Lecture 9: Regression Diagnostics and Plotting with `ggplot2`

ECON 480 - Econometrics - Fall 2018

Ryan Safner

September 26, 2018

Solvable Regression Problem #1: Heteroskedasticity

Solvable Regression Problem #2: Outliers

Advanced Plotting in `R` with `ggplot2`

Solvable Regression Problem #1: Heteroskedasticity

Homoskedasticity

- Recall assumption #2 about the regression residuals (ϵ) are that they are homoskedastic:

$$var(\epsilon) = \sigma_{\epsilon}^{2}$$

Homoskedasticity

- Recall assumption #2 about the regression residuals (ϵ) are that they are homoskedastic:

$$var(\epsilon) = \sigma^2_\epsilon$$

- A fancy way of saying that the **variance of the residuals is constant**, i.e. does not change over values of X

- Recall assumption #2 about the regression residuals (ϵ) are that they are homoskedastic:

$$var(\epsilon) = \sigma^2_\epsilon$$

- A fancy way of saying that the **variance of the residuals is constant**, i.e. does not change over values of X
- Combined with assumption #1 (the mean of the residuals $E[\epsilon] = 0$) $\implies$ residuals are i.i.d. and come from the same distribution $\sim (0, \sigma^2_\epsilon)$

Homoskedasticity

- If residuals are heteroskedastic, they do *not* have the same variance over values of X

$$var(\epsilon) \neq \sigma_\epsilon^2$$

- If residuals are heteroskedastic, they do *not* have the same variance over values of X

$$var(\epsilon) \neq \sigma^2_\epsilon$$

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

- If residuals are heteroskedastic, they do *not* have the same variance over values of X

$$var(\epsilon) \neq \sigma_\epsilon^2$$

- **This does not cause $\hat{\beta}_1$ to be biased**, but it does cause the standard error of $\hat{\beta}_1$ to be incorrect
  - Recall, $se(\hat{\beta}_1)$ is used to calculate our test statistic for hypothesis testing

Homoskedasticity

- If residuals are heteroskedastic, they do *not* have the same variance over values of X

$$var(\epsilon) \neq \sigma^2_\epsilon$$

- **This does not cause $\hat{\beta}_1$ to be biased**, but it does cause the standard error of $\hat{\beta}_1$ to be incorrect
    - Recall, $se(\hat{\beta}_1)$ is used to calculate our test statistic for hypothesis testing
    - **May overstate the statistical significance of a finding!**

- The formula for $se(\hat{\beta}_1)$ assumes homoskedasticity, recall (from Lecture 8) it was:

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

Homoskedasticity

- The formula for $se(\hat{\beta}_1)$ assumes homoskedasticity, recall (from Lecture 8) it was:

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

- When errors are heteroskedastic, the formula mutates to

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

- The formula for $se(\hat{\beta}_1)$ assumes homoskedasticity, recall (from Lecture 8) it was:

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

- When errors are heteroskedastic, the formula mutates to

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

- This is heteroskedasticity-robust (“robust”) method of calculating $se(\hat{\beta}_1)$

Homoskedasticity

- The formula for $se(\hat{\beta}_1)$ assumes homoskedasticity, recall (from Lecture 8) it was:

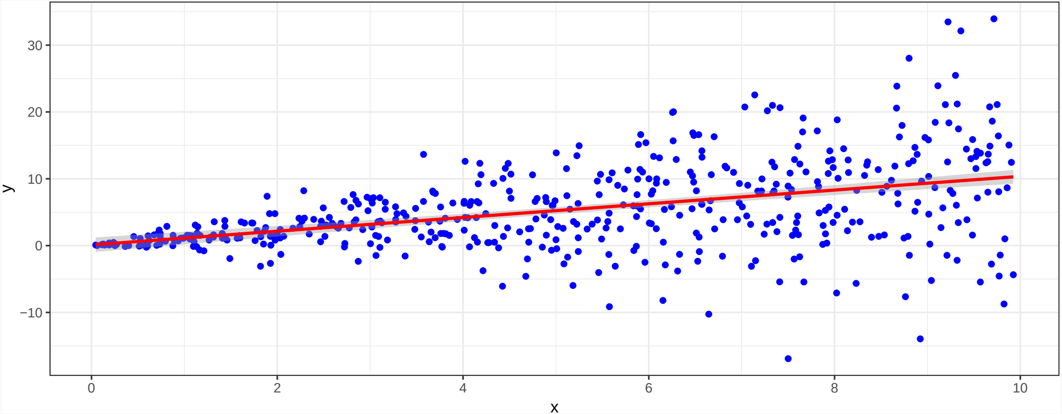
$$se(\hat{\beta}_1) = \sqrt{\frac{(SER)^2}{n \times var(X)}}$$

- When errors are heteroskedastic, the formula mutates to

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

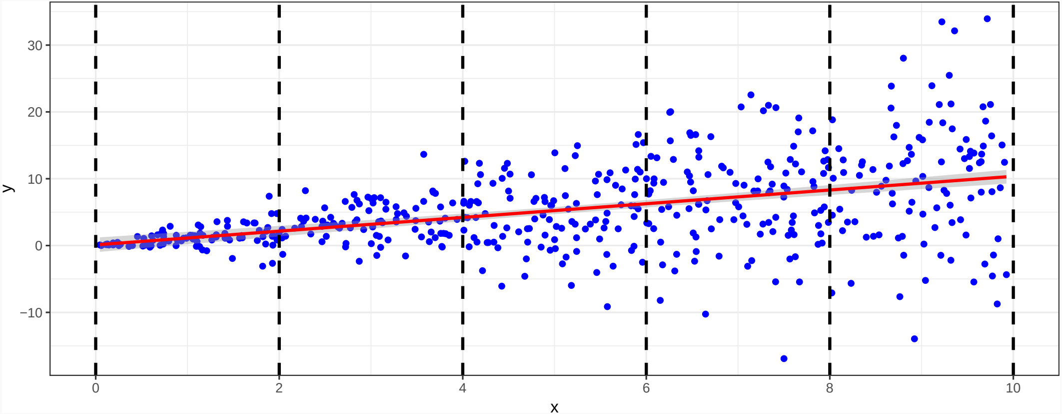
- This is heteroskedasticity-robust (“robust”) method of calculating $se(\hat{\beta}_1)$
- Don’t learn formula, **do learn what heteroskedasticity is and how it affects our model!**

Visualizing Heteroskedasticity



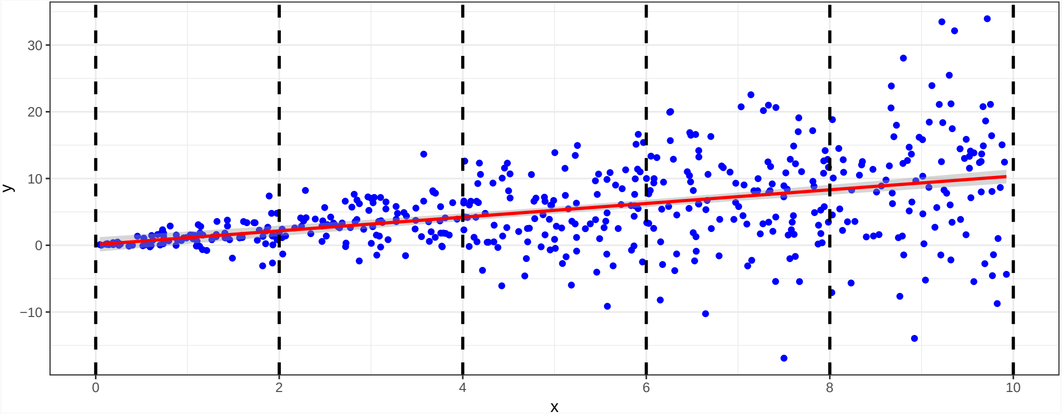
- The average residual (distance from point to OLS line) changes as X changes

Visualizing Heteroskedasticity II

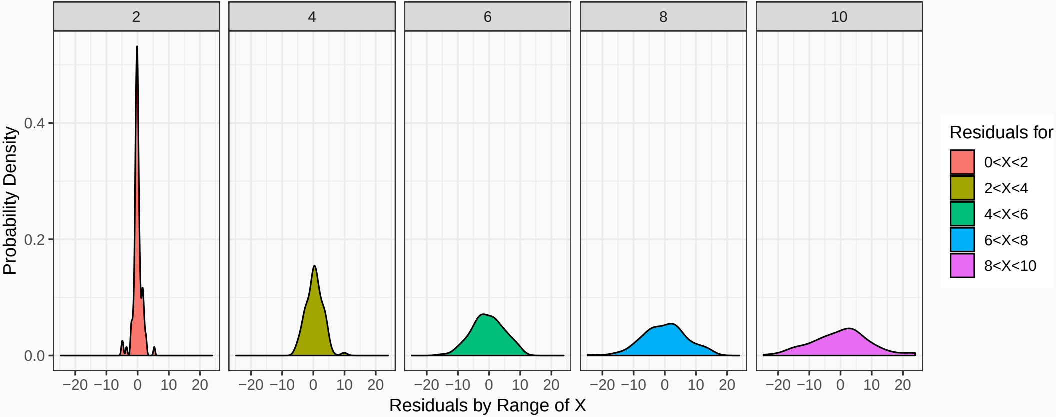


- We would expect the distribution of the residuals ($\hat{\epsilon}$) to be the same at every value of X

Visualizing Heteroskedasticity II

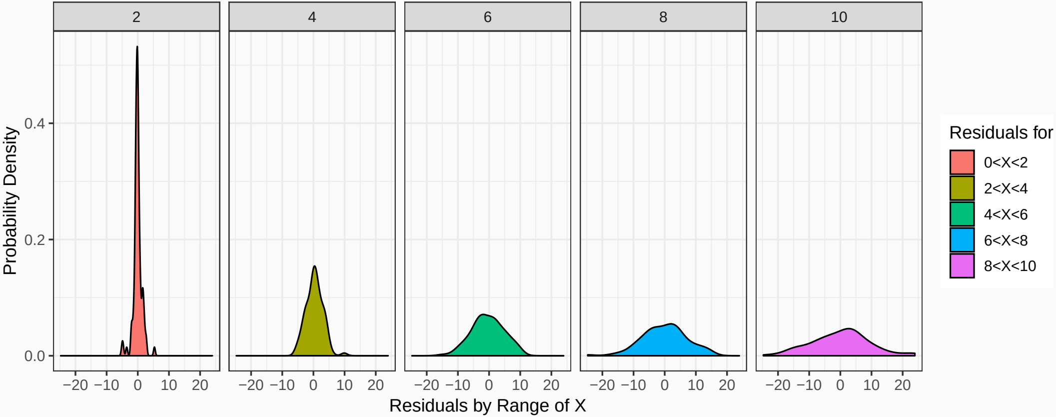


- We would expect the distribution of the residuals ($\hat{\epsilon}$) to be the same at every value of X
- Clearly there are very different distributions of residuals across X



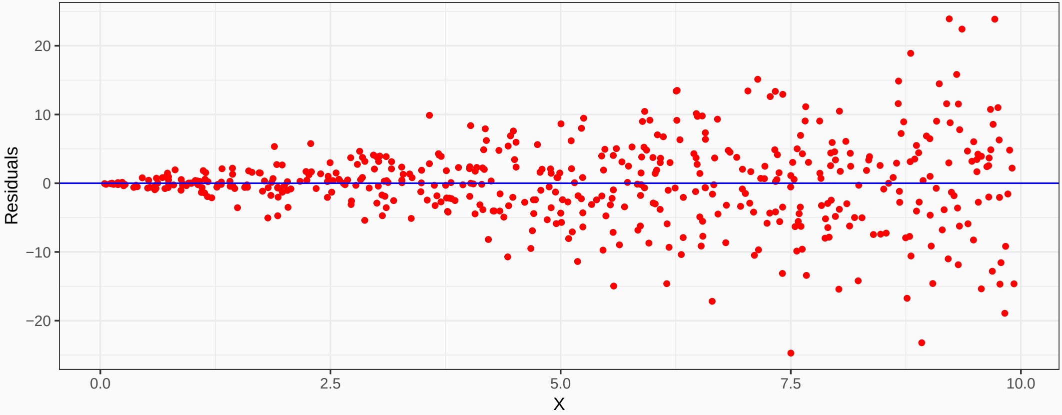
- We would expect the distribution of the residuals ($\hat{\epsilon}$) to be the same at every value of X

Visualizing Heteroskedasticity III



- We would expect the distribution of the residuals ($\hat{\epsilon}$) to be the same at every value of X
- Clearly there are very different distributions of residuals across X

Visualizing Heteroskedasticity IV



- We can also see in the residual plot that the size of residuals increases as X increases

Detecting Heteroskedasticity

- Several tests to check if data is heteroskedastic

Detecting Heteroskedasticity

- Several tests to check if data is heteroskedastic
- One common test is **Breusch-Pagan test**

Detecting Heteroskedasticity

- Several tests to check if data is heteroskedastic
- One common test is **Breusch-Pagan test**
- Can use `bptest()` with `lmtest` package in R

Detecting Heteroskedasticity

- 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

Detecting Heteroskedasticity

- 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

- 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

```
library("lmtest") #load lmtest package, install if first time
bptest(het.reg)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  het.reg
## BP = 77.934, df = 1, p-value < 2.2e-16
```

- Before we generate robust SE's, let's look at plain SEs

- Before we generate robust SE's, let's look at plain SEs
- Regression creates a **variance-covariance** matrix ('`vcov`') of OLS estimators (betas), e.g.

$$\begin{bmatrix} cov(\hat{\beta}_0, \hat{\beta}_0) & cov(\hat{\beta}_0, \hat{\beta}_1) \\ cov(\hat{\beta}_0, \hat{\beta}_1) & cov(\hat{\beta}_1, \hat{\beta}_1) \end{bmatrix}$$

- Before we generate robust SE's, let's look at plain SEs
- Regression creates a **variance-covariance** matrix ('`vcov`') of OLS estimators (betas), e.g.

$$\begin{bmatrix} cov(\hat{\beta}_0, \hat{\beta}_0) & cov(\hat{\beta}_0, \hat{\beta}_1) \\ cov(\hat{\beta}_0, \hat{\beta}_1) & cov(\hat{\beta}_1, \hat{\beta}_1) \end{bmatrix}$$

- Why a matrix?

Standard Errors: Technical Digression

- Before we generate robust SE's, let's look at plain SEs
- Regression creates a **variance-covariance** matrix ('`vcov`') of OLS estimators (betas), e.g.

$$\begin{bmatrix} cov(\hat{\beta}_0, \hat{\beta}_0) & cov(\hat{\beta}_0, \hat{\beta}_1) \\ cov(\hat{\beta}_0, \hat{\beta}_1) & cov(\hat{\beta}_1, \hat{\beta}_1) \end{bmatrix}$$

- Why a matrix?
  - We currently have 2 OLS estimators $(\hat{\beta}_0, \hat{\beta}_1)$ for 1 X_i variable

- Before we generate robust SE's, let's look at plain SEs
- Regression creates a **variance-covariance** matrix ('`vcov`') of OLS estimators (betas), e.g.

$$\begin{bmatrix} cov(\hat{\beta}_0, \hat{\beta}_0) & cov(\hat{\beta}_0, \hat{\beta}_1) \\ cov(\hat{\beta}_0, \hat{\beta}_1) & cov(\hat{\beta}_1, \hat{\beta}_1) \end{bmatrix}$$

- Why a matrix?
  - We currently have 2 OLS estimators $(\hat{\beta}_0, \hat{\beta}_1)$ for 1 X_i variable
  - We eventually will have k X_i variables, so a $\big(k+1 \times k+1\big)$ matrix

- Before we generate robust SE's, let's look at plain SEs
- Regression creates a **variance-covariance** matrix (`vcov`) of OLS estimators (betas), e.g.

$$\begin{bmatrix} cov(\hat{\beta}_0, \hat{\beta}_0) & cov(\hat{\beta}_0, \hat{\beta}_1) \\ cov(\hat{\beta}_0, \hat{\beta}_1) & cov(\hat{\beta}_1, \hat{\beta}_1) \end{bmatrix}$$

- Why a matrix?
  - We currently have 2 OLS estimators $(\hat{\beta}_0, \hat{\beta}_1)$ for 1 X_i variable
  - We eventually will have k X_i variables, so a $\left(k+1 \times k+1\right)$ matrix
- The 'diagonal' of the matrix contains the variance of each OLS estimator, since $cov(X, X) = var(X)$

Standard Errors: Technical Digression

- Before we generate robust SE's, let's look at plain SEs
- Regression creates a **variance-covariance** matrix ('`vcov`') of OLS estimators (betas), e.g.

$$\begin{bmatrix} cov(\hat{\beta}_0, \hat{\beta}_0) & cov(\hat{\beta}_0, \hat{\beta}_1) \\ cov(\hat{\beta}_0, \hat{\beta}_1) & cov(\hat{\beta}_1, \hat{\beta}_1) \end{bmatrix}$$

- Why a matrix?
  - We currently have 2 OLS estimators $(\hat{\beta}_0, \hat{\beta}_1)$ for 1 X_i variable
  - We eventually will have k X_i variables, so a $\big(k+1 \times k+1\big)$ matrix
- The 'diagonal' of the matrix contains the variance of each OLS estimator, since $cov(X, X) = var(X)$
- Taking the diagonal and square rooting the terms gives us the SE of each estimator

Standard Errors: In R

```
# the variance-covariance matrix
vcov(het.reg)
```

```
##             (Intercept)           x
## (Intercept)  0.30764996 -0.04573739
## x           -0.04573739  0.00885630
```

```
#the var(beta)'s are the diagonal of the matrix

diag(vcov(het.reg)) # look at just the diagonal values
```

```
## (Intercept)           x
##   0.3076500   0.0088563
```

```
#convert into standard errors by square rooting
sqrt(diag(vcov(het.reg)))
```

```
## (Intercept)           x
##  0.55466202  0.09410792
```

Standard Errors: In R II

```r
# confirming the SE's match what R finds automatically with lm()
summary(het.reg)
```

```
##
## Call:
## lm(formula = y ~ x, data = het.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -24.7135  -2.7754  -0.0032   2.9312  23.9151
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.10384    0.55466   0.187    0.852
## x            1.02592    0.09411  10.902   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.977 on 498 degrees of freedom
## Multiple R-squared:  0.1927, Adjusted R-squared:  0.191
```

Using Robust Standard Errors in R

```r
library("sandwich") # package that allows for robust SE estimation, install if first use
library("lmtest") # package that allows for coefficient tests, install if first use

# take original regression and change standard errors to robust SEs #

# create Robust Standard Errors for regression as 'het.reg$rse'
het.reg$rse <-sqrt(diag(vcovHC(het.reg, type="HC1")))
# same procedure as above but now we generate vcov with "HC1" method (technical)
```

Using Robust Standard Errors in R II

- Using **`coeftest()`** function in the **`lmtest`** package, hypothesis tests with robust SEs

```
coeftest(het.reg) # test with normal SEs
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.103839   0.554662  0.1872   0.8516
## x           1.025921   0.094108 10.9015   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coeftest(het.reg,vcov=vcovHC(het.reg,"HC1")) # tests with robust SEs
```

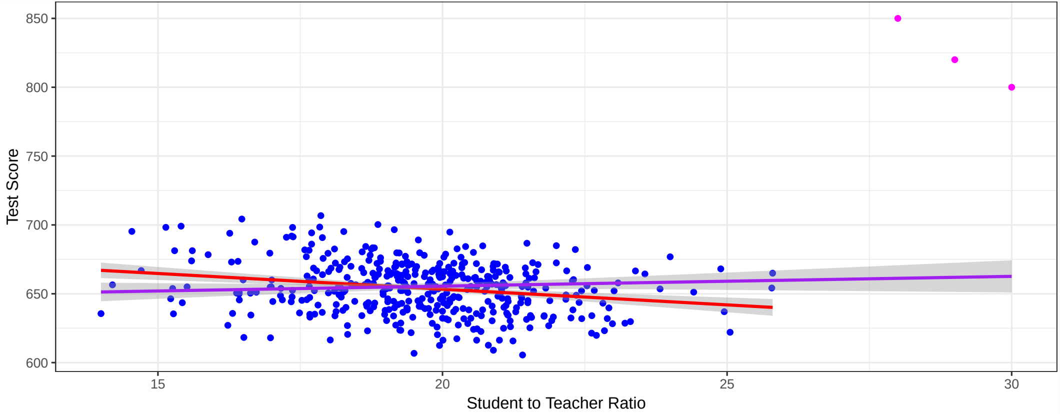
```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.10384    0.35275  0.2944   0.7686
## x            1.02592    0.10067 10.1914   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Using Robust Standard Errors in R: `stargazer` output

	y	
	Normal SEs	Robust SEs
	(1)	(2)
x	1.026***	1.026***
	(0.094)	(0.101)
Constant	0.104	0.104
	(0.555)	(0.353)
N	500	500
R^2	0.193	0.193
Residual Std. Error (df = 498)	5.977	5.977
Notes:	***Significant at the 1 percent level.	
	**Significant at the 5 percent level.	
	*Significant at the 10 percent level.	

Solvable Regression Problem #2: Outliers

Outliers Can Bias OLS



- Outliers can affect the slope (and intercept) of the line

Outliers Can Bias OLS

	testscr	
	With Outliers	Without Outliers
	(1)	(2)
str	0.708	−2.280***
	(0.566)	(0.480)
Constant	641.404***	698.933***
	(11.215)	(9.467)
N	423	420
R^2	0.004	0.051
Residual Std. Error	23.764 (df = 421)	18.581 (df = 418)

Notes: ***Significant at the 1 percent level.
**Significant at the 5 percent level.
*Significant at the 10 percent level.

- Plot the data and look!

Detecting Outliers

- Plot the data and look!
- A few methods to detect influence: ability of individual observations to affect OLS estimates

```r
library("car")

# Use Bonferonni test
outlierTest(school.outlier.reg) # will point out which obs #s seem outliers
```

```
##     rstudent unadjusted p-value Bonferonni 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
```

DFBETAS

- **dfbetas** measure how different each OLS coefficient will be without each observation

DFBETAS

- **dfbetas** measure how different each OLS coefficient will be without each observation
  - Scales the measure by standard error of OLS coefficient with observation deleted

- **dfbetas** measure how different each OLS coefficient will be without each observation
  - Scales the measure by standard error of OLS coefficient with observation deleted
  - Name means “difference in beta” from deleted observation

DFBETAS

- **dfbetas** measure how different each OLS coefficient will be without each observation
  - Scales the measure by standard error of OLS coefficient with observation deleted
  - Name means "difference in beta" from deleted observation
    - e.g. $dfbeta_i = -3$: observation i decreases coefficient by 3 standard errors

DFBETAS

- **dfbetas** measure how different each OLS coefficient will be without each observation
  - Scales the measure by standard error of OLS coefficient with observation deleted
  - Name means “difference in beta” from deleted observation
    - e.g. $dfbeta_i = -3$: observation i decreases coefficient by 3 standard errors
  - Downside: calculates this measure for *each* observation for *each* beta ($n \times k$ dfbetas)!

DFBETAS IN R

```r
dfbetas(school.outlier.reg)
```

```
##      (Intercept)           str
## 1    7.471830e-02 -6.728767e-02
## 2   -7.320670e-03  8.328531e-03
## 3   -1.346882e-02  1.119610e-02
## 4   -1.536989e-02  1.417645e-02
## 5   -1.716026e-02  1.432668e-02
## 6    7.630992e-02 -8.761930e-02
## 7   -2.033255e-02  1.010752e-02
## 8    4.622949e-02 -5.654660e-02
## 9    1.255955e-03 -1.042826e-02
## 10   3.869530e-02 -4.817225e-02
## 11   5.386822e-02 -6.283208e-02
## 12   4.334903e-02 -5.209908e-02
## 13   2.711156e-02 -3.570977e-02
## 14   3.538636e-03 -1.191752e-02
## 15  -7.150381e-02  6.384636e-02
## 16   1.283115e-02 -2.107274e-02
## 17  -1.045438e-01  9.754538e-02
## 18  -1.193732e-01  1.125672e-01
## 19   1.056295e-01 -1.142546e-01
## 20  -2.511522e-04 -7.236259e-03
## 21  -5.379359e-02  4.686663e-02
## 22   9.784139e-02 -1.060655e-01
## 23  -1.571693e-02  8.692185e-03
## 24   1.938071e-01 -2.028677e-01
```

- Often, outliers may be the result of human error (measurement, transcribing, etc)

Dealing with Outliers

- Often, outliers may be the result of human error (measurement, transcribing, etc)
- Outliers may be meaningful and accurate

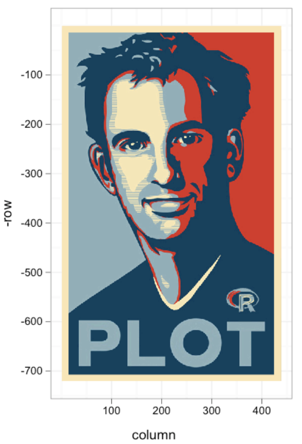
Dealing with Outliers

- Often, outliers may be the result of human error (measurement, transcribing, etc)
- Outliers may be meaningful and accurate
- In any case, compare how including/dropping outliers affects regression and always discuss outliers!

Advanced Plotting in R with `ggplot2`

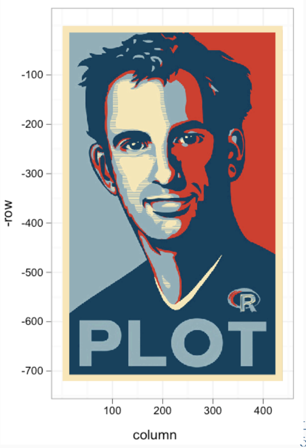
ggplot2

- `ggplot2` is the most popular package



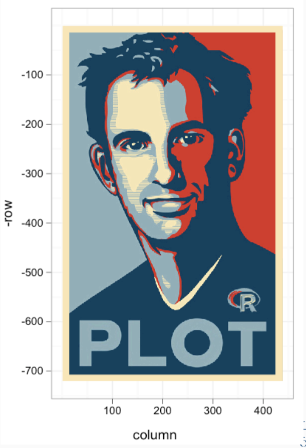
ggplot2

- `ggplot2` is the most popular package
- Created by Hadley Wickham, part of `tidyverse` (really the ”Hadleyverse”)



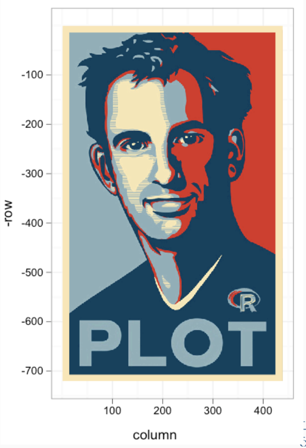
ggplot2

- `ggplot2` is the most popular package
- Created by Hadley Wickham, part of `tidyverse` (really the ”Hadleyverse”)
- Very powerful and beautiful, but requires a bit more of a learning curve



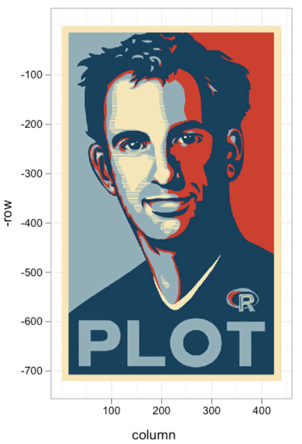
ggplot2

- `ggplot2` is the most popular package
- Created by Hadley Wickham, part of `tidyverse` (really the ”Hadleyverse”)
- Very powerful and beautiful, but requires a bit more of a learning curve
- All those ”cool graphics” you've seen in the New York Times, fivethirtyeight, the Economist, Vox, etc. used ggplot2

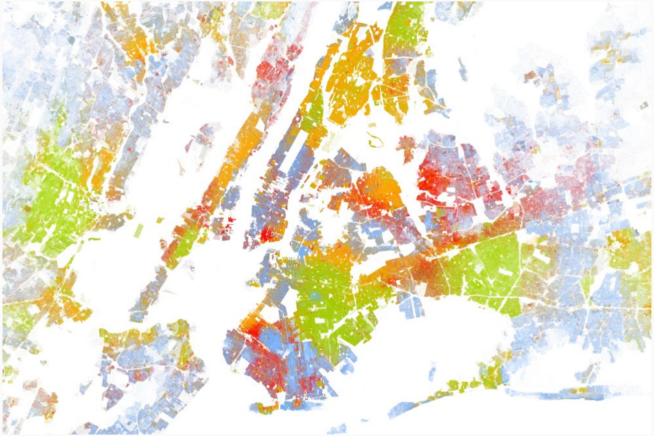


ggplot2

- `ggplot2` is the most popular package
- Created by Hadley Wickham, part of **`tidyverse`** (really the "Hadleyverse")
- Very powerful and beautiful, but requires a bit more of a learning curve
- All those "cool graphics" you've seen in the New York Times, fivethirtyeight, the Economist, Vox, etc. used ggplot2
- **gg** stands for a **grammar of graphics**



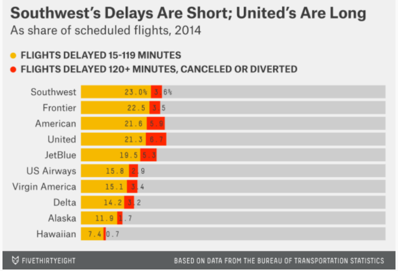
A Picture is Worth a Thousand Words



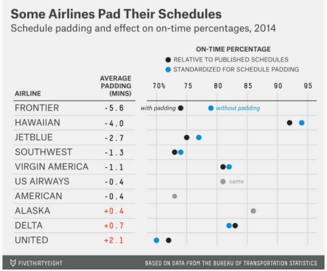
Dustin Cable's Racial Dot Map of NYC[1], The Best Map Ever Made of America's Racial Segretation; his (Python) code is open-source and available on Github

[1] 1 dot = 1 person, colors: White, African-American, Asian, Latino, All Other

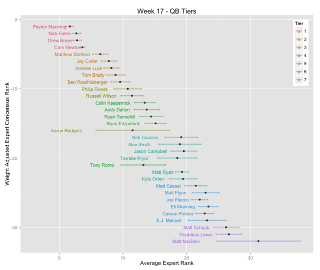
ggplot2: All Your Figure Are Belong to Us



From fivethirtyyeight



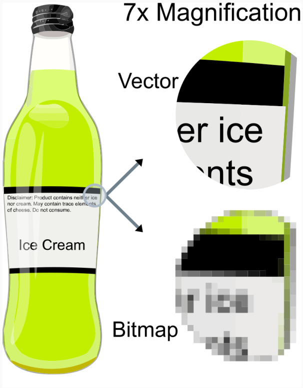
From fivethirtyyeight



From New York Times

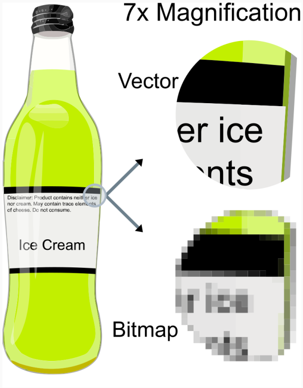
HOOD COLLEGE

- Figures produced are **vector-based** graphics (.pdf, .svg)
    - Can rescale to any size and not look ”pixely”
    - But big file size

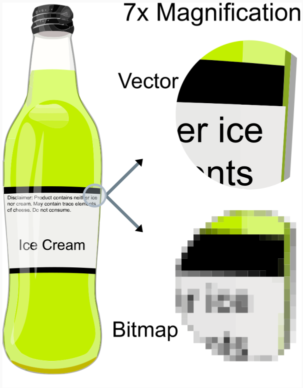


`ggplot2` FIGURES

- Figures produced are **vector-based** graphics (.pdf, .svg)
    - Can rescale to any size and not look "pixely"
    - But big file size
- Can save figures as **raster-based** graphics (.png, .jpeg, .bmp, .gif)
    - Look *awful* when blown up
    - But small file size, great for MS Office and Web



- Figures produced are **vector-based** graphics (.pdf, .svg)
    - Can rescale to any size and not look ”pixely”
    - But big file size
- Can save figures as **raster-based** graphics (.png, .jpeg, .bmp, .gif)
    - Look *awful* when blown up
    - But small file size, great for MS Office and Web
- Suggestions:
    - If printing on paper, save graphics as .pdf
    - If posting to the web, save as .png and specify size



Two Ways to Plot

1. Just the single `ggplot` command

```
ggplot(...) # make and view plot
ggplot(some.options) # remake plot with new options and view plot
```

1. Just the single `ggplot` command
    - Will view plot right after producing it

```
ggplot(...) # make and view plot
ggplot(some.options) # remake plot with new options and view plot
```

1. Just the single `ggplot` command

    - Will view plot right after producing it
    - Does not save as an object

```
ggplot(...) # make and view plot
ggplot(some.options) # remake plot with new options and view plot
```

Two Ways to Plot

1. Just the single `ggplot` command
    - Will view plot right after producing it
    - Does not save as an object
    - Need to rerun or copy/paste full command producing plot in order to modify or view it again

```r
ggplot(...) # make and view plot
ggplot(some.options) # remake plot with new options and view plot
```

Two Ways to Plot

1. Just the single `ggplot` command
    - Will view plot right after producing it
    - Does not save as an object
    - Need to rerun or copy/paste full command producing plot in order to modify or view it again
    - Can still put it in a document

```
ggplot(...) # make and view plot
ggplot(some.options) # remake plot with new options and view plot
```

2. Create an object (as usual in R)

```
plot.name<-ggplot(...) # make plot
plot.name<-plot.name+some.options # add new options to existing plot
plot.name # view plot
```

2. Create an object (as usual in R)
    - This allows you to save the plot for later (re)use

```r
plot.name<-ggplot(...) # make plot
plot.name<-plot.name+some.options # add new options to existing plot
plot.name # view plot
```

2. Create an object (as usual in R)
    - This allows you to save the plot for later (re)use
    - Also allows you to modify it

```
plot.name<-ggplot(...) # make plot
plot.name<-plot.name+some.options # add new options to existing plot
plot.name # view plot
```

2. Create an object (as usual in R)

   - This allows you to save the plot for later (re)use
   - Also allows you to modify it
   - Any time you want to view display it (i.e. for putting it in a document), just call up the plot by name

```
plot.name<-ggplot(...) # make plot
plot.name<-plot.name+some.options # add new options to existing plot
plot.name # view plot
```

```
plot.name<-ggplot(data=mydf, mapping=aes(x=xvar,y=yvar))+
  geom_something(options)+
  moreoptions...
```

- gg "grammar of graphics" implies any graphic can be built from the same components/layers:

```
plot.name<-ggplot(data=mydf, mapping=aes(x=xvar,y=yvar))+
  geom_something(options)+
  moreoptions...
```

- gg "grammar of graphics" implies any graphic can be built from the same components/layers:
  1. **Data**: base-layer describes the data used

```
plot.name<-ggplot(data=mydf, mapping=aes(x=xvar,y=yvar))+
  geom_something(options)+
  moreoptions...
```

- `gg` "grammar of graphics" implies any graphic can be built from the same components/layers:
  1. **Data**: base-layer describes the data used
     - `mydf` is the dataframe containing `xvar` and `yvar`

```
plot.name<-ggplot(data=mydf, mapping=aes(x=xvar,y=yvar))+
  geom_something(options)+
  moreoptions...
```

- `gg` "grammar of graphics" implies any graphic can be built from the same components/layers:
  1. **Data**: base-layer describes the data used
     - `mydf` is the dataframe containing `xvar` and `yvar`
     - `aes()` "aesthetics" identifies `xvar` (and if applicable `yvar`) from data to be "mapped" to a visual mark

```
plot.name<-ggplot(data=mydf, mapping=aes(x=xvar,y=yvar))+
  geom_something(options)+
  moreoptions...
```

- gg "grammar of graphics" implies any graphic can be built from the same components/layers:
  1. **Data**: base-layer describes the data used
     - `mydf` is the dataframe containing `xvar` and `yvar`
     - `aes()` "aesthetics" identifies `xvar` (and if applicable `yvar`) from data to be "mapped" to a visual mark
  2. **Geoms**: visual marks that represent data observations or models, common examples:

```
plot.name<-ggplot(data=mydf, mapping=aes(x=xvar,y=yvar))+
  geom_something(options)+
  moreoptions...
```

- gg "grammar of graphics" implies any graphic can be built from the same components/layers:
  1. **Data**: base-layer describes the data used
     - `mydf` is the dataframe containing `xvar` and `yvar`
     - `aes()` "aesthetics" identifies `xvar` (and if applicable `yvar`) from data to be "mapped" to a visual mark
  2. **Geoms**: visual marks that represent data observations or models, common examples:
     - e.g. `geom_point`, `geom_line`, `geom_bar`, `geom_histogram`, `geom_density`, `geom_boxplot`

```
plot.name<-ggplot(data=mydf, mapping=aes(x=xvar,y=yvar))+
  geom_something(options)+
  moreoptions...
```

- `gg` "grammar of graphics" implies any graphic can be built from the same components/layers:
  1. **Data**: base-layer describes the data used
     - `mydf` is the dataframe containing `xvar` and `yvar`
     - `aes()` "aesthetics" identifies `xvar` (and if applicable `yvar`) from data to be "mapped" to a visual mark
  2. **Geoms**: visual marks that represent data observations or models, common examples:
     - e.g. `geom_point`, `geom_line`, `geom_bar`, `geom_histogram`, `geom_density`, `geom_boxplot`
  3. **Coordinates**: Cartesian coordinates are default

```
plot.name<-ggplot(data=mydf, mapping=aes(x=xvar,y=yvar))+
  geom_something(options)+
  moreoptions...
```

- gg "grammar of graphics" implies any graphic can be built from the same components/layers:
  1. **Data**: base-layer describes the data used
     - `mydf` is the dataframe containing `xvar` and `yvar`
     - `aes()` "aesthetics" identifies `xvar` (and if applicable `yvar`) from data to be "mapped" to a visual mark
  2. **Geoms**: visual marks that represent data observations or models, common examples:
     - e.g. `geom_point`, `geom_line`, `geom_bar`, `geom_histogram`, `geom_density`, `geom_boxplot`
  3. **Coordinates**: Cartesian coordinates are default
     - change scales, axes, labels, etc; advanced options like maps

Example

For our example, we'll use the mpg dataset loaded with the `ggplot2` package

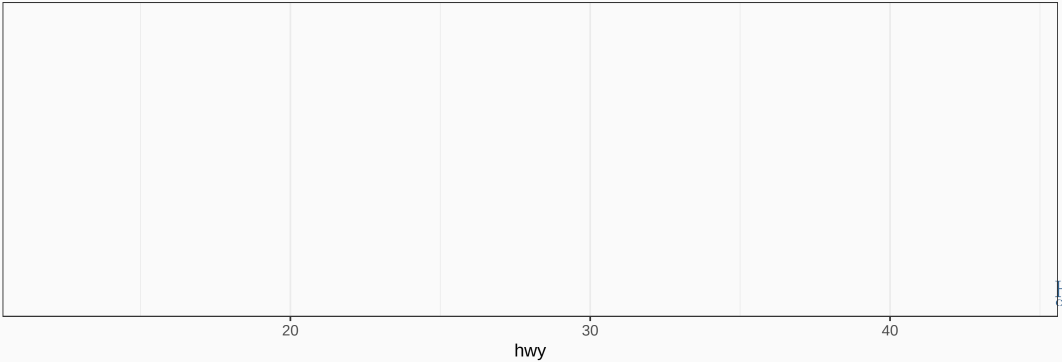
```
library("ggplot2") #load ggplot2
mpg #look at dataset
```

```
## # A tibble: 234 x 11
##    manufacturer model      displ  year   cyl trans    drv     cty   hwy fl
##    <chr>        <chr>      <dbl> <int> <int> <chr>    <chr> <int> <int> <chr>
##  1 audi         a4           1.8  1999     4 auto(l~ f        18    29 p
##  2 audi         a4           1.8  1999     4 manual~ f        21    29 p
##  3 audi         a4           2    2008     4 manual~ f        20    31 p
##  4 audi         a4           2    2008     4 auto(a~ f        21    30 p
##  5 audi         a4           2.8  1999     6 auto(l~ f        16    26 p
##  6 audi         a4           2.8  1999     6 manual~ f        18    26 p
##  7 audi         a4           3.1  2008     6 auto(a~ f        18    27 p
```

gg Histogram: Base Layer

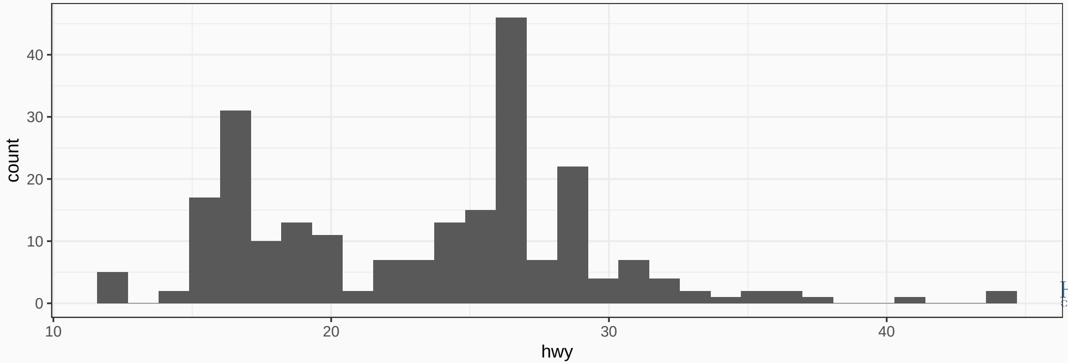
- Start with the base layer: establish the data source, define *x* variable

```
mpg.h<-ggplot(data=mpg,mapping=aes(x=hwy))
mpg.h
```



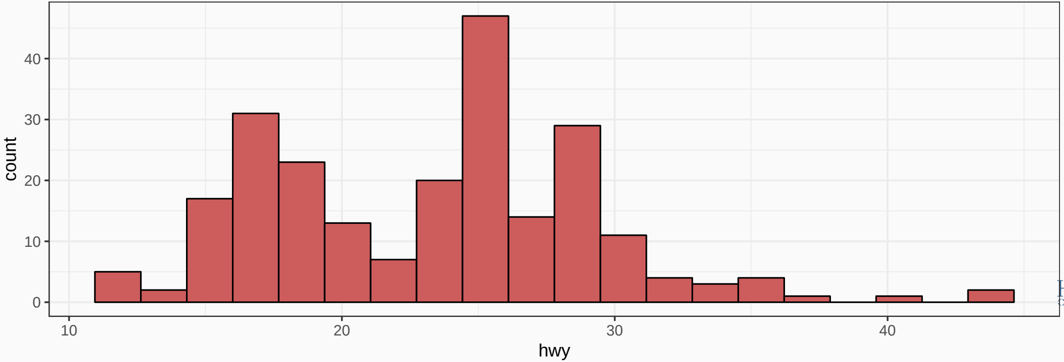
- Add a histogram layer of **hwy**

```
mpg.h1<-mpg.h+geom_histogram()
mpg.h1
```



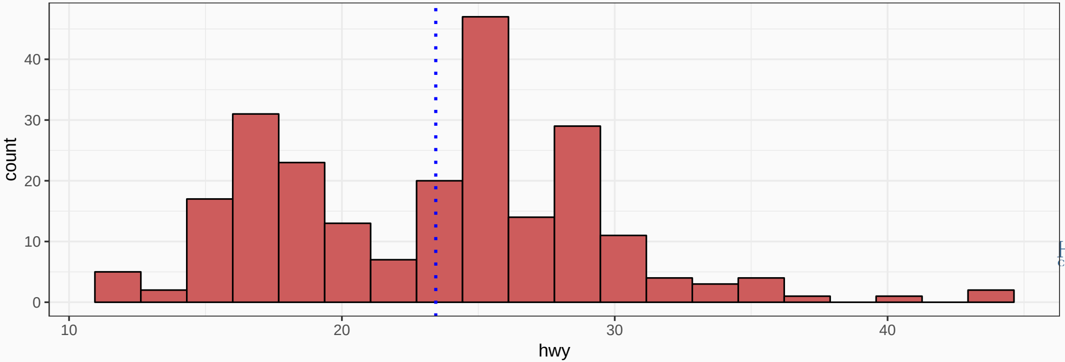
- Edit the histogram (# of bins, color, etc)

```
mpg.h2<-mpg.h+geom_histogram(bins=20, color="black",fill="indianred")
mpg.h2
```



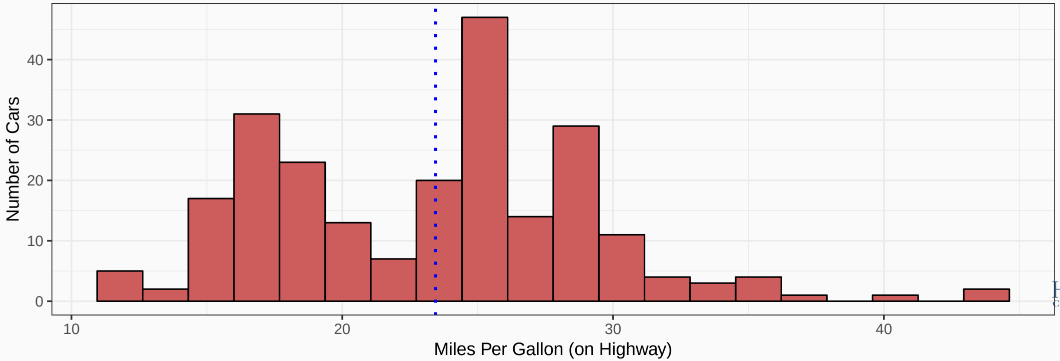
- Add a vertical line for the mean with another `geom` called `vline`

```
mpg.h2<-mpg.h2+
  geom_vline(xintercept=mean(mpg$hwy),linetype="dotted",color="blue",size=1)
mpg.h2
```



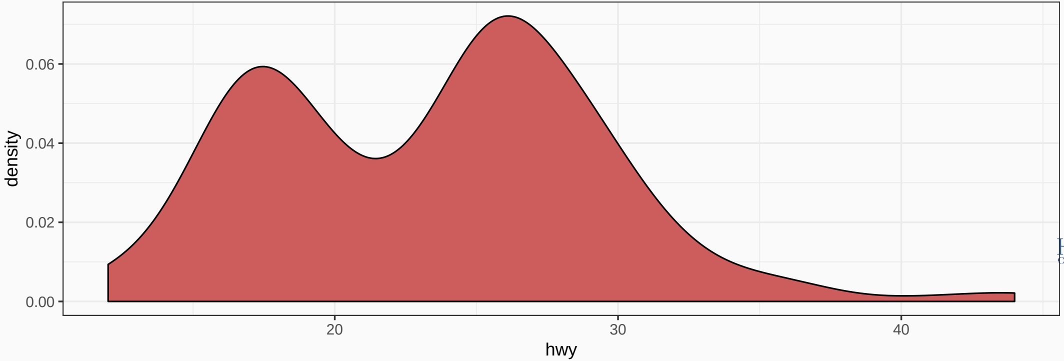
- Change the labels on the axes with `xlab()` and `ylab()`

```
mpg.h2<-mpg.h2+xlab("Miles Per Gallon (on Highway)")+ylab("Number of Cars")
mpg.h2
```



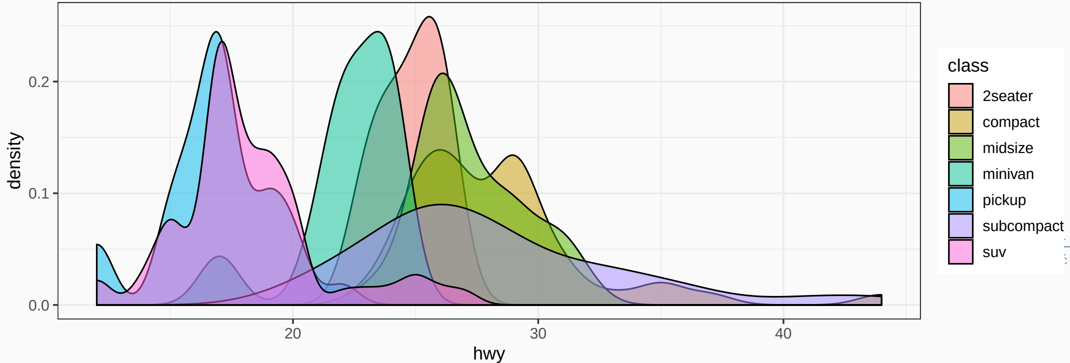
- How about a **density plot**: use `geom_density()` instead of `geom_histogram()`

```
mpg.d<-ggplot(data=mpg,aes(x=hwy))+
  geom_density(fill="indianred")
mpg.d
```



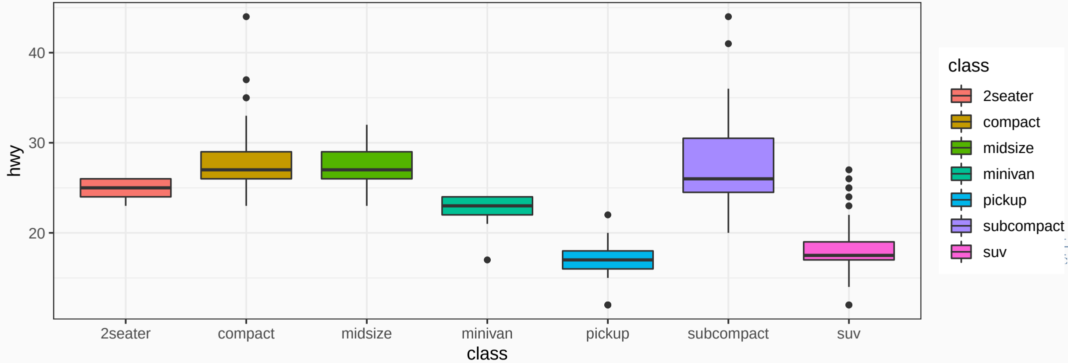
- Let's make a separate density plot for each `class`, set `aes` to `fill` by `class`

```
mpg.d<-ggplot(data=mpg,aes(x=hwy,fill=class))+
  geom_density(alpha=0.5) # alpha adds transparency
mpg.d
```



- Instead of a density plot, a **boxplot** by **class** (note now **x** is **class** and **y** is **hwy**):

```
mpg.b<-ggplot(data=mpg,aes(x=class,y=hwy,fill=class))+
  geom_boxplot()
mpg.b
```

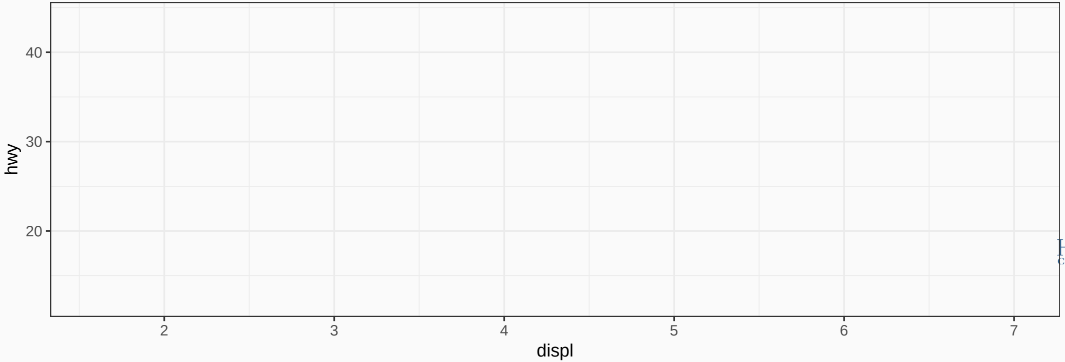


Scatterplot

- Start with the base layer: establish data source, define *x* and *y* variables

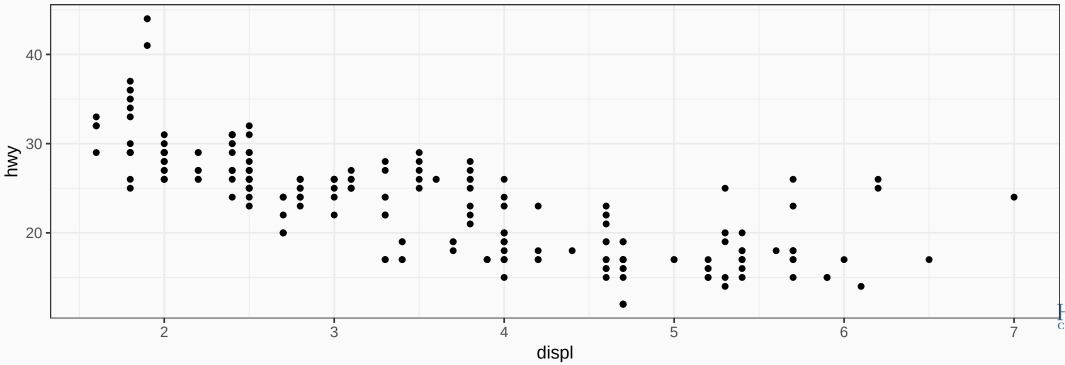
```
mpg.p<-ggplot(data=mpg,aes(x=displ, y=hwy)) #use mtcars df, let x=displ, y=hwy

mpg.p
```



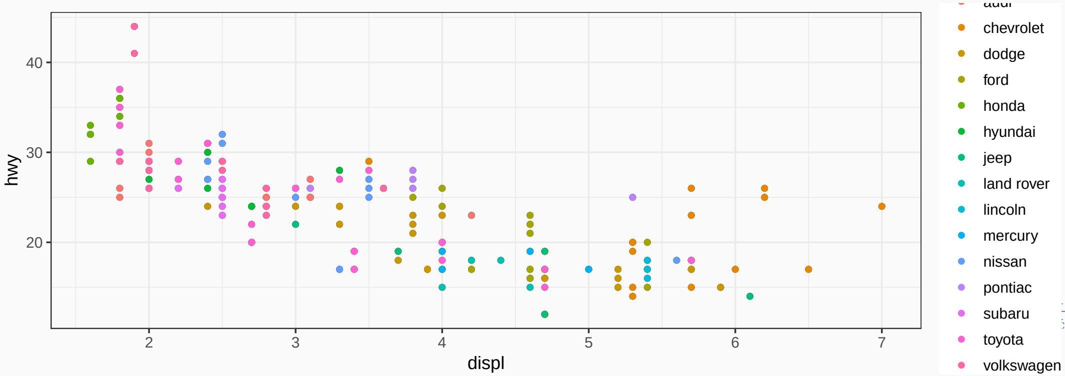
```
mpg.p<-mpg.p+geom_point() # specify observations as points on graph

mpg.p
```



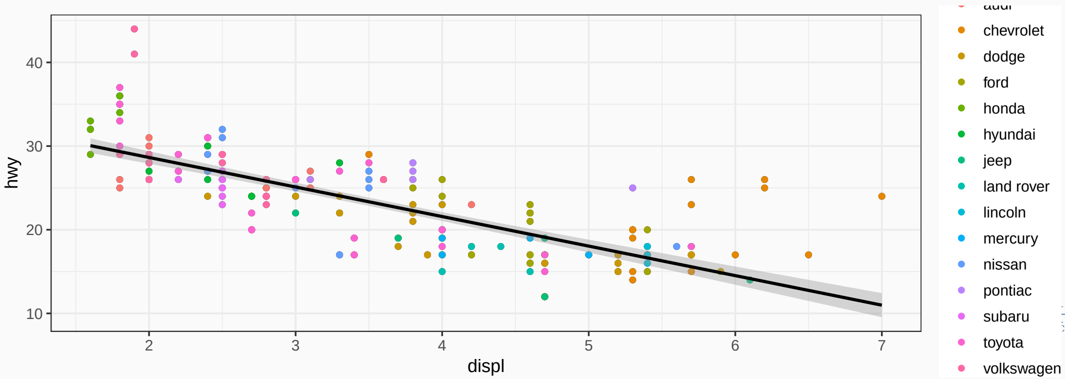
```
mpg.p<-mpg.p+geom_point(aes(color=manufacturer)) # color data points by manuf.

mpg.p
```



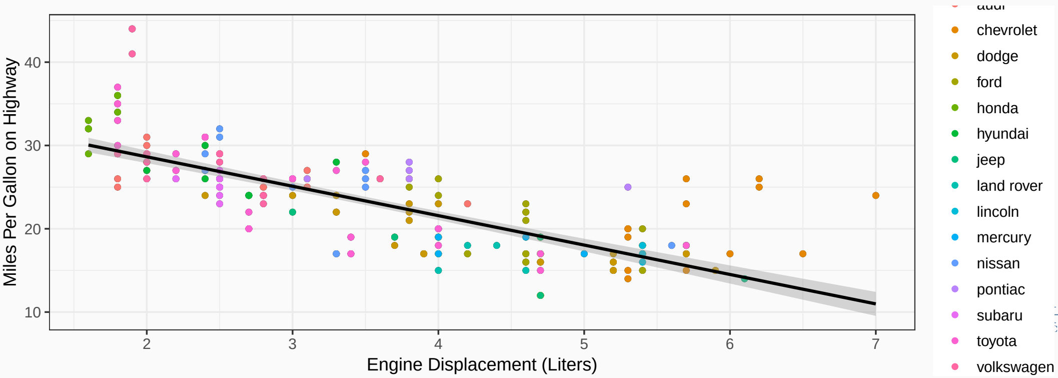
```r
mpg.p<-mpg.p+geom_smooth(method="lm", color="black") # add a black OLS line

mpg.p
```



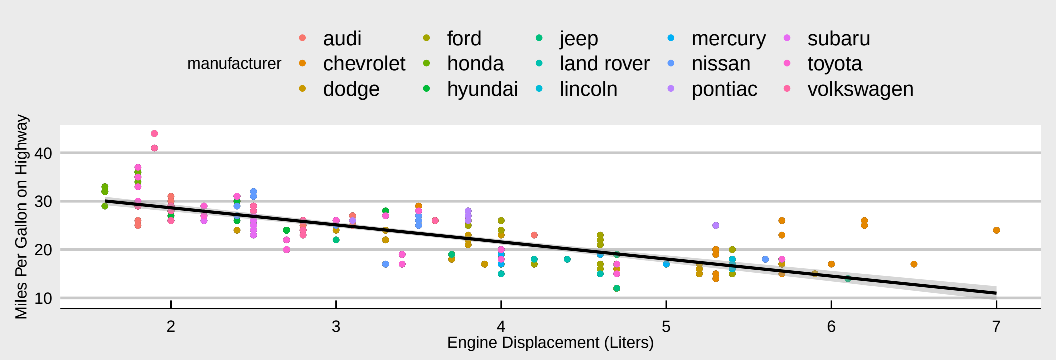
Scatterplot: Coordinate Layer

```
mpg.p<-mpg.p+xlab("Engine Displacement (Liters)")+
  ylab("Miles Per Gallon on Highway")
mpg.p
```



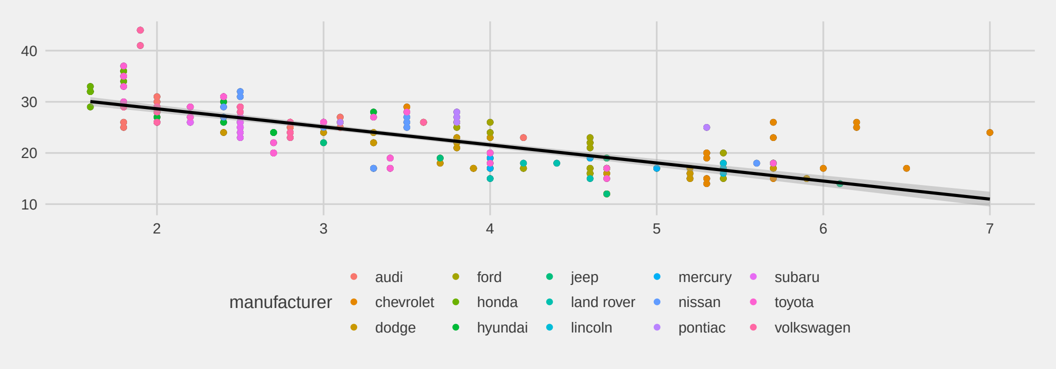
- Let's have some fun changing the theme

```
library("ggthemes") # need ggthemes package (install if first use)
mpg.p<-mpg.p+theme_economist_white() #make it look like The Economist magazine
mpg.p
```



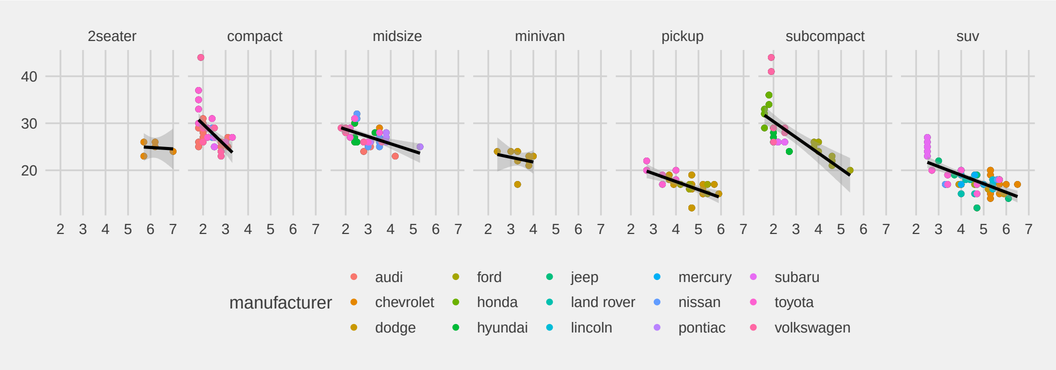
HOOD COLLEGE

```
mpg.p<-mpg.p+theme_fivethirtyeight() #make it look like fivethirtyeight
mpg.p
```

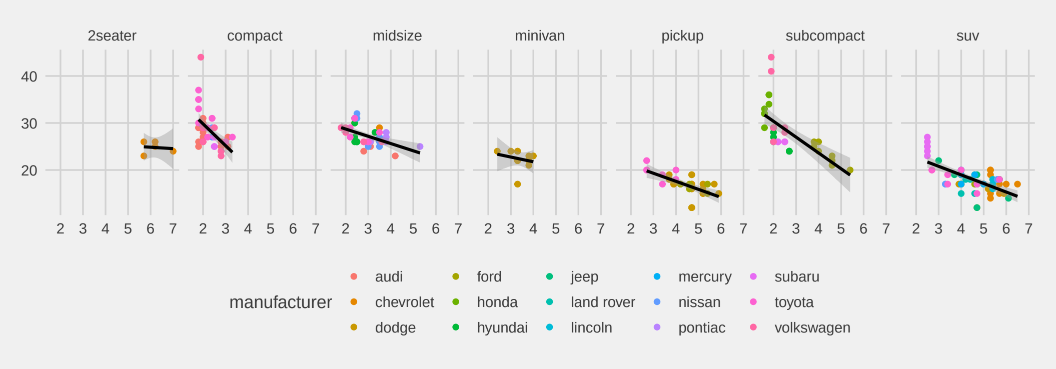


Scatterplot: Coordinate Options: Facetting

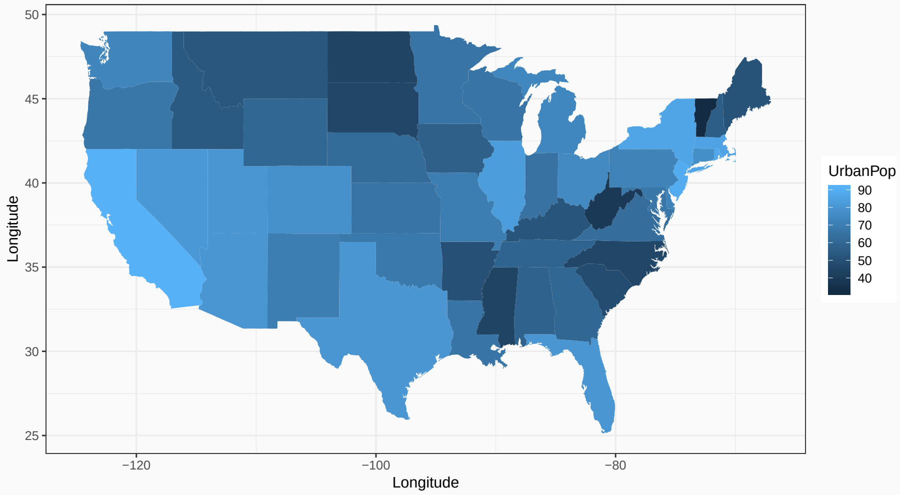
```
# make columns of separate 'facet' figures for each class of car
mpg.p<-mpg.p+facet_grid(cols = vars(class)) # make 'columns' by variable 'class'
mpg.p
```



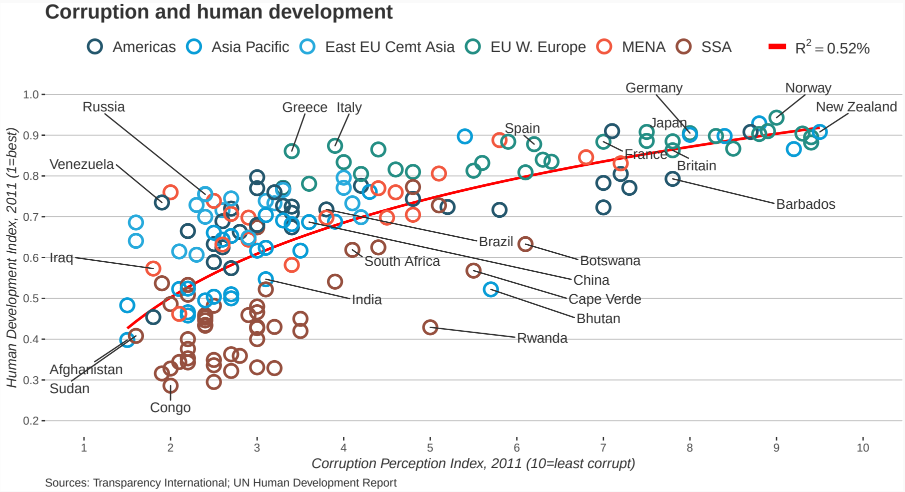
```
ggplot(data=mpg,aes(x=displ, y=hwy))+geom_point(aes(color=manufacturer))+
  geom_smooth(color="black",method="lm")+
  xlab("Engine Displacement (Liters)")+ylab("Miles Per Gallon on Highway")+
  theme_fivethirtyeight()+facet_grid(cols = vars(class))
```



Advanced Uses of `ggplot2`: Maps (See Rmd for Code)

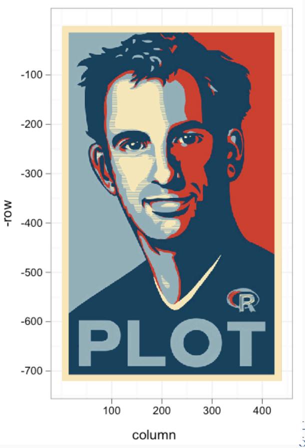


Advanced Uses of ggplot2: Maps (See Rmd for Code) II

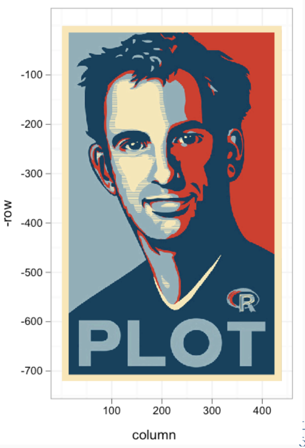


HOOD COLLEGE

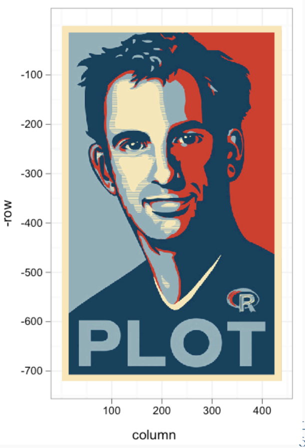
- Every single command and option has extensive documentation with examples



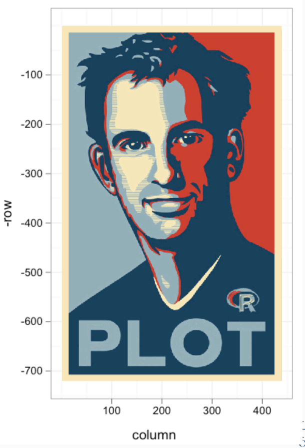
- Every single command and option has extensive documentation with examples
  - Can type `?` in front of any function to get help and examples (e.g. `?aes()`, `?geom_smooth()`)



- Every single command and option has extensive documentation with examples
    - Can type `?` in front of any function to get help and examples (e.g. `?aes()`, `?geom_smooth()`)
- Data work is a science, but it should also be **art**!



- Every single command and option has extensive documentation with examples
    - Can type `?` in front of any function to get help and examples (e.g. `?aes()`, `?geom_smooth()`)
- Data work is a science, but it should also be **art**!
- I will *not* grade you on the beauty of your graphics (but we're all secretly judging!)



- Every single command and option has extensive documentation with examples
    - Can type `?` in front of any function to get help and examples (e.g. `?aes()`, `?geom_smooth()`)
- Data work is a science, but it should also be **art**!
- I will *not* grade you on the beauty of your graphics (but we're all secretly judging!)
    - `hist()` and `plot()` are fine, you are not required to use `ggplot2` (but you really should!)