2.5 — OLS: Precision and Diagnostics

ECON 480 • Econometrics • Fall 2020

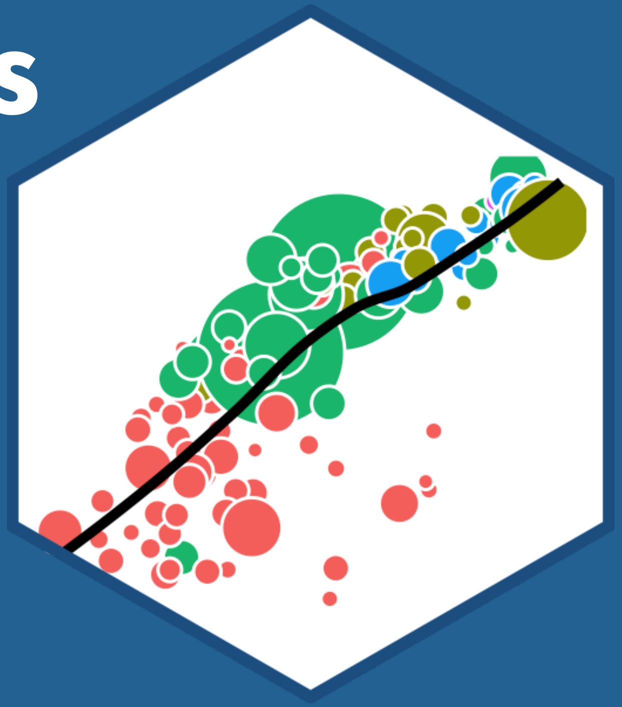
Ryan Safner

Assistant Professor of Economics

safner@hood.edu

[ryansafner/metricsF20](https://github.com/ryansafner/metricsF20)

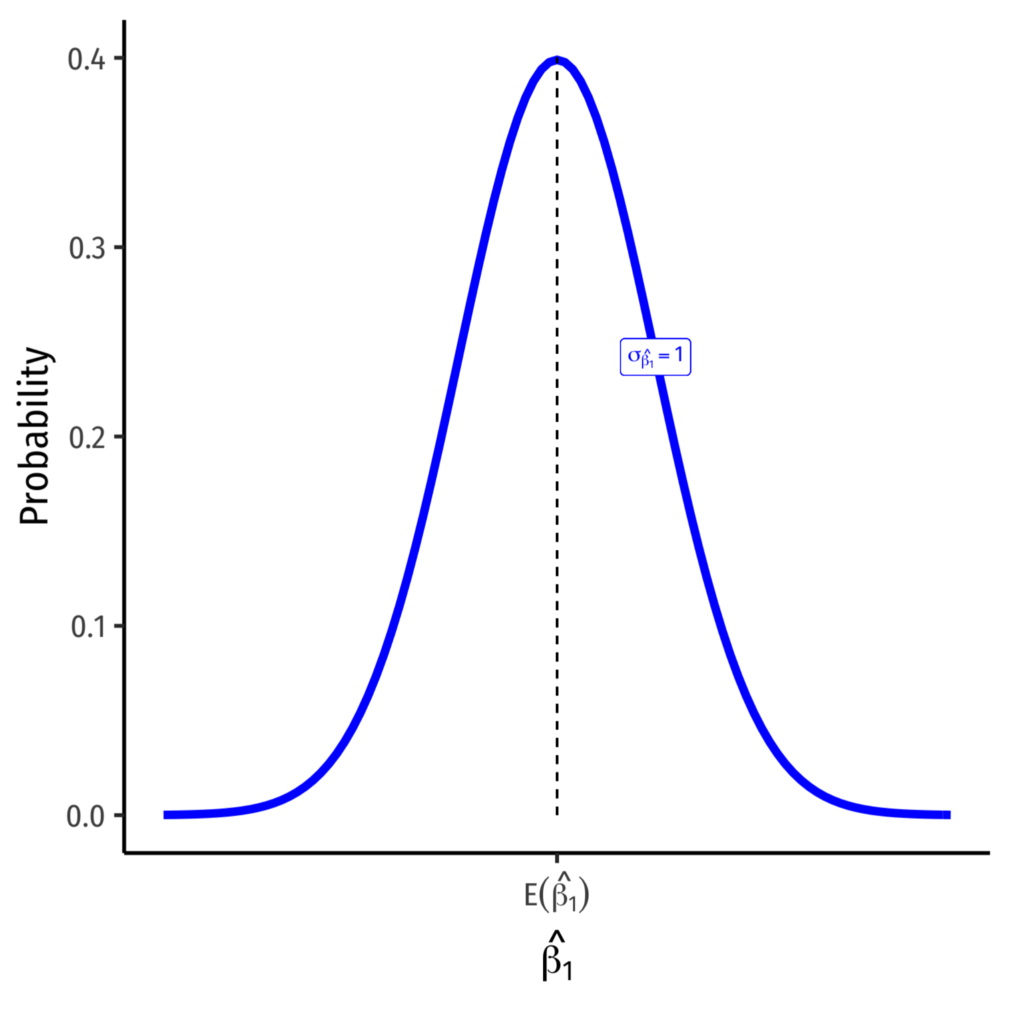
metricsF20.classes.ryansafner.com



The Sampling Distribution of $\hat{\beta}_1$

$$\hat{\beta}_1 \sim N(E[\hat{\beta}_1], \sigma_{\hat{\beta}_1})$$

1. **Center** of the distribution (last class)
   - $E[\hat{\beta}_1] = \beta_1$ †



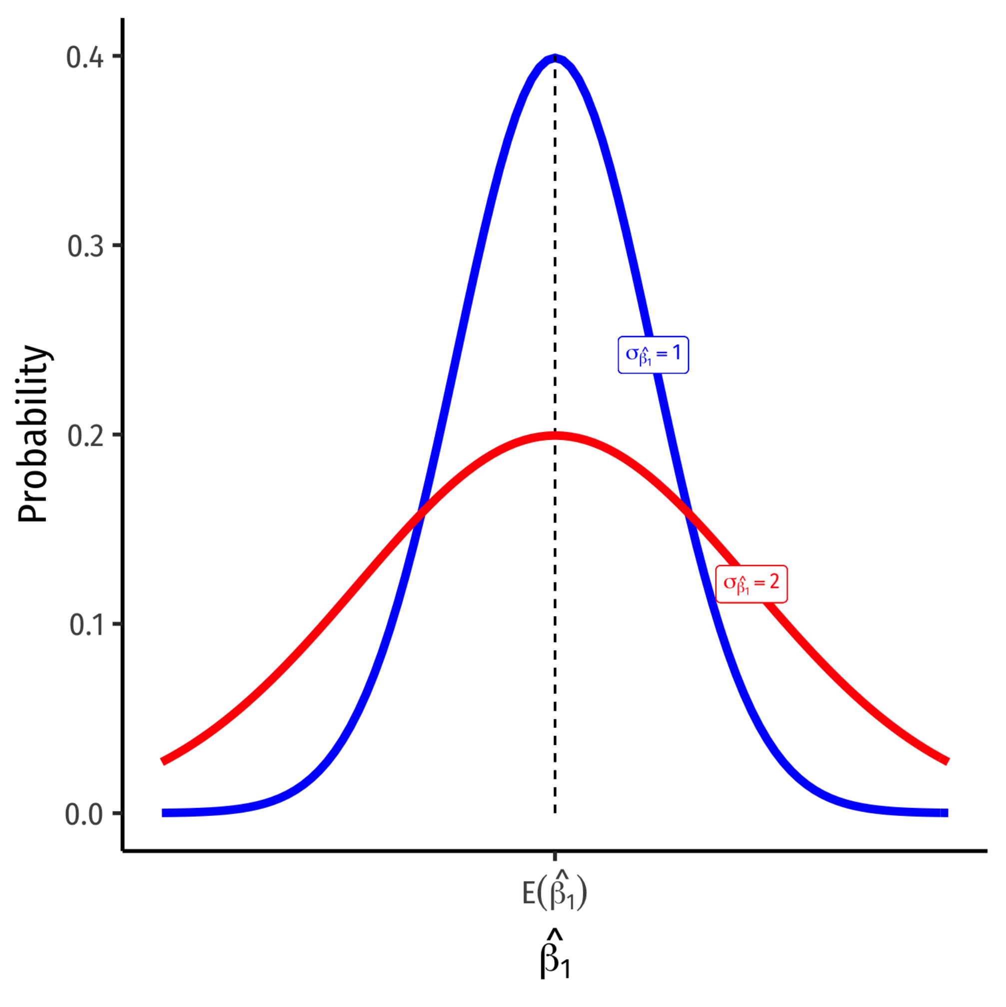
The Sampling Distribution of $\hat{\beta}_1$

$$\hat{\beta}_1 \sim N(E[\hat{\beta}_1], \sigma_{\hat{\beta}_1})$$

1. **Center** of the distribution (last class)
    - $E[\hat{\beta}_1] = \beta_1$ †
2. How **precise** is our estimate? (today)
    - **Variance** $\sigma^2_{\hat{\beta}_1}$ or **standard error**‡ $\sigma_{\hat{\beta}_1}$

† Under the 4 assumptions about u (particularly, $cor(X, u) = 0$).

‡ Standard **"error"** is the analog of standard *deviation* when talking about the *sampling distribution* of a sample statistic (such as $\bar{X}$ or $\hat{\beta}_1$).



Variation in $\hat{\beta}_1$

What Affects Variation in $\hat{\beta}_1$

$$var(\hat{\beta}_1) = \frac{(SER)^2}{n \times var(X)}$$

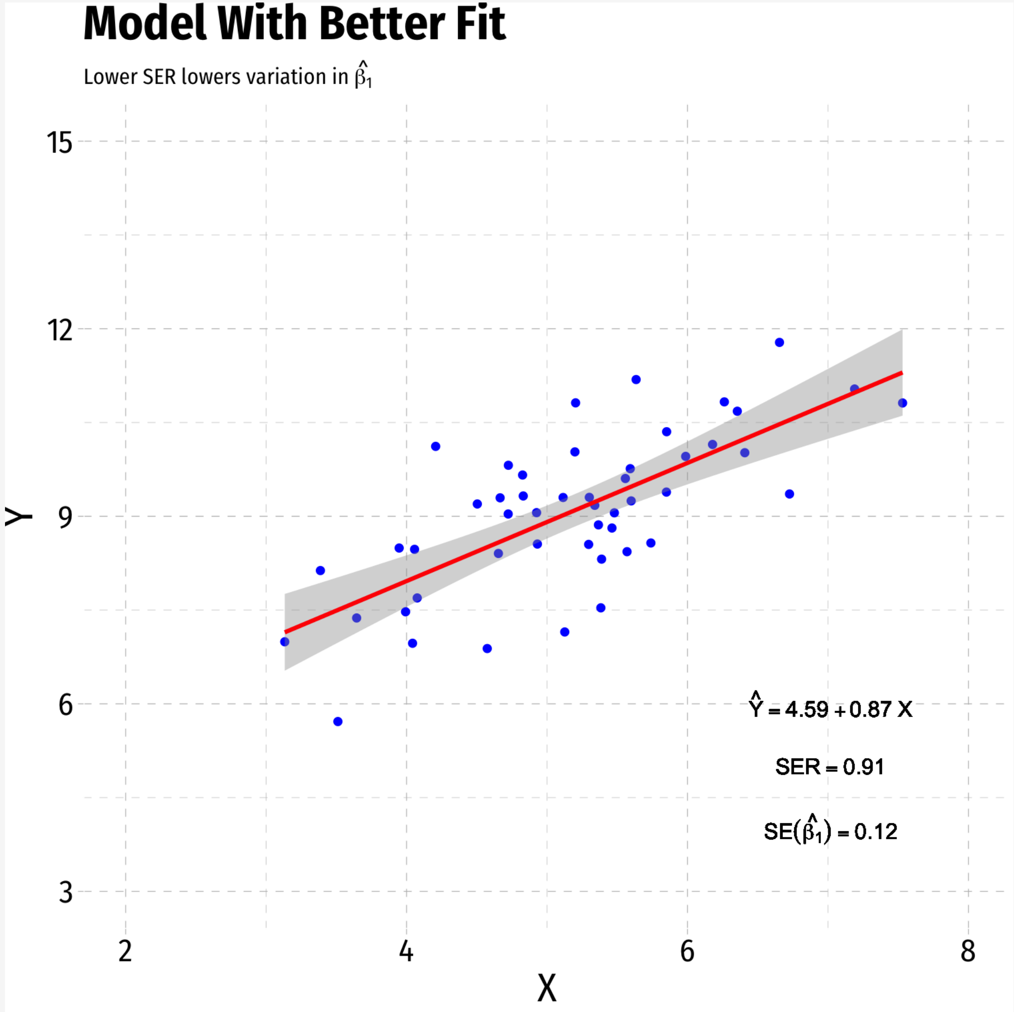
$$se(\hat{\beta}_1) = \sqrt{var(\hat{\beta}_1)} = \frac{SER}{\sqrt{n} \times sd(X)}$$

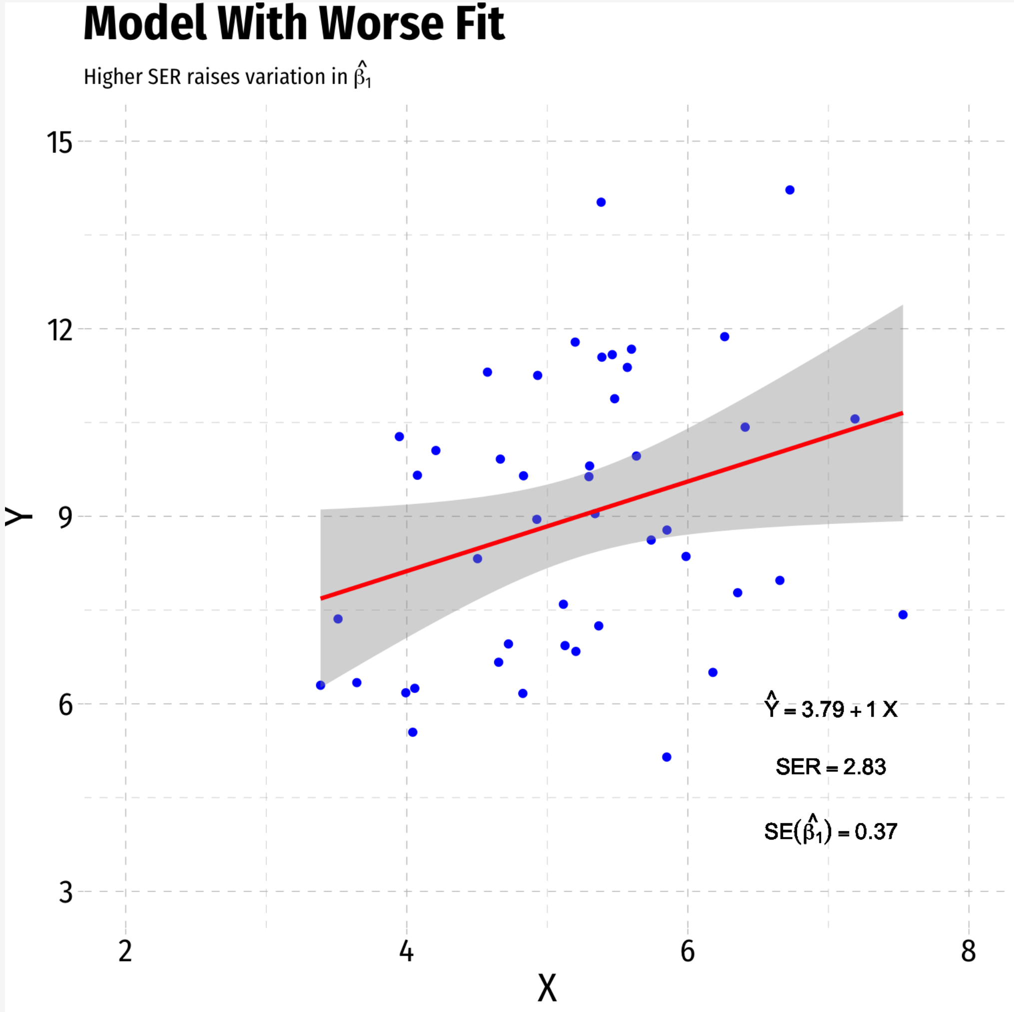
- Variation in $\hat{\beta}_1$ is affected by 3 things:

1. **Goodness of fit of the model (SER)**†
   - Larger $SER \rightarrow$ larger $var(\hat{\beta}_1)$
2. **Sample size, n**
   - Larger $n \rightarrow$ smaller $var(\hat{\beta}_1)$
3. **Variance of X**
   - Larger $var(X) \rightarrow$ smaller $var(\hat{\beta}_1)$

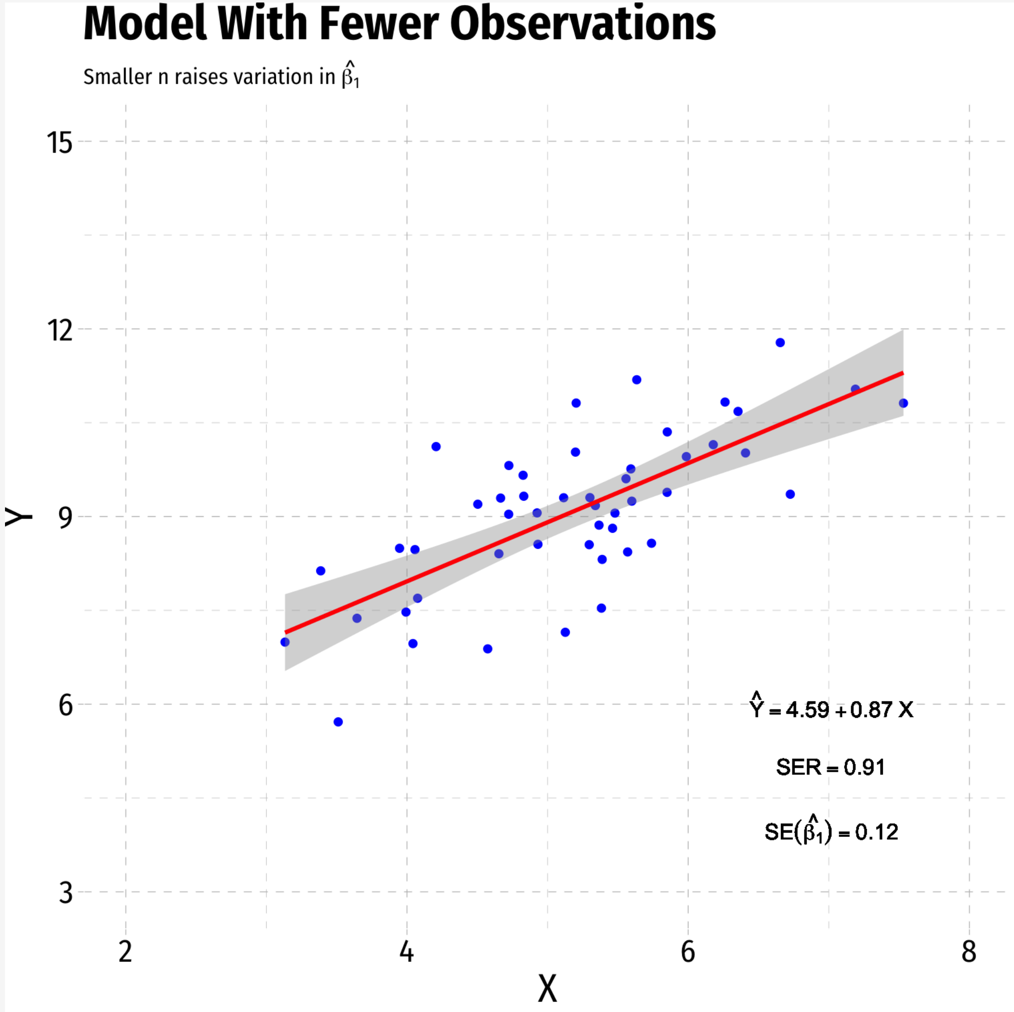
† Recall from last class, the **S**tandard **E**rror of the **R**egression $\hat{\sigma}_u = \sqrt{\frac{\sum \hat{u}_i^2}{n-2}}$

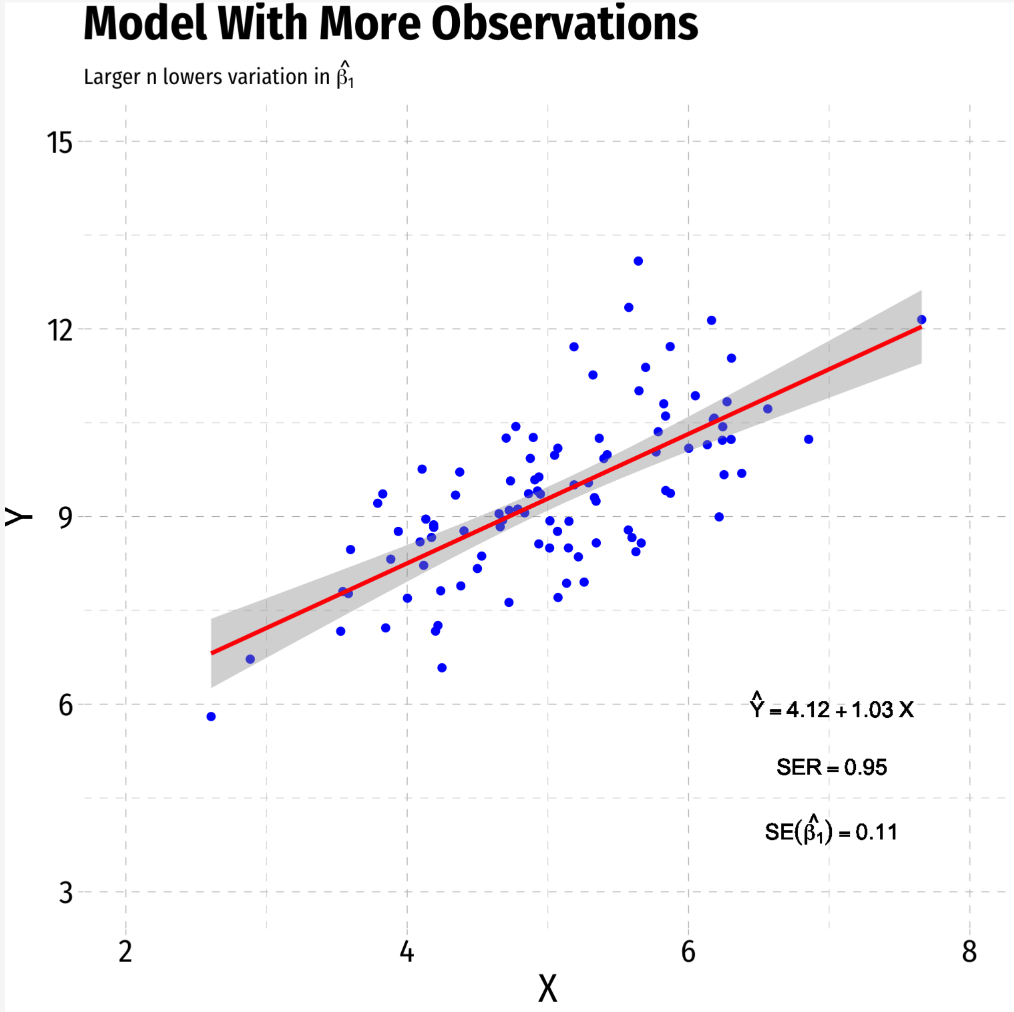
Variation in $\hat{\beta}_1$: Goodness of Fit



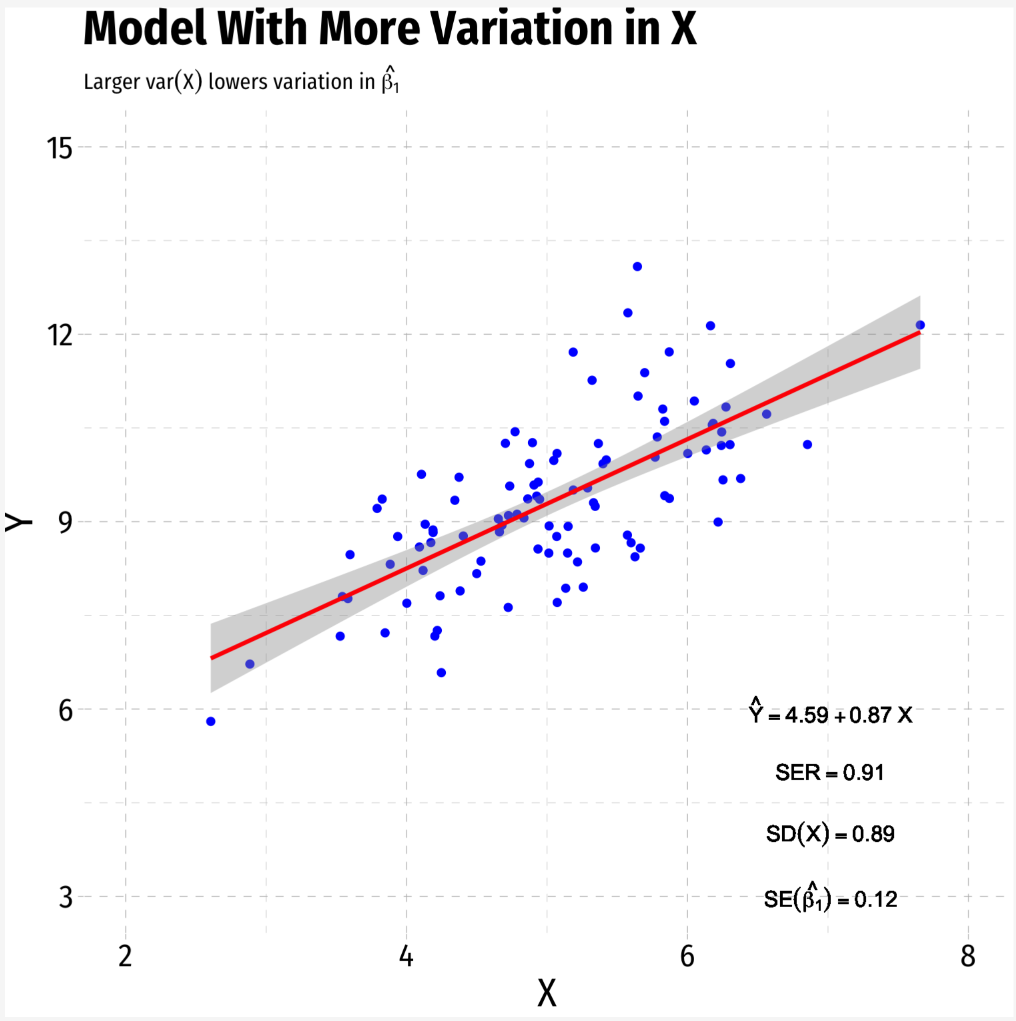


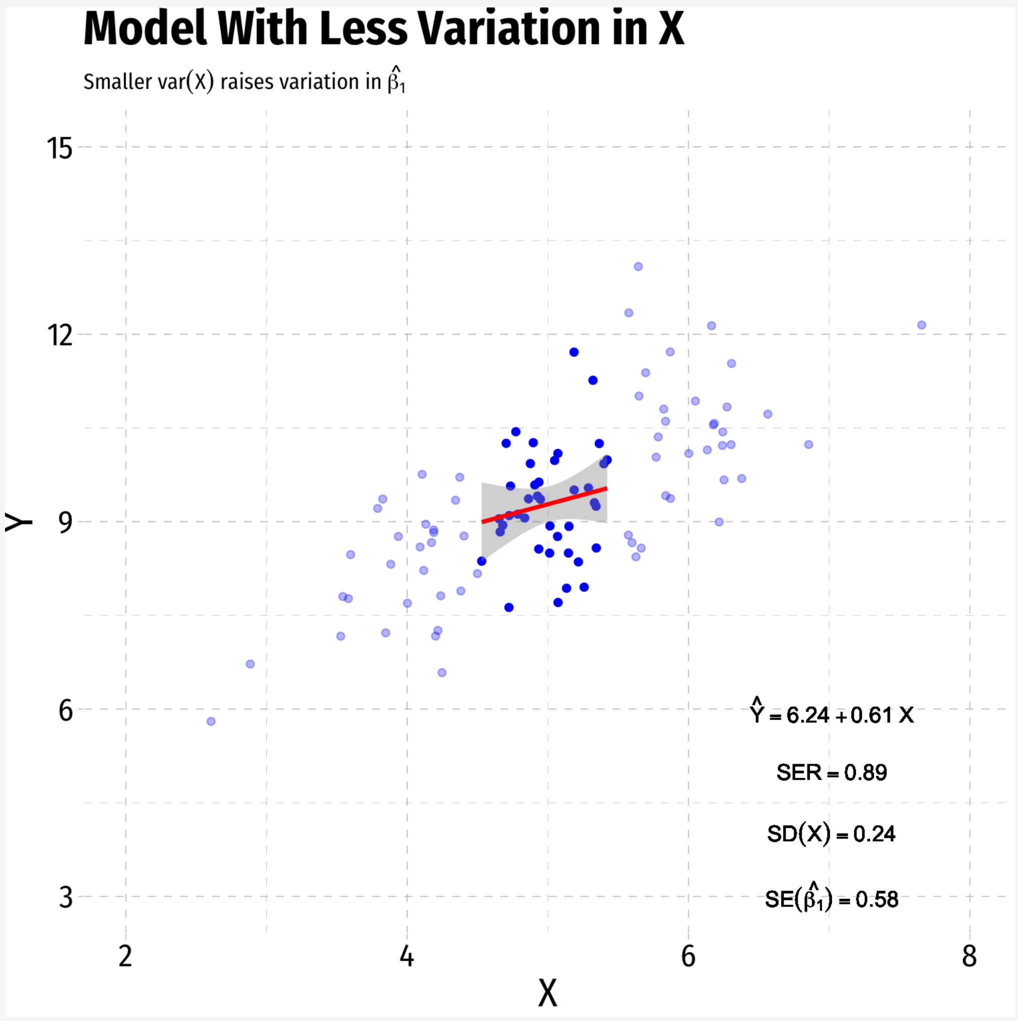
Variation in $\hat{\beta}_1$: Sample Size





Variation in $\hat{\beta}_1$: Variation in X





Presenting Regression Results

Our Class Size Regression: Base R

- How can we present all of this information in a tidy way?

```r
summary(school_reg) # get full summary
```

```
## 
## Call:
## lm(formula = testscr ~ str, data = CASchool)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -47.727 -14.251   0.483  12.822  48.540 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 698.9330     9.4675  73.825  < 2e-16 ***
## str          -2.2798     0.4798  -4.751 2.78e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 18.58 on 418 degrees of freedom
## Multiple R-squared:  0.05124,	Adjusted R-squared:  0.04897 
## F-statistic: 22.58 on 1 and 418 DF,  p-value: 2.783e-06
```

Our Class Size Regression: Broom I

- `broom`'s `tidy()` function creates a tidy tibble of regression output

```r
# load broom
library(broom)

# tidy regression output
tidy(school_reg)
```

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	698.932952	9.4674914	73.824514	6.569925e-242
str	-2.279808	0.4798256	-4.751327	2.783307e-06

2 rows

Our Class Size Regression: Broom II

- `broom`'s `glance()` gives us summary statistics about the regression

```
glance(school_reg)
```

r.squared <dbl>	adj.r.squared <dbl>	sigma <dbl>	statistic <dbl>	p.value <dbl>	df <dbl>	logLik <dbl>	AIC <dbl>
0.0512401	0.04897033	18.58097	22.57511	2.783307e-06	1	-1822.25	3650.499

1 row | 1-8 of 12 columns

Presenting Regressions in a Table

- Professional journals and papers often have a **regression table**, including:
  - Estimates of $\hat{\beta}_0$ and $\hat{\beta}_1$
  - Standard errors of $\hat{\beta}_0$ and $\hat{\beta}_1$ (often below, in parentheses)
  - Indications of statistical significance (often with asterisks)
  - Measures of regression fit: R^2, SER, etc
- Later: multiple rows & columns for multiple variables & models

	Test Score
Intercept	698.93 ***
	(9.47)
STR	-2.28 ***
	(0.48)
N	420
R-Squared	0.05
SER	18.58

*** p < 0.001; ** p < 0.01; * p < 0.05.

Regression Output with huxtable I

- You will need to first `install.packages("huxtable")`
- Load with `library(huxtable)`
- Command: `huxreg()`
- Main argument is the name of your `lm` object
- Default output is fine, but often we want to customize a bit

```r
# install.packages("huxtable")
library(huxtable)
huxreg(school_reg)
```

	(1)
(Intercept)	698.933 ***
	(9.467)
str	-2.280 ***
	(0.480)
N	420
R2	0.051
logLik	-1822.250
AIC	3650.499

*** p < 0.001; ** p < 0.01; * p < 0.05.

Regression Output with huxtable II

- Can give title to each column

```
"Test Score" = school_reg
```

- Can change name of coefficients from default

```
coefs = c("Intercept" = "(Intercept)",
          "STR" = "str")
```

- Decide what statistics to include, and rename them

```
statistics = c("N" = "nobs",
               "R-Squared" = "r.squared",
               "SER" = "sigma")
```

- Choose how many decimal places to round to

```
number_format = 2
```

Regression Output with huxtable III

```r
huxreg("Test Score" = school_reg,
       coefs = c("Intercept" = "(Intercept)",
                 "STR" = "str"),
       statistics = c("N" = "nobs",
                      "R-Squared" = "r.squared",
                      "SER" = "sigma"),
       number_format = 2)
```

	Test Score
Intercept	698.93 ***
	(9.47)
STR	-2.28 ***
	(0.48)
N	420
R-Squared	0.05
SER	18.58
*** p < 0.001; ** p < 0.01; * p < 0.05.	

Regression Outputs

- `huxtable` is one package you can use
  - See here for more options
- I used to only use `stargazer`, but as it was originally meant for STATA, it has limits and problems
  - A great cheetsheat by my friend Jake Russ

Diagnostics about Regression

Diagnostics: Residuals I

- We often look at the residuals of a regression to get more insight about its **goodness of fit** and its **bias**

- Recall `broom`'s `augment` creates some useful new variables
    - `.fitted` are fitted (predicted) values from model, i.e. $\hat{Y}_i$
    - `.resid` are residuals (errors) from model, i.e. $\hat{u}_i$

Diagnostics: Residuals II

- Often a good idea to store in a new object (so we can make some plots)

```
aug_reg<-augment(school_reg)

aug_reg %>% head()
```

testscr	str	.fitted	.resid	.std.resid	.hat	.sigma	.cooksd
691	17.9	658	32.7	1.76	0.00442	18.5	0.00689
661	21.5	650	11.3	0.612	0.00475	18.6	0.000893
644	18.7	656	-12.7	-0.685	0.00297	18.6	0.0007
648	17.4	659	-11.7	-0.629	0.00586	18.6	0.00117
641	18.7	656	-15.5	-0.836	0.00301	18.6	0.00105
606	21.4	650	-44.6	-2.4	0.00446	18.5	0.013

Recap: Assumptions about Errors

- We make **4 critical assumptions about u**:

1. The expected value of the residuals is 0

$$E[u] = 0$$

2. The variance of the residuals over X is constant:

$$var(u|X) = \sigma_u^2$$

3. Errors are not correlated across observations:

$$cor(u_i, u_j) = 0 \quad \forall i \neq j$$

4. There is no correlation between X and the error term:

$$cor(X, u) = 0 \text{ or } E[u|X] = 0$$

Assumptions 1 and 2: Errors are i.i.d.

- Assumptions 1 and 2 assume that errors are coming from the same (*normal*) distribution

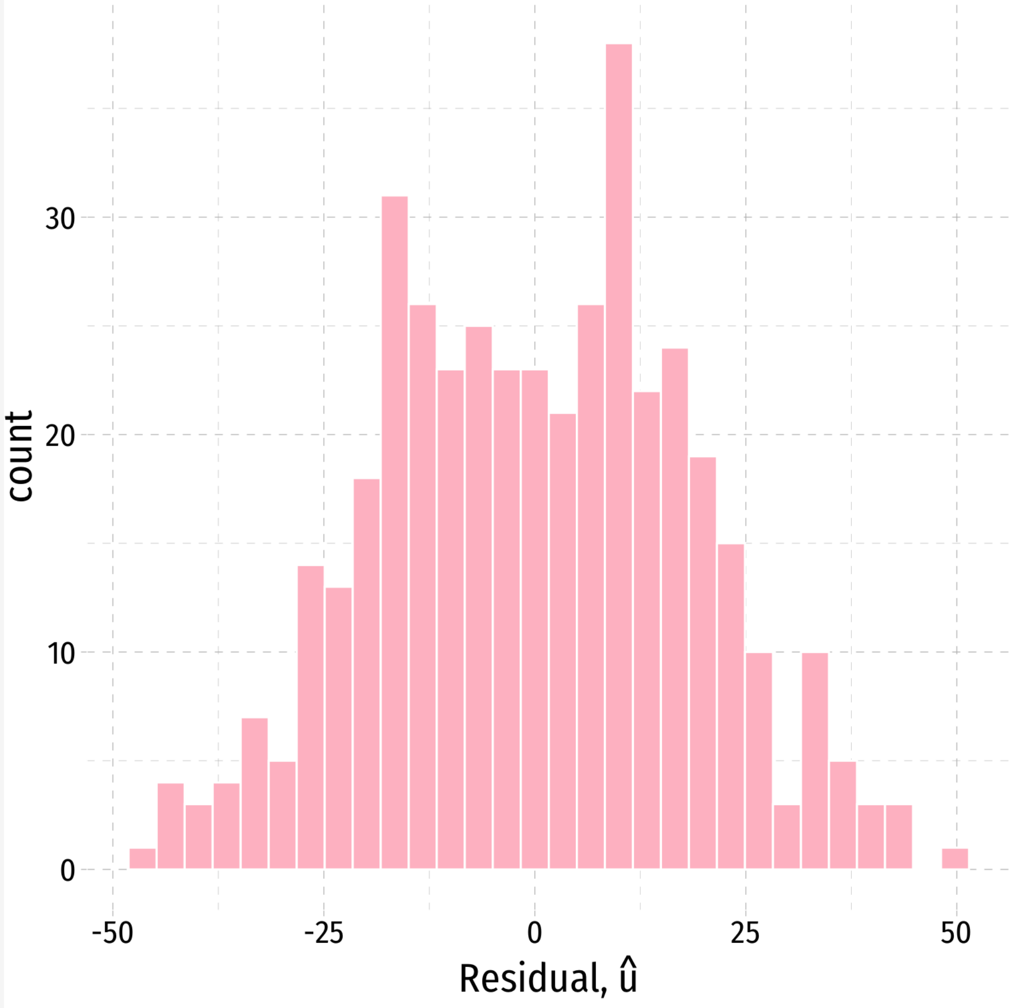
$$u \sim N(0, \sigma_u)$$

  - Assumption 1: $E[u] = 0$
  - Assumption 2: $sd(u|X) = \sigma_u$
    - virtually always unknown...

- We often can visually check by plotting a **histogram** of u

Plotting Residuals

```r
ggplot(data = aug_reg)+
  aes(x = .resid)+
  geom_histogram(color="white", fill = "pink")+
  labs(x = expression(paste("Residual, ", hat(u))))+
  theme_pander(base_family = "Fira Sans Condensed",
           base_size=20)
```



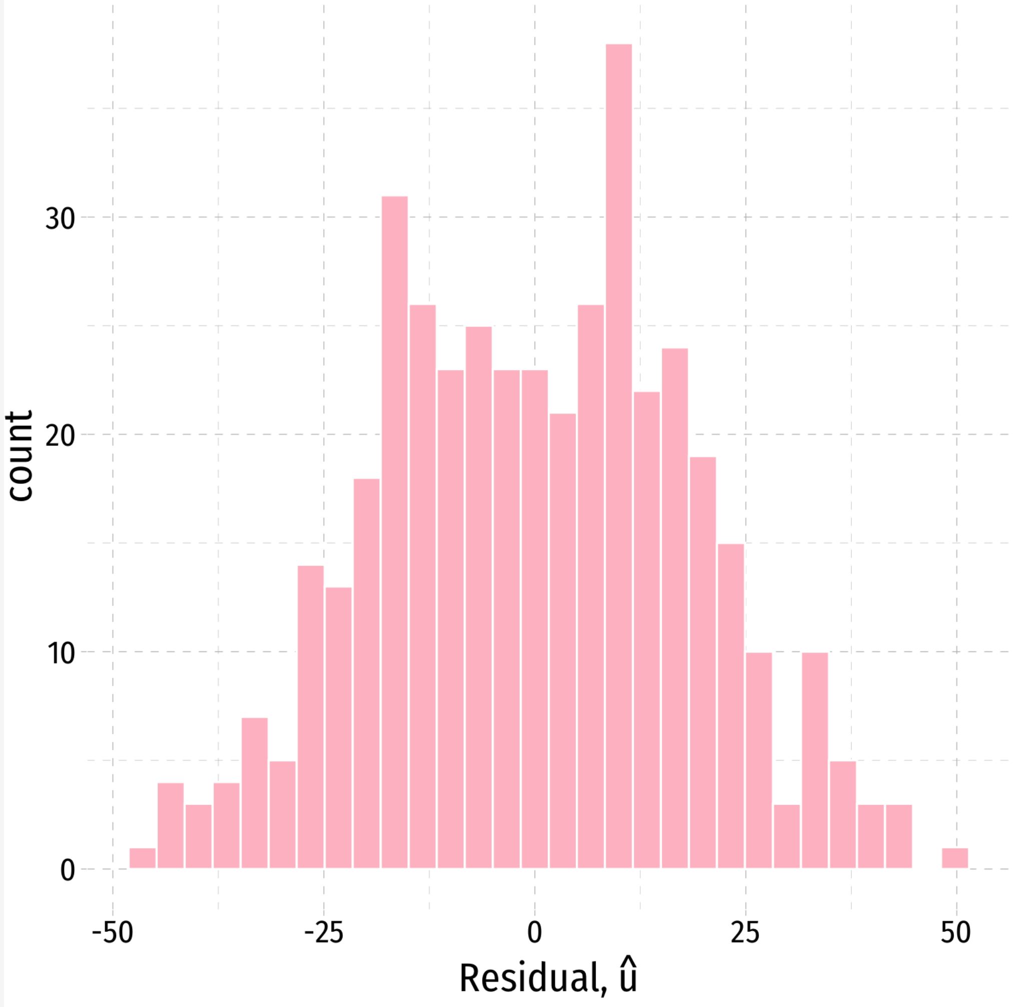
Plotting Residuals

```r
ggplot(data = aug_reg)+
  aes(x = .resid)+
  geom_histogram(color="white", fill = "pink")+
  labs(x = expression(paste("Residual, ", hat(u))))+
  theme_pander(base_family = "Fira Sans Condensed",
           base_size=20)
```

- Just to check:

```r
aug_reg %>%
  summarize(E_u = mean(.resid),
            sd_u = sd(.resid))
```

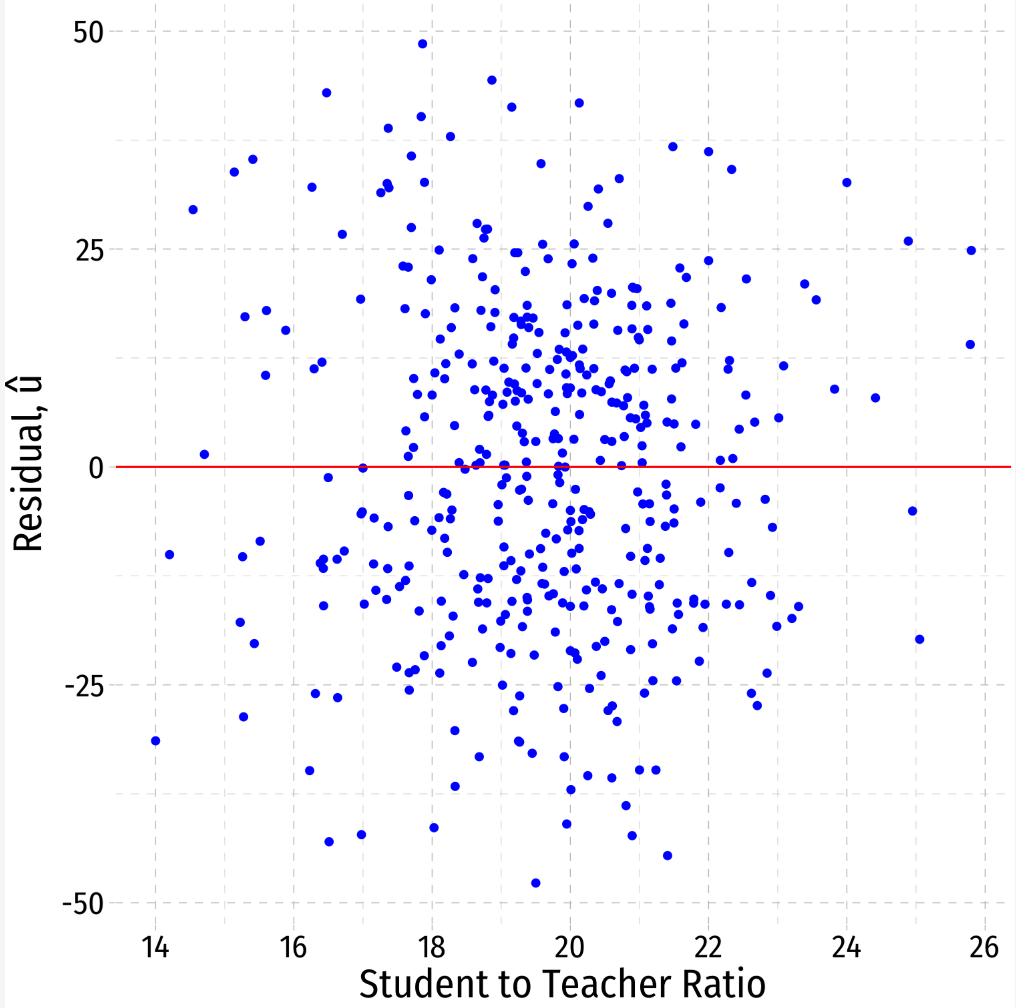
E_u	sd_u
3.7e-13	18.6



Residual Plot

- We often plot a **residual plot** to see any odd patterns about residuals
  - x-axis are X values (`str`)
  - y-axis are u values (`.resid`)

```
ggplot(data = aug_reg)+
  aes(x = str,
      y = .resid)+
  geom_point(color="blue")+
  geom_hline(aes(yintercept = 0), color="red")+
  labs(x = "Student to Teacher Ratio",
       y = expression(paste("Residual, ", hat(u))))
  theme_pander(base_family = "Fira Sans Condensed",
          base_size=20)
```



Problem: Heteroskedasticity

Homoskedasticity

- "**Homoskedasticity**:" variance of the residuals over X is constant, written:

$$var(u|X) = \sigma_u^2$$

- Knowing the value of X does not affect the variance (spread) of the errors

Heteroskedasticity I

- "**Heteroskedasticity**:" variance of the residuals over X is *NOT* constant:

$$var(u|X) \neq \sigma_u^2$$

- **This does not cause $\hat{\beta}_1$ to be biased**, but it does cause the standard error of $\hat{\beta}_1$ to be incorrect

- This **does** cause a problem for **inference**!

Heteroskedasticity II

- Recall the formula for the standard error of $\hat{\beta}_1$:

$$se(\hat{\beta}_1) = \sqrt{var(\hat{\beta}_1)} = \frac{SER}{\sqrt{n} \times sd(X)}$$

- This actually *assumes* homoskedasticity

Heteroskedasticity III

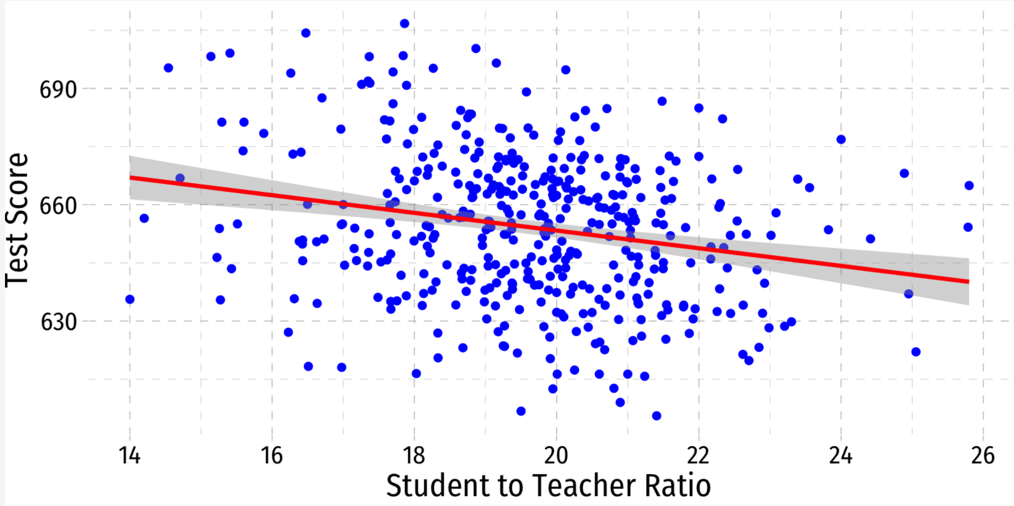
- Under heteroskedasticity, the standard error of $\hat{\beta}_1$ mutates to:

$$se(\hat{\beta}_1) = \sqrt{\frac{\displaystyle\sum_{i=1}^{n}(X_i - \bar{X})^2 \hat{u}^2}{\left[\displaystyle\sum_{i=1}^{n}(X_i - \bar{X})^2\right]^2}}$$

- This is a **heteroskedasticity-robust** (or just **"robust"**) method of calculating $se(\hat{\beta}_1)$
- Don't learn formula, **do learn what heteroskedasticity is and how it affects our model!**

Visualizing Heteroskedasticity I

- Our original scatterplot with regression line

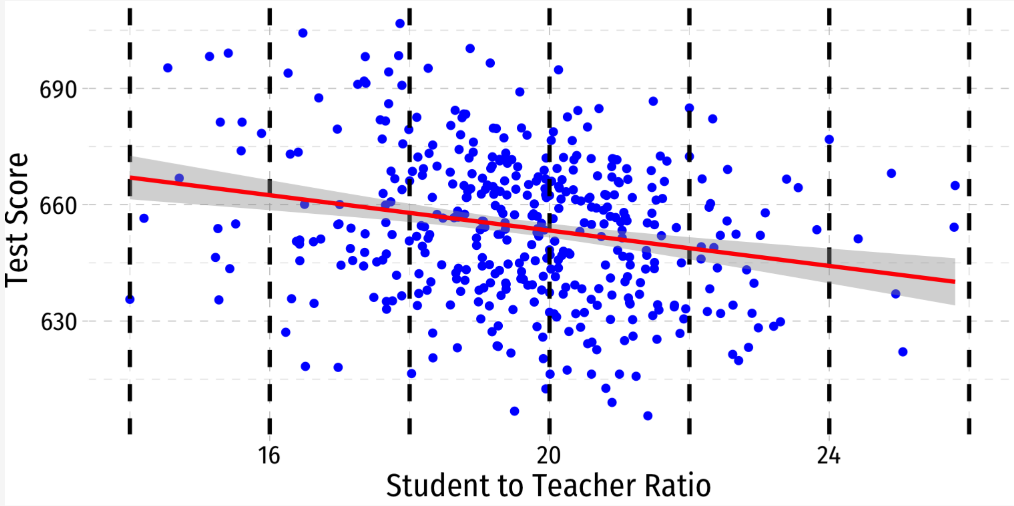


Visualizing Heteroskedasticity I

- Our original scatterplot with regression line

- Does the spread of the errors change over different values of str?

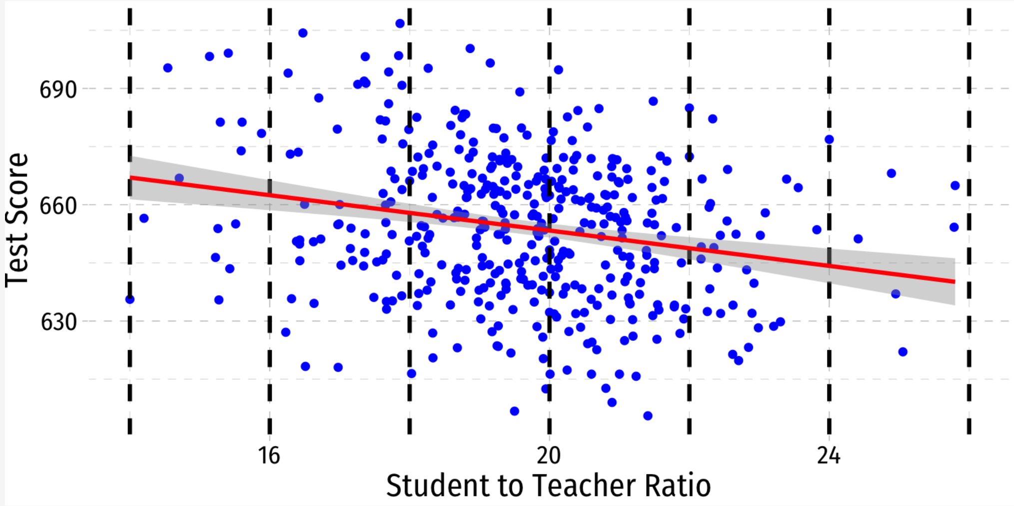
  - No: homoskedastic
  - Yes: heteroskedastic



Visualizing Heteroskedasticity I

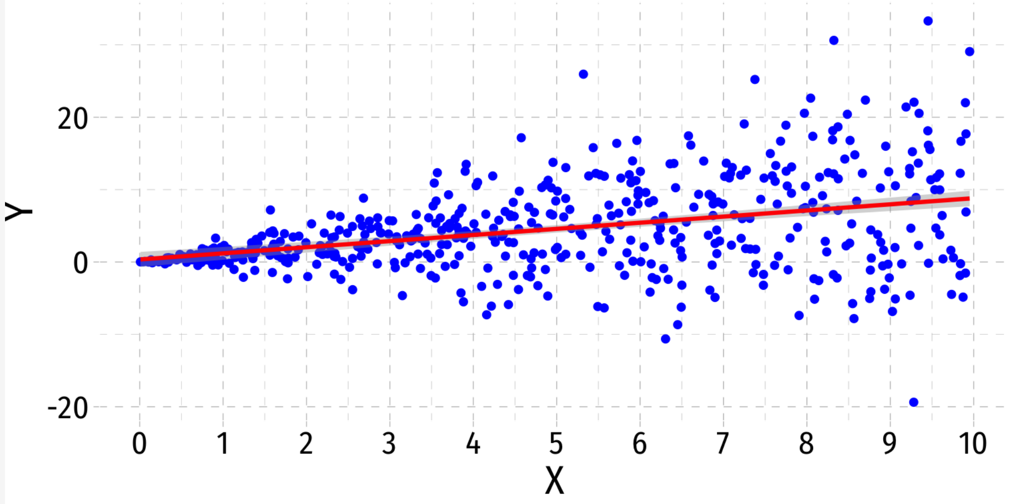
- Our original scatterplot with regression line

- Does the spread of the errors change over different values of str?
  - No: homoskedastic
  - Yes: heteroskedastic



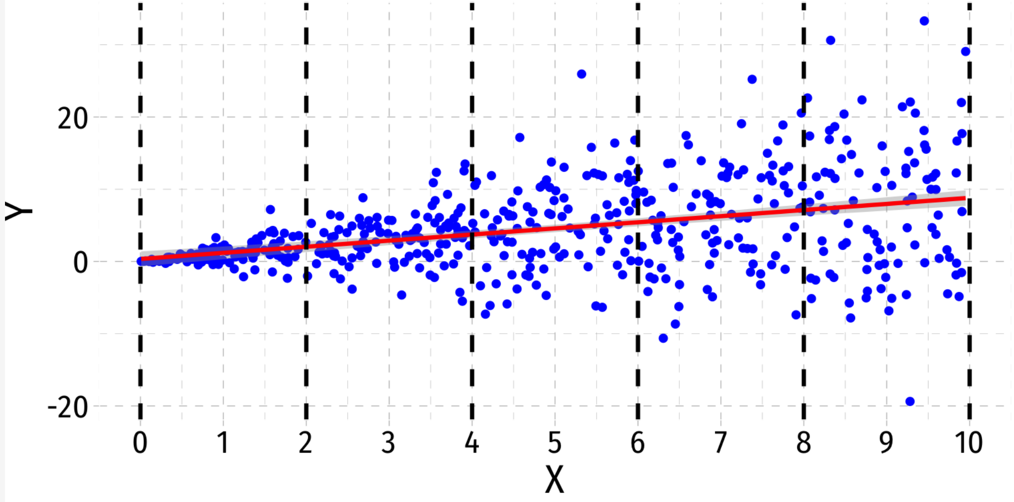
More Obvious Heteroskedasticity

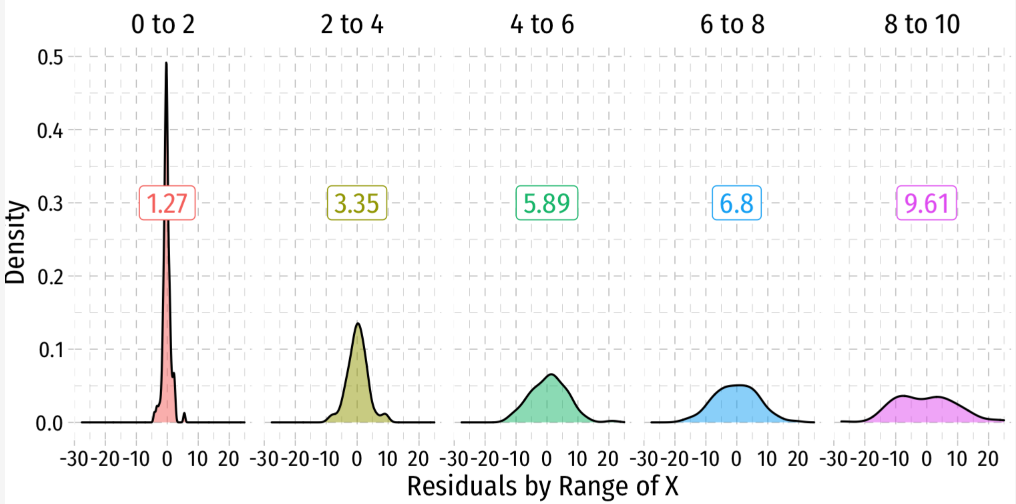
- Visual cue: data is "fan-shaped"
    - Data points are closer to line in some areas
    - Data points are more spread from line in other areas



More Obvious Heteroskedasticity

- Visual cue: data is "fan-shaped"
  - Data points are closer to line in some areas
  - Data points are more spread from line in other areas

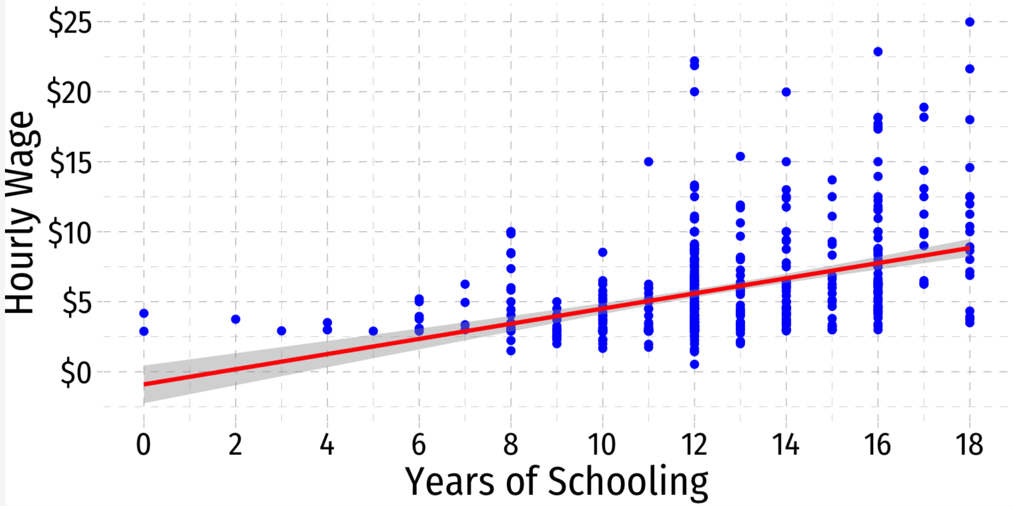




What Might Cause Heteroskedastic Errors?

$$\widehat{wage_i} = \hat{\beta}_0 + \hat{\beta}_1 educ_i$$

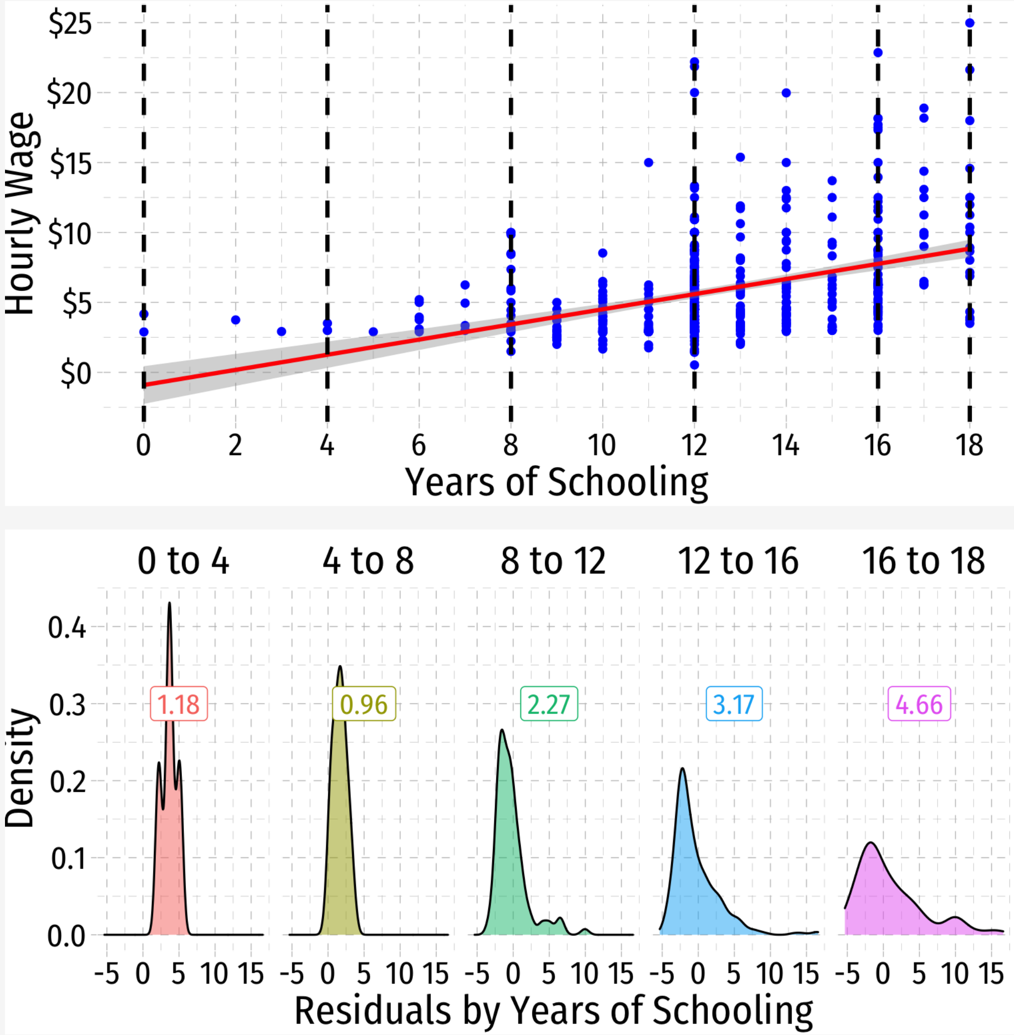
	Wage
Intercept	-0.90
	(0.68)
Years of Schooling	0.54 ***
	(0.05)
N	526
R-Squared	0.16



What Might Cause Heteroskedastic Errors?

$$\widehat{wage_i} = \hat{\beta}_0 + \hat{\beta}_1 educ_i$$

	Wage
Intercept	-0.90
	(0.68)
Years of Schooling	0.54 ***
	(0.05)
N	526
R-Squared	0.16
SER	3.38



Detecting Heteroskedasticity I

- Several tests to check if data is heteroskedastic
- One common test is **Breusch-Pagan test**
- Can use `bptest()` with `lmtest` package in `R`
  - H_0: homoskedastic
  - If p-value < 0.05, reject H_0 $\implies$ heteroskedastic

```r
# install.packages("lmtest")
library("lmtest")
bptest(school_reg)
```

```
## 
##  studentized Breusch-Pagan test
## 
## data:  school_reg
## BP = 5.7936, df = 1, p-value = 0.01608
```

Detecting Heteroskedasticity II

- How about our wage regression?

```r
# install.packages("lmtest")
library("lmtest")
bptest(wage_reg)
```

```
##
##      studentized Breusch-Pagan test
##
## data:  wage_reg
## BP = 15.306, df = 1, p-value = 9.144e-05
```

Fixing Heteroskedasticity I

- Heteroskedasticity is easy to fix with software that can calculate **robust** standard errors (using the more complicated formula above)

- Easiest method is to use `estimatr` package
  - `lm_robust()` command (instead of `lm`) to run regression
  - set `se_type="stata"` to calculate robust SEs using the formula above

```
#install.packages("estimatr")
library(estimatr)

school_reg_robust <-lm_robust(testscr ~ str, data = CASchool,
                              se_type = "stata")

school_reg_robust
```

```
##               Estimate Std. Error   t value      Pr(>|t|)   CI Lower   CI Upper
## (Intercept) 698.932952 10.3643599 67.436191 9.486678e-227 678.560192 719.305713
## str          -2.279808  0.5194892 -4.388557  1.446737e-05  -3.300945  -1.258671
##               DF
```

Fixing Heteroskedasticity II

```r
library(huxtable)
huxreg("Normal" = school_reg,
       "Robust" = school_reg_robust,
       coefs = c("Intercept" = "(Intercept)",
                 "STR" = "str"),
       statistics = c("N" = "nobs",
                      "R-Squared" = "r.squared",
                      "SER" = "sigma"),
       number_format = 2)
```

	Normal	Robust
Intercept	698.93 ***	698.93 ***
	(9.47)	(10.36)
STR	-2.28 ***	-2.28 ***
	(0.48)	(0.52)
N	420	420
R-Squared	0.05	0.05
SER	18.58	

*** p < 0.001; ** p < 0.01; * p < 0.05.

Assumption 3: No Serial Correlation

- Errors are not correlated across observations:

$$cor(u_i, u_j) = 0 \quad \forall i \neq j$$

- For simple cross-sectional data, this is rarely an issue

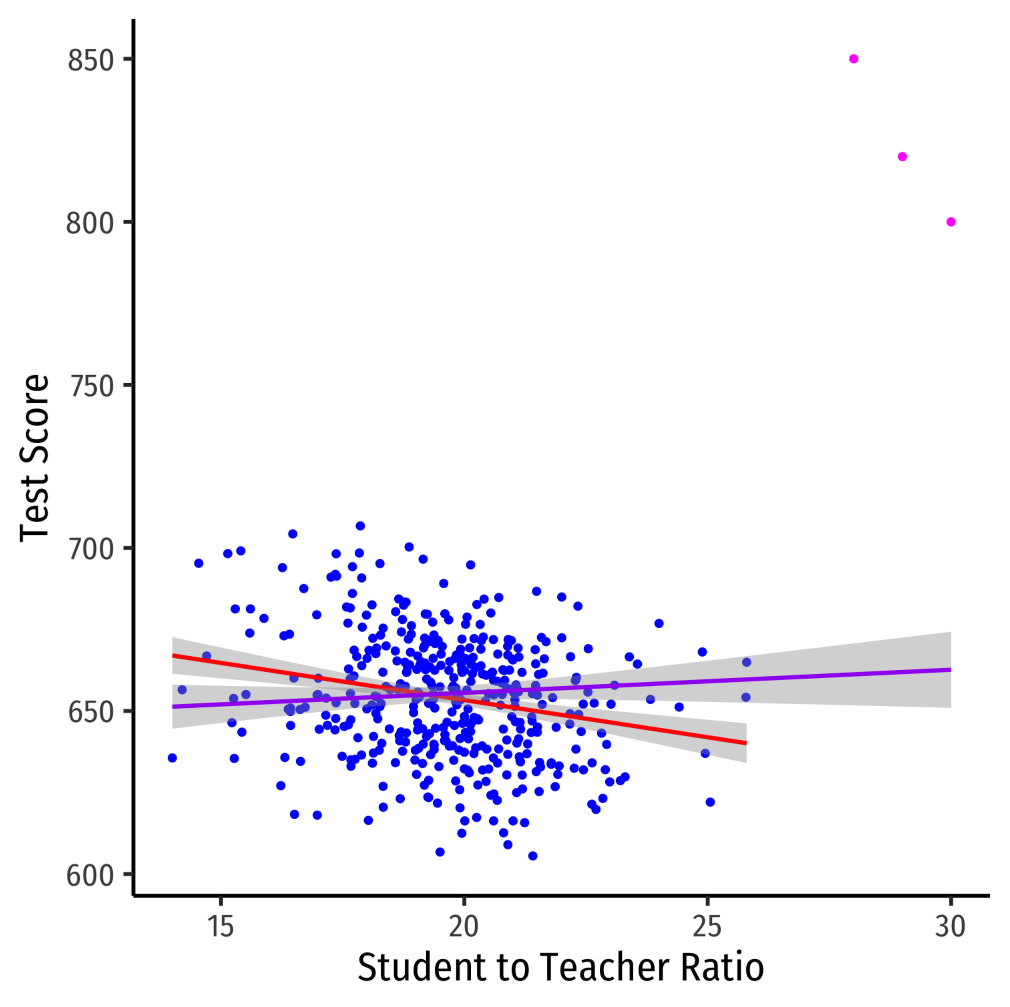
- Time-series & panel data nearly always contain **serial correlation** or **autocorrelation** between errors

- Errors may be **clustered**
  - **by group**: e.g. all observations from Maryland, all observations from Virginia, etc.
  - **by time**: GDP in 2006 around the world, GDP in 2008 around the world, etc.

Outliers

Outliers Can Bias OLS! I

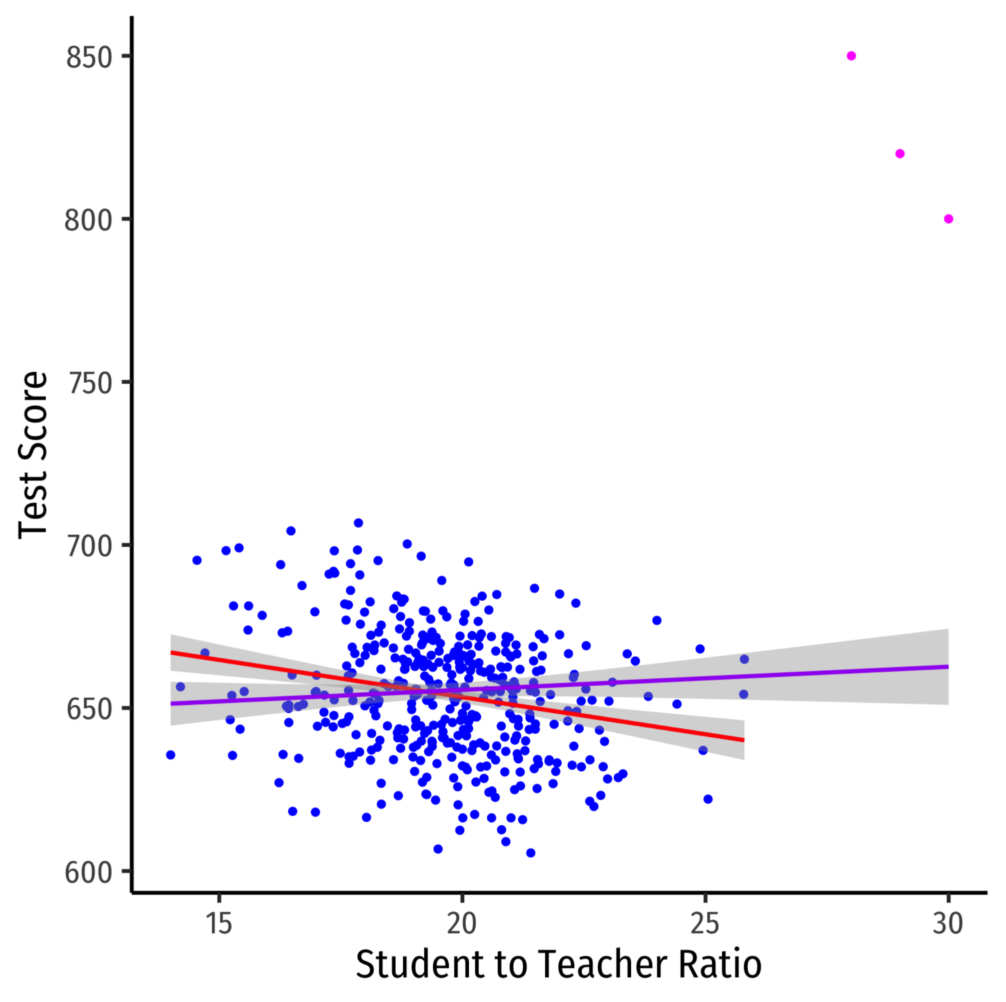
- **Outliers** can affect the slope (and intercept) of the line and add **bias**
  - May be result of human error (measurement, transcribing, etc)
  - May be meaningful and accurate
- In any case, compare how including/dropping outliers affects regression and always discuss outliers!



Outliers Can Bias OLS! II

```r
huxreg("No Outliers" = school_reg,
       "Outliers" = school_outlier_reg,
       coefs = c("Intercept" = "(Intercept)",
                 "STR" = "str"),
       statistics = c("N" = "nobs",
                      "R-Squared" = "r.squared"
                      "SER" = "sigma"),
       number_format = 2)
```

	No Outliers	Outliers
Intercept	698.93 ***	641.40 ***
	(9.47)	(11.21)
STR	-2.28 ***	0.71
	(0.48)	(0.57)



Detecting Outliers

- The `car` package has an `outlierTest` command to run on the regression

```r
library("car")

# Use Bonferonni test
outlierTest(school_outlier_reg) # will point out which obs #s seem outliers
```

```
##     rstudent unadjusted p-value Bonferroni p
## 422 8.822768         3.0261e-17   1.2800e-14
## 423 7.233470         2.2493e-12   9.5147e-10
## 421 6.232045         1.1209e-09   4.7414e-07
```

```r
# find these observations
CA.outlier %>%
  slice(c(422,423,421))
```

observat	district	testscr	str
422	Crazy School 2	850	28
423	Crazy School 3	820	29
421	Crazy School 1	800	30