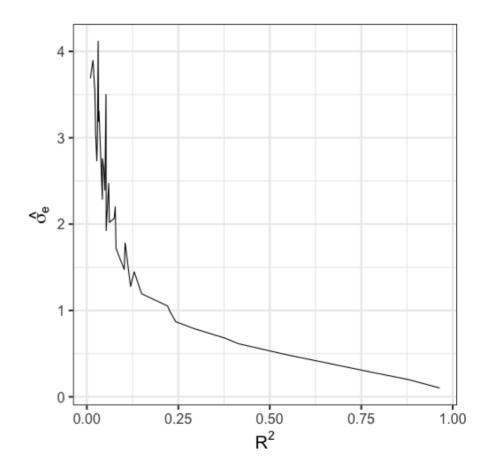
- $Residuals = Y \hat{Y}$
- There is a residual for each individual (each person has an observed score and a predicted score)
- You can plot these residuals. The *dispersion* or spread of these residuals is called the **Residual Standard Error (RSE)**
- The RSE is the standard deviation of all of these residuals (in original units); it is the standard deviation of Y that is **not** accounted for by the model
- If it's a fat distribution, that means the residuals are large; we're not doing great
- If it's a skinny distribution, then the residuals are smaller; we're doing a good job!

### **RSE**

```
fit.1 = lm(child ~ parent, data = galton.data)
summary(fit.1)
##
## Call:
## lm(formula = child ~ parent, data = galton.data)
##
## Residuals:
      Min
          10 Median 30
##
                                    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 '**' 0.05 '.' 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
```

## ${\mathbb R}^2$ and residual standard deviation

- two sides of same coin
- one in original units, the other standardized



## **Inferential Tests**

#### **Omnibus test**

• Test of whether the model fits the data

#### **Regression Coefficients**

• Is our predictor significant?

### **Omnibus test**

Overall, our goal is to partition variance. We want to know if the variance explained by our model is larger than the variance that is left over or *unexplained*.

Our sampling distribution will be the F distribution. The z and the t test for differences in means. F distribution looks at the size of a **ratio of variances**. The ratio of explained to unexplained variance. The ratio of your regression to error.

Yes, this is analogous to ANOVA. But ANOVAs require categorical predictors. Regression is more flexible!

# ANOVA is a special case of regression!

# ANOVA is a special case of regression!

```
summary(fit.1)
##
## Call:
## lm(formula = child ~ parent, data = galton.data)
##
## Residuals:
      Min 10 Median 30
##
                                    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 '**' 0.05 '.' 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
```

## **Predictors**

$$H_0: \beta_1=0$$

$$H_1:eta_1
eq 0$$

## **Predictors**

- Does X provide any predictive information?
- Does X provide any explanatory power regarding the variability of Y?
- Is the the average value the best guess (i.e., is Y bar equal to the predicted value of Y?)
- Is the regression line flat?
- Are X and Y correlated?

## **Predictors**

- One-sample *t*-tests
- $t = \frac{b}{se}$

```
summary(fit.1)
```

```
## Call:
## lm(formula = child ~ parent, data = galton.data)
##
## Residuals:
             10 Median
      Min
                             30
## -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
```

 You either need to store the lm model as it's own object, and then call summary() on it. OR, you can nest lm() within the summary() function like: summary(lm(child ~ parent, data = galton.data))

```
# summary(fit.1)
summary(lm(child ~ parent, data = galton.data))
```

```
##
## Call:
## lm(formula = child ~ parent, data = galton.data)
##
## Residuals:
      Min
               10 Median
                              30
## -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 ***
            0.64629
                       0.04114 15.711 <2e-16 ***
## parent
## ---
## 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
```

• The broom package is part of the tidyverse, but it does not load automatically; you'll need to load it separately

tidy() creates a data.frame from the output table

```
tidy(fit.1)
```

• The broom package is part of the tidyverse, but it does not load automatically; you'll need to load it separately

augment adds columns to a dataset, including things like fitted values and residuals. If it has a . in front of the column name, it was added. Also stored as a data.frame so we can use them later.

```
augment(fit.1)
```

```
## # A tibble: 928 x 8
##
     child parent .fitted .resid .std.resid .hat .sigma .cooksd
     <dbl>
           <dbl>
                  <dbl> <dbl>
                                          <dbl>
                                                <dbl>
                                   <dbl>
                                                        <dbl>
##
   1 61.7
          70.5 69.5 -7.81
                                   -3.49 0.00270 2.22 0.0165
##
##
   2 61.7 68.5 68.2 -6.51
                                   -2.91 0.00109 2.23 0.00462
##
   3 61.7 65.5 66.3 -4.57
                                   -2.05 0.00374
                                                 2.23 0.00787
           64.5
                   65.6 -3.93
##
   4 61.7
                                   -1.76 0.00597
                                                 2.24 0.00931
   5 61.7
                   65.3 -3.60
                                   -1.62 0.00735
                                                 2.24 0.00966
##
          64
                   67.6 - 5.37
##
   6 62.2
            67.5
                                   -2.40 0.00130
                                                 2.23 0.00374
##
  7 62.2
          67.5
                   67.6 - 5.37
                                   -2.40 0.00130
                                                 2.23 0.00374
                   67.6 -5.37
##
   8 62.2
            67.5
                                   -2.40 \ 0.00130
                                                 2.23 0.00374
##
                   66.9 -4.72
   9 62.2
            66.5
                                   -2.11 0.00218
                                                 2.23 0.00487
## 10
      62.2
            66.5
                   66.9 - 4.72
                                   -2.11 0.00218
                                                 2.23 0.00487
```

augment adds columns to a dataset, including things like fitted values and residuals. If it has a . in front of the column name, it was added. Also stored as a data.frame so we can use them later.

• The broom package is part of the tidyverse, but it does not load automatically; you'll need to load it separately

glance gives you the F-test & fit measures

## Adding more predictors

- You can enter lots of variables into your regression; you aren't limited to just 1
- $Y = b_0 + b_1 X_1 + b_2 X_2 + \ldots + b_n X_n + e$
- Interpretation of b coefficients changes slightly: "a 1-unit change in  $X_1$  predicts a \_ change in Y, while controlling for  $X_2$
- Interactions get even more tricky (beyond our scope, sadly)

# It's. All. Regression.

Really want to up your stats games? Go through this site so you can see how it's all just actually regression.

#### A cheatsheet:

-	Common name	Built-in function in R	Equivalent linear model in R	Exact?	The linear model in words	Icon
,	y is independent of x P: One-sample t-test N: Wilcoxon signed-rank	t.test(y) wilcox.test(y)	Im(y ~ 1) Im(signed_rank(y) ~ 1)	√ for N >14	One number (intercept, i.e., the mean) predicts <b>y</b> .  - (Same, but it predicts the <i>signed rank</i> of <b>y</b> .)	;; ;
	P: Paired-sample t-test N: Wilcoxon matched pairs	t.test(y <sub>1</sub> , y <sub>2</sub> , paired=TRUE) wilcox.test(y <sub>1</sub> , y <sub>2</sub> , paired=TRUE)	$\begin{aligned} & \text{Im}(y_2 - y_1 \sim 1) \\ & \text{Im}(\text{signed\_rank}(y_2 - y_1) \sim 1) \end{aligned}$	√ f <u>or N &gt;14</u>	One intercept predicts the pairwise y <sub>2</sub> -y <sub>1</sub> differences.  - (Same, but it predicts the <i>signed rank</i> of y <sub>2</sub> -y <sub>1</sub> .)	Z:
	y ~ continuous x P: Pearson correlation N: Spearman correlation	cor.test(x, y, method='Pearson') cor.test(x, y, method='Spearman')	Im(y ~ 1 + x) Im(rank(y) ~ 1 + rank(x))	√ for N >10	One intercept plus <b>x</b> multiplied by a number (slope) predicts <b>y</b> .  - (Same, but with <i>ranked</i> <b>x</b> and <b>y</b> )	لببسر
	y ~ discrete x P: Two-sample t-test P: Welch's t-test N: Mann-Whitney U	t.test(y <sub>1</sub> , y <sub>2</sub> , var.equal=TRUE) t.test(y <sub>1</sub> , y <sub>2</sub> , var.equal=FALSE) wilcox.test(y <sub>1</sub> , y <sub>2</sub> )	$Im(y \sim 1 + G_2)^A$ $gls(y \sim 1 + G_2, weights=^B)^A$ $Im(signed_rank(y) \sim 1 + G_2)^A$	√ √ for N >11	An intercept for <b>group 1</b> (plus a difference if <b>group 2</b> ) predicts <b>y</b> .  - (Same, but with one variance <i>per group</i> instead of one common.)  - (Same, but it predicts the <i>signed rank</i> of <b>y</b> .)	Y
- 11	P: One-way ANOVA N: Kruskal-Wallis	aov(y ~ group) kruskal.test(y ~ group)	$\begin{aligned} & \text{Im}(y\sim 1+G_2+G_3++G_N)^A \\ & \text{Im}(\text{rank}(y)\sim 1+G_2+G_3++G_N)^A \end{aligned}$	√ for N >11	An intercept for <b>group 1</b> (plus a difference if group ≠ 1) predicts <b>y</b> .  - (Same, but it predicts the <i>rank</i> of <b>y</b> .)	Ķ
	P: One-way ANCOVA	aov(y ~ group + x)	Im(y ~ 1 + $G_2$ + $G_3$ ++ $G_N$ + x) <sup>A</sup>	~	- (Same, but plus a slope on x.) Note: this is discrete AND continuous. ANCOVAs are ANOVAs with a continuous x.	<del>i</del>
-	P: Two-way ANOVA	aov(y ~ group * sex)	$\begin{split} &\text{Im}(y \sim 1 + G_2 + G_3 + + G_N + \\ &S_2 + S_3 + + S_K + \\ &G_2^* S_2 + G_3^* S_3 + + G_N^* S_K) \end{split}$	1	Interaction term: changing sex changes the $\mathbf{y} \sim \mathbf{group}$ parameters.  Note: $G_{n:w}$ is an $\underline{indicator(0.or.1)}$ for each non-intercept levels of the $\underline{\mathbf{group}}$ variable.  Similarly for $S_{1:ow}$ for sex. The first line (with $G_i$ is main effect of group, the second (with $G_i$ ) for sex and the third is the $\underline{\mathbf{group}} \times \mathbf{sex}$ interaction. For two levels (e.g. male/female), line 2 would just be "S," and line 3 would be S; multiplied with each $G_i$ .	[Com
- 11	Counts ~ discrete x N: Chi-square test	chisq.test(groupXsex_table)	Equivalent log-linear model glm(y ~ 1 + $G_2$ + $G_3$ + + $G_N$ + $S_2$ + $S_3$ + + $S_N$ + $S_2$ + $S_3$ + + $S_N$ + $S$	~	Interaction term: (Same as Two-way ANOVA.)  Note: Run gim using the following arguments: $glm (model, family=poisson())$ As linear-model, the Chi-square test is $log(y) = log(N) + log(a) + log(b) + log(a\beta)$ where $a_i$ and $\beta_i$ are proportions. See more info in the accompanying notebook.	Same Two-I ANC
	N: Goodness of fit	chisq.test(y)	glm(v ~ 1 + G <sub>2</sub> + G <sub>3</sub> ++ G <sub>N</sub> , family=) <sup>A</sup>	1	(Same as One-way ANOVA and see Chi-Square note.)	1W-AI

## **Next Time**

Principles behind data visualizations

Get ready for some rants...