RDD, Instrumental Variable, ggplot2

Anh Le

2SLS

Let's simulate the following data generating process (DGP).

- Y is outcome, X is the independent variable, O is the omitted variable. X and O are correlated, so we have an omitted variable bias, which is an endogeneity problem.
- Z is the instrument for X (we can have multiple Z's as well).

Example: Y is `grade`, X is `attendance`, O is `time spent studying`. X and O are often correlated – hard-working students attend class more and also spend more time studying. The Z's are instruments for `attendance` (What could Z be?)

```
library(AER) # for ivreg
library(mvtnorm) # to generate multivariate normal

data <- rmvnorm(100, mean = c(0, 0, 0, 0),
                sigma = matrix(c(1, 0.5, 0.5, 0.5,
                                 0.5, 1, 0.5, 0,
                                 0.5, 0.5, 1, 0,
                                 0.5, 0, 0, 1), ncol = 4))
colnames(data) <- c("X", "Z1", "Z2", "O")


# Notice the correlation structure of the data
cor(data)
```

```
##            X        Z1         Z2          O
## X  1.0000000 0.5919883 0.48739015 0.54217986
## Z1 0.5919883 1.0000000 0.53469504 0.10718224
## Z2 0.4873902 0.5346950 1.00000000 0.02526246
## O  0.5421799 0.1071822 0.02526246 1.00000000
```

```
# Just creating new variables for ease of exposition, nothing conceptual here
X <- data[, 1] ; Z1 <- data[, 2] ; Z2 <- data[, 3] ; O <- data[, 4]

# Generate Y
Y <- 2 * X + 3 * O + rnorm(100)

# Run normal regression
summary(lm(Y ~ X))
```

```
##
## Call:
## lm(formula = Y ~ X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9969 -2.1831 -0.0651  1.8447  5.5187
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.5023     0.2804   1.791   0.0763 .
## X             3.8012     0.2828  13.441   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.784 on 98 degrees of freedom
## Multiple R-squared:  0.6483, Adjusted R-squared:  0.6447
## F-statistic: 180.6 on 1 and 98 DF,  p-value: < 2.2e-16
```

```r
# Run IV regression
# recall that true value of beta X is 2
summary(ivreg(Y ~ X | Z1 + Z2),
        diagnostics = TRUE)
```

```
##
## Call:
## ivreg(formula = Y ~ X | Z1 + Z2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -8.04866 -2.14829  0.04288  2.30347  5.46463
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.3606     0.3078   1.172    0.244
## X             2.6214     0.4906   5.343 5.95e-07 ***
##
## Diagnostic tests:
##                  df1 df2 statistic  p-value
## Weak instruments   2  97    31.182 3.49e-11 ***
## Wu-Hausman         1  97    12.503 0.000625 ***
## Sargan             1  NA     0.001 0.978596
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.021 on 98 degrees of freedom
## Multiple R-Squared: 0.5858,  Adjusted R-squared: 0.5816
## Wald test: 28.55 on 1 and 98 DF,  p-value: 5.946e-07
```

```r
# Weak instrument: F-test of the first stage. Null hypothesis = instruments are not correlated with X
# Wu-Hausman: check the endogeneity of X. Null hypothesis = X is not endogenous
# Sargan test: over-identification test, only runnable when there are more instruments than endogenous
```

```r
# Run IV regression by hand
m_1ststage <- lm(X ~ Z1 + Z2)
xhat <- predict(m_1ststage)
m_2ndstage <- lm(Y ~ xhat)
summary(m_2ndstage)
```

```
##
## Call:
## lm(formula = Y ~ xhat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -12.2057  -3.1923  -0.2877   3.8120   7.9155
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.3606     0.4484   0.804 0.423226
## xhat          2.6214     0.7149   3.667 0.000399 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.402 on 98 degrees of freedom
## Multiple R-squared:  0.1206, Adjusted R-squared:  0.1117
## F-statistic: 13.45 on 1 and 98 DF,  p-value: 0.0003991
```

Notice that running IV using package and by hand give the same coefficient estimate, but different standard error. It's because if we run the second stage like above, we don't take into account the uncertainty in estimating $\hat{x}$ (Notice how we just plug in $\hat{x}$, paying no attention to the standard error in the first stage.)

So, in real research, just use a package.

RDD

Concepts

Sherlock Holmes How often have I said to you that when you have eliminated the impossible, whatever remains, however improbable, must be the truth?

1. Prove that there is a difference in outcome at the discontinuity

Regression with a subset of observations near the cut-off. What choices do we have to make?

- How flexible the regression line is (i.e. the polynomial order)
- How large is the bandwidth?

2. Prove that this difference cannot be caused by anything else besides the discontinuity

The key issue that invalidates RDD is if people can bargain their ways into either side of the cut-off. If yes, the difference in outcome may be explained by this (unobserved) ability to bargain. To some extent, we can control for this ability to bargain (e.g. controling for the person's wealth or connection). However, if people choose one side of the cut-off because they have private information about why they would benefit more on that side, then we fundamentally cannot control for this *private* information.

How to check?

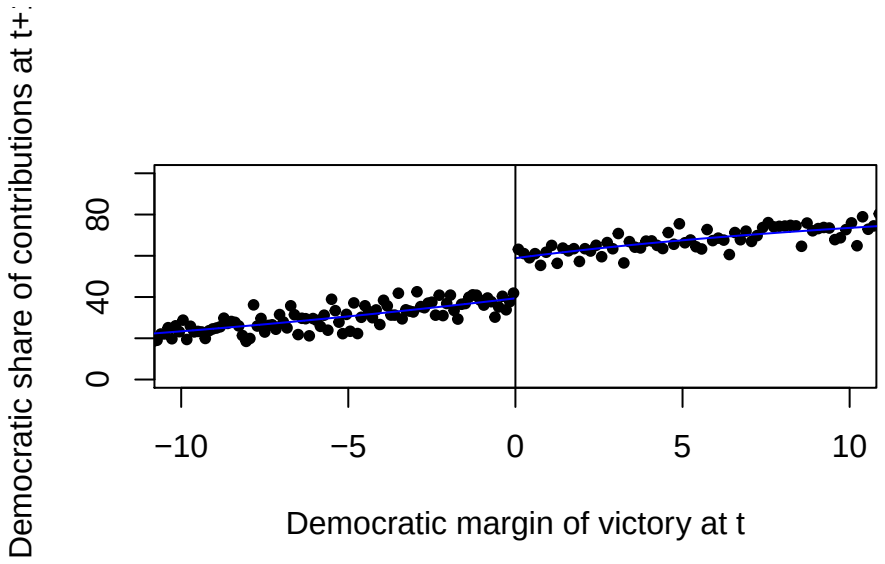
- Density test: are there the same number of obs on both sides of the cut-off? If there is a difference, it must mean that people see one side is more beneficial, and they force their way into it.
- Placebo test / Balance check: are obs on both sides of the cut-off the same? If not, it must mean that people can force their way into one side using, say, their wealth and connections.

Implementation

Taken from this excellent guide on best RDD practices (Skovron & Titiunik). Our RDD is close election in the US. The forcing variable is Democratic vote share advantage (i.e. the discontinuity happens at 0, when Dem and Rep got the same vote share).

1. We estimate the treatment effect

```r
library(dplyr) ; library(ggplot2) ; library(foreign)
library(rdrobust)

# Download these scripts from https://sites.google.com/site/rdpackages/rddensity/R
# Documentation at https://sites.google.com/site/rdpackages/rddensity
# Hopefully they put these scripts into a package soon
source("rddensity_fun.R")
source("rdbwdensity.R")
source("rddensity.R")

dat = read.dta('fouirnaies_hall_financial_incumbency_advantage.dta')
dat = filter(dat, statelevel==1) # For this replication, only use state level legislature

# Outcome variable
rdplot(dat$dv_money, dat$rv, x.lim = c(-10,10),
       x.lab="Democratic margin of victory at t",
       y.lab="Democratic share of contributions at t+1", title = "")
```




```r
# RDD estimate
rdrobust(dat$dv_money,dat$rv,all=TRUE)
```

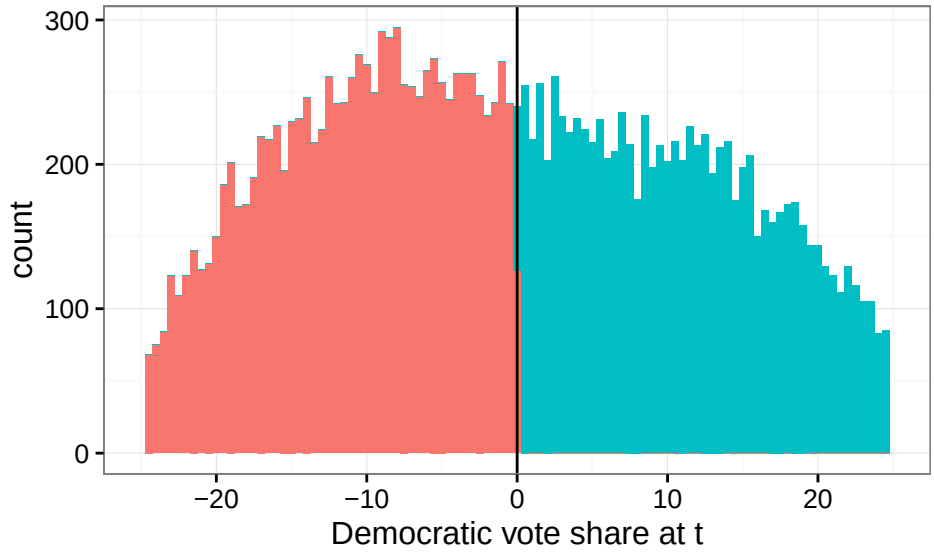
```
## Call:
## rdrobust(y = dat$dv_money, x = dat$rv, all = TRUE)
##
## Summary:
##
## Number of Obs 27203
## BW Type       mserd
## Kernel Type   Triangular
## VCE Type      NN
##
##                     Left    Right
## Number of Obs       13292   13911
## Eff. Number of Obs 3997    3436
## Order Loc Poly (p) 1       1
## Order Bias (q)     2       2
```

```
## BW Loc Poly (h)    9.2748  9.2748
## BW Bias (b)        19.0047 19.0047
## rho (h/b)          0.4880  0.4880
##
## Estimates:
##                Coef    Std. Err. z       P>|z|  CI Lower CI Upper
## Conventional   20.5317 1.2842    15.9879 0.0000 18.0147  23.0487
## Bias-Corrected 19.9707 1.2842    15.5511 0.0000 17.4537  22.4877
## Robust         19.9707 1.4229    14.0356 0.0000 17.1820  22.7595
```

2. We make sure that the treatment effect is truly due to the discontinuity

```
# Density test (graphical)

ggplot(dat,aes(x=rv, fill =factor(dat$rv>0)))+
  geom_histogram(binwidth=0.5) + xlim(-25,25) +
  geom_vline(xintercept = 0) + xlab("Democratic vote share at t") +
  scale_colour_manual(values = c("red","blue")) + theme_bw() +
  theme(legend.position='none')
```

```
## Warning: Removed 12647 rows containing non-finite values (stat_bin).
```

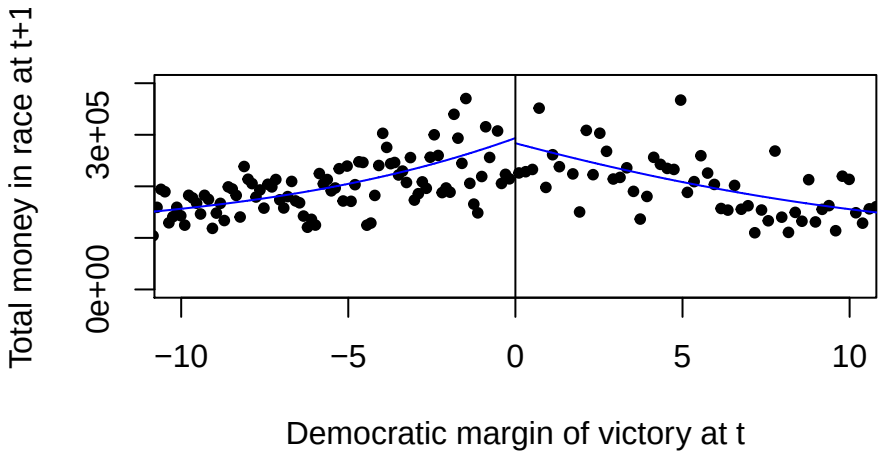



```
# Density test
rddensity(X = dat$rv, vce="jackknife")
```

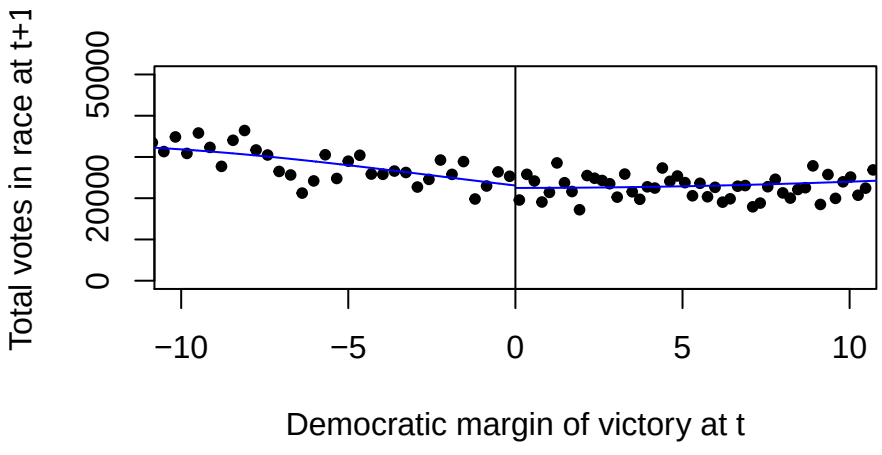
```
##
## RD Manipulation Test using local polynomial density estimation.
##
## Number of obs =               32670
## Model =                       unrestricted
## Kernel =                      triangular
## BW method =                   comb
## VCE method =                  jackknife
##
## Cutoff c = 0                  Left of c           Right of c
## Number of obs                 16281               16389
## Eff. Number of obs            2770                3176
## Min Running var.              -50                 0.006
```

```
## Max Running var.              -0.005              50
## Order loc. poly. (p)          2                   2
## Order BC (q)                  3                   3
## Bandwidths (hl,hr)            estimated           estimated
## Bandwidth values              5.458               6.935
##
## Method                        T                   P > |T|
## Robust Bias-Corrected         -0.9165             0.3594
```

```
# Placebo test (graphical)
rdplot(dat$total_race_money,dat$rv,x.lim = c(-10,10),y.lim = c(0,400000),
       x.lab="Democratic margin of victory at t",
       y.lab="Total money in race at t+1", title = "")
```




```
rdplot(dat$total_votes,dat$rv,x.lim = c(-10,10),y.lim = c(0,50000),
       x.lab="Democratic margin of victory at t",
       y.lab="Total votes in race at t+1", title = "")
```




```
# Placebo test
rdrobust(dat$total_race_money,dat$rv,all=TRUE)
```

```
## Call:
## rdrobust(y = dat$total_race_money, x = dat$rv, all = TRUE)
##
## Summary:
##
```

```
## Number of Obs 32670
## BW Type       mserd
## Kernel Type   Triangular
## VCE Type      NN
##
##                     Left    Right
## Number of Obs       16281   16389
## Eff. Number of Obs 5888    4982
## Order Loc Poly (p) 1       1
## Order Bias (q)     2       2
## BW Loc Poly (h)    11.2761 11.2761
## BW Bias (b)        17.4065 17.4065
## rho (h/b)          0.6478  0.6478
##
## Estimates:
##                 Coef        Std. Err.  z       P>|z|  CI Lower
## Conventional   -7708.5268  22876.1407 -0.3370 0.7361 -52544.9388
## Bias-Corrected -14621.7935 22876.1407 -0.6392 0.5227 -59458.2055
## Robust         -14621.7935 27476.5156 -0.5322 0.5946 -68474.7746
##                 CI Upper
## Conventional   37127.8851
## Bias-Corrected 30214.6184
## Robust         39231.1875
```

```
rdrobust(dat$total_votes,dat$rv,all=TRUE)
```

```
## Call:
## rdrobust(y = dat$total_votes, x = dat$rv, all = TRUE)
##
## Summary:
##
## Number of Obs 32670
## BW Type       mserd
## Kernel Type   Triangular
## VCE Type      NN
##
##                     Left    Right
## Number of Obs       16281   16389
## Eff. Number of Obs 3094    2791
## Order Loc Poly (p) 1       1
## Order Bias (q)     2       2
## BW Loc Poly (h)    6.0546  6.0546
## BW Bias (b)        10.4511 10.4511
## rho (h/b)          0.5793  0.5793
##
## Estimates:
##                 Coef       Std. Err. z       P>|z|  CI Lower   CI Upper
## Conventional   -1623.9334 1651.0538 -0.9836 0.3253 -4859.9395 1612.0727
## Bias-Corrected -2159.0323 1651.0538 -1.3077 0.1910 -5395.0384 1076.9738
## Robust         -2159.0323 1917.9505 -1.1257 0.2603 -5918.1462 1600.0816
```

Using IV in fuzzy RDD

Recall the Hidalgo & Nichter's paper. If a district has more than 80% of population as the electorate, it *may* be audited. What's the forcing variable and the treatment here?

ggplot2 (from last year)

Introduction

```
f_install_and_load <- function(packagename) {
  if (!packagename %in% rownames(installed.packages())) {
    install.packages(packagename)
  }
  library(packagename, character.only = TRUE)
}
packs <- c("ggplot2", "gridExtra")
lapply(packs, f_install_and_load)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## [[1]]
##  [1] "rdrobust"  "foreign"   "ggplot2"   "dplyr"     "mvtnorm"
##  [6] "AER"       "survival"  "sandwich"  "lmtest"    "zoo"
## [11] "car"       "stats"     "graphics"  "grDevices" "utils"
## [16] "datasets"  "methods"   "base"
##
## [[2]]
##  [1] "gridExtra" "rdrobust"  "foreign"   "ggplot2"   "dplyr"
##  [6] "mvtnorm"   "AER"       "survival"  "sandwich"  "lmtest"
## [11] "zoo"       "car"       "stats"     "graphics"  "grDevices"
## [16] "utils"     "datasets"  "methods"   "base"
```

This tutorial aims to teach you not only about the `ggplot2` syntax but also a new way of thinking about graphics. What's revolutionary about `ggplot2` is really this "Grammar of Graphics" thinking (i.e. what `gg-` stands for).

Deconstructing a Scatterplot (learning about Aesthetics)

Below is a simple scatterplot. How exactly is data represented in a scatterplot? Each observation is represented by a point, whose x- and y- coordinate represent the values of the two variables. Along with the coordinates, each point can also have size, color, and shape.

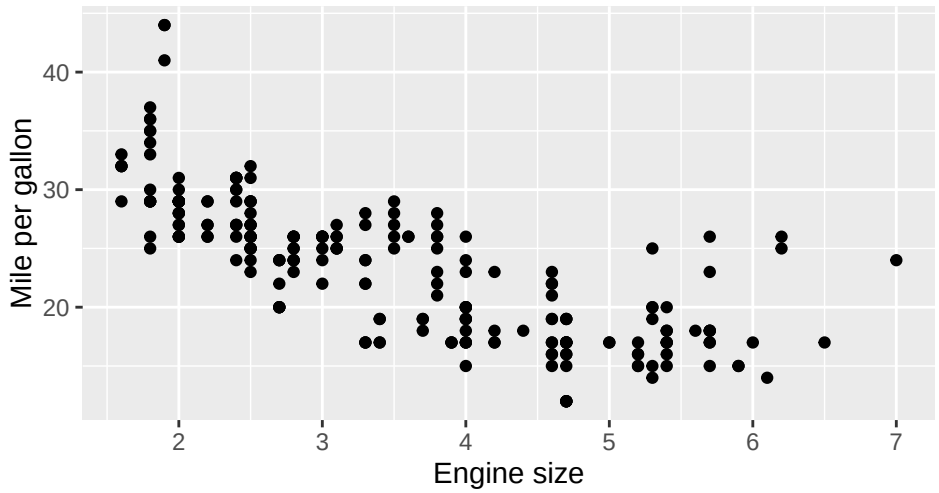
These attribute are called **aesthetics**, and are the properties that can be perceived on the graphic. Each aesthetic can be mapped to a variable, or set to a constant value.

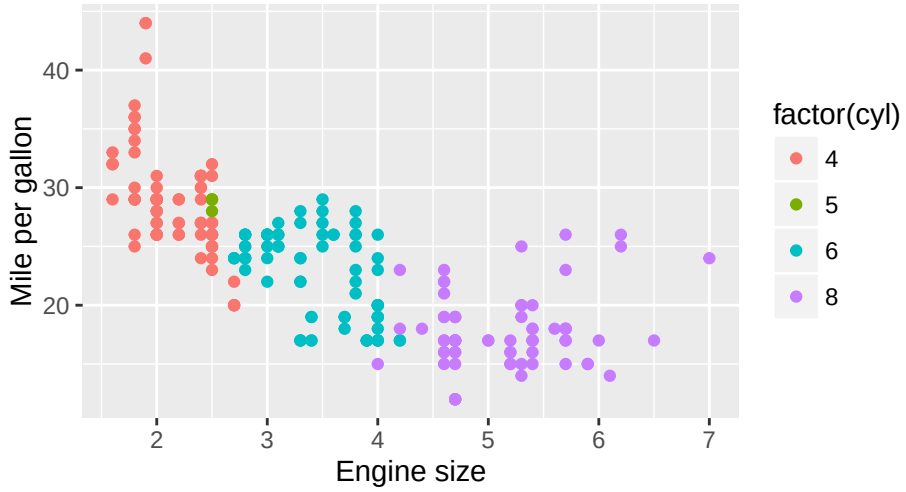
Q (conceptual): Without looking at the code, in this plot, what are the aesthetics, and what variables are they mapped to?

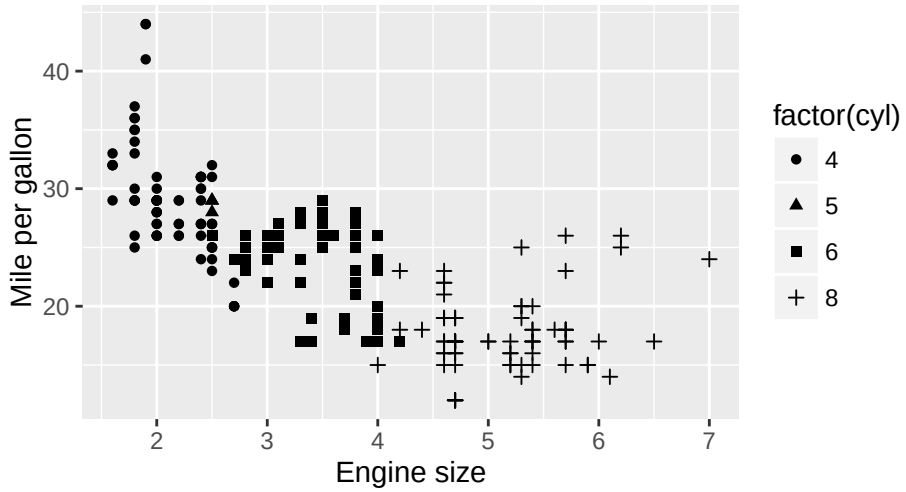
Q (ggplot syntax): Now look at the code. How did I specify the mapping?

Q (comparison with base R): How would you create the color plot in base R? Do you see how `ggplot2` is less ad-hoc and more expressive?

```
data("mpg")

p1 <-  ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  labs(y = "Mile per gallon", x = "Engine size")

p2 <- ggplot(data = mpg, aes(x = displ, y = hwy, color = factor(cyl))) +
  geom_point() +
  labs(y = "Mile per gallon", x = "Engine size")

p3 <- ggplot(data = mpg, aes(x = displ, y = hwy, shape = factor(cyl))) +
  geom_point() +
  labs(y = "Mile per gallon", x = "Engine size")

grid.arrange(p1, p2, p3, nrow = 3)
```





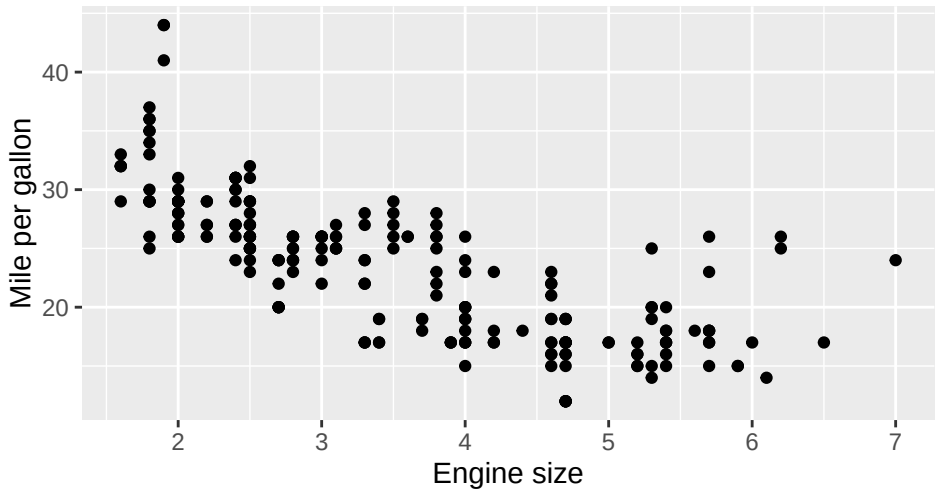


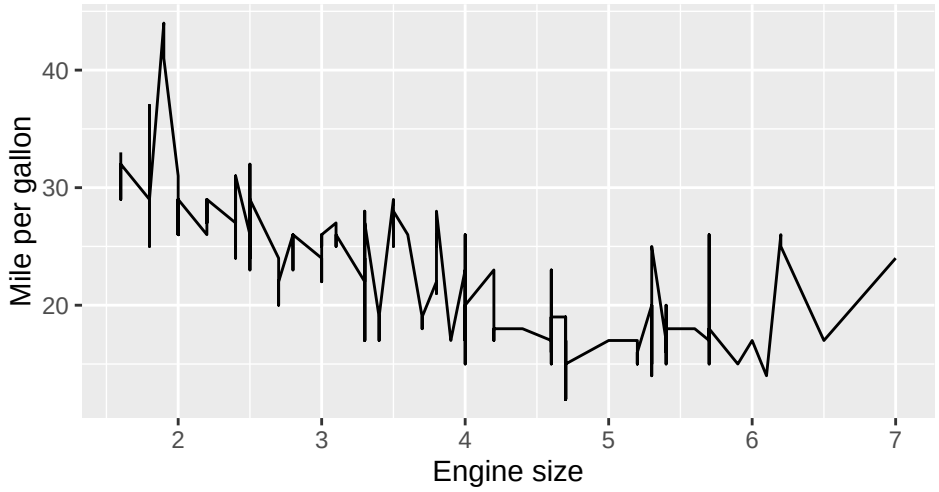
Note that the **aesthetics** is all we need to map the data to the graphics. In the plot above, I use **points** as the **geometric** form (hence `geom_point`), but it doesn't have to be. I can also use other **geometric** form, such as **line**, or **bar** (`geom_line`, `geom_bar`). Notice how the geomtric form changes, but the fundamental mapping of data to graphics does not change.

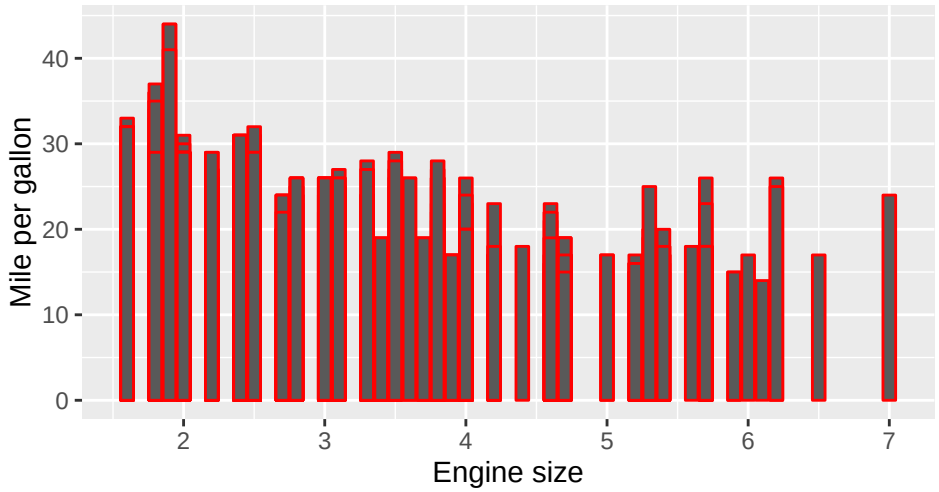
This is what we means by "grammar of graphics". The line plot and bar plot are grammatically correct, even

though it makes little sense.

```
p_point <- ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  labs(y = "Mile per gallon", x = "Engine size")

p_line <- ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_line() +
  labs(y = "Mile per gallon", x = "Engine size")

p_bar <- ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_bar(stat = 'identity', position = "identity", color = 'red') +
  labs(y = "Mile per gallon", x = "Engine size")

grid.arrange(p_point, p_line, p_bar, nrow=3)
```



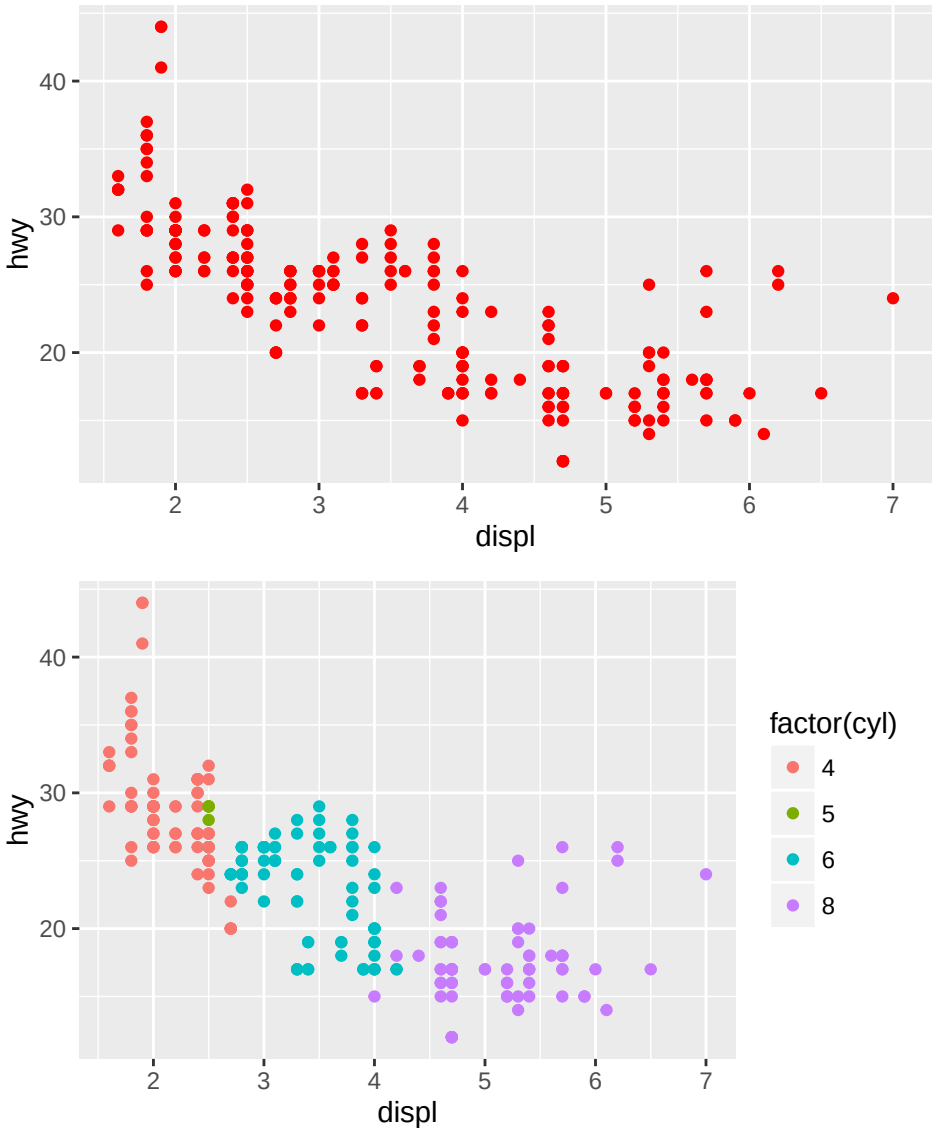




Setting aesthetics to a constant

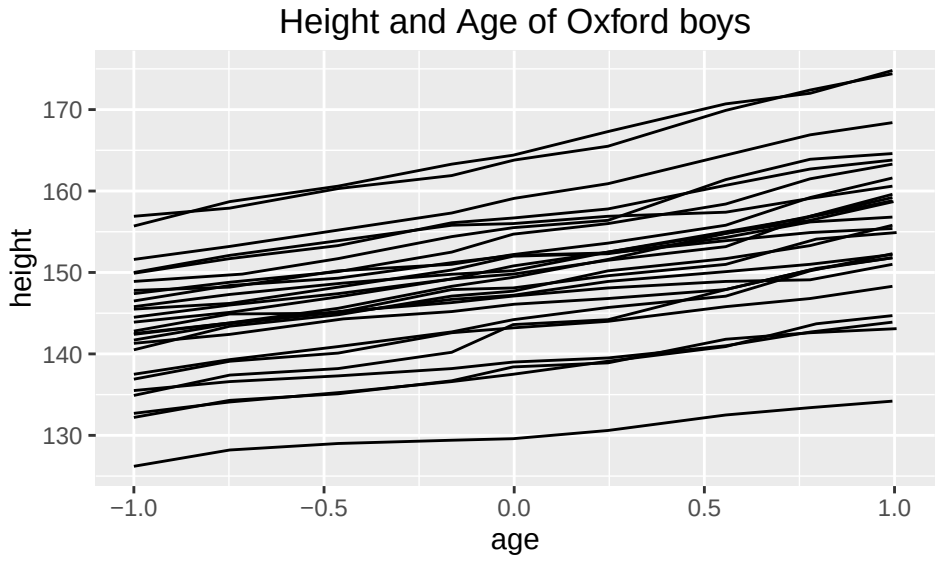
Sometimes I want to simply use a different color for my geoms, which is different from mapping the color to a variable. See:

```
p1 <- ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point(color = 'red')

p2 <- ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = factor(cyl)))

grid.arrange(p1, p2, nrow = 2)
```



Grouping

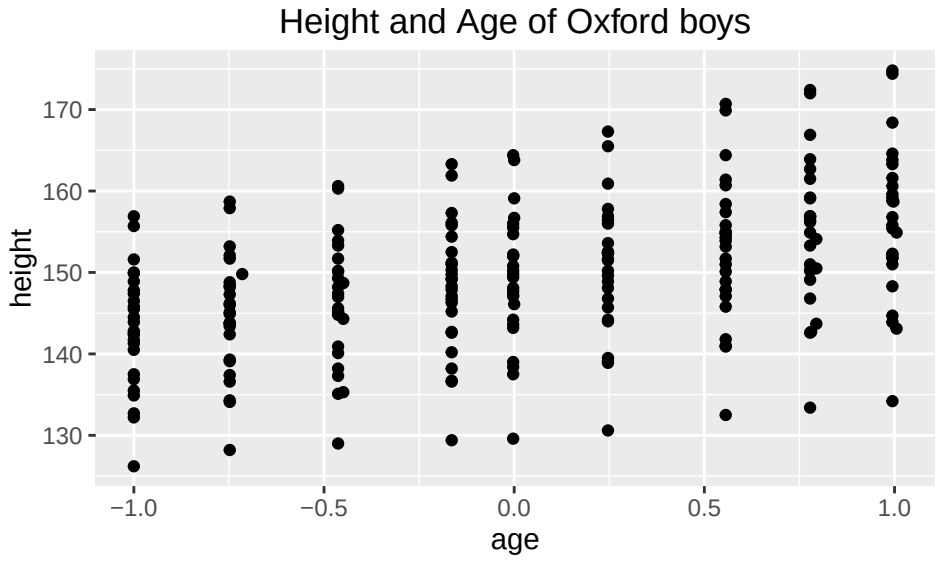
Q: Without looking at the code, what **aes** are specified in this plot?



The code that produces the line plot is below (notice the **group** aesthetics). What would be consequence of not specifying this aesthetics?

```
ggplot(Oxboys, aes(age, height, group = Subject)) +
  geom_line() +
  labs(title = "Height and Age of Oxford boys")
```

```
ggplot(Oxboys, aes(age, height)) +
  geom_line() +
  labs(title = "Height and Age of Oxford boys")
```


Height and Age of Oxford boys
height
170
160
150
140
130
−1.0
−0.5
0.0
0.5
1.0
age


Height and Age of Oxford boys
height
170
160
150
140
130
−1.0
−0.5
0.0
0.5
1.0
age


Height and Age of Oxford boys
height
170
160
150
140
130
−1.0
−0.5
0.0
0.5
1.0
age

More about geoms

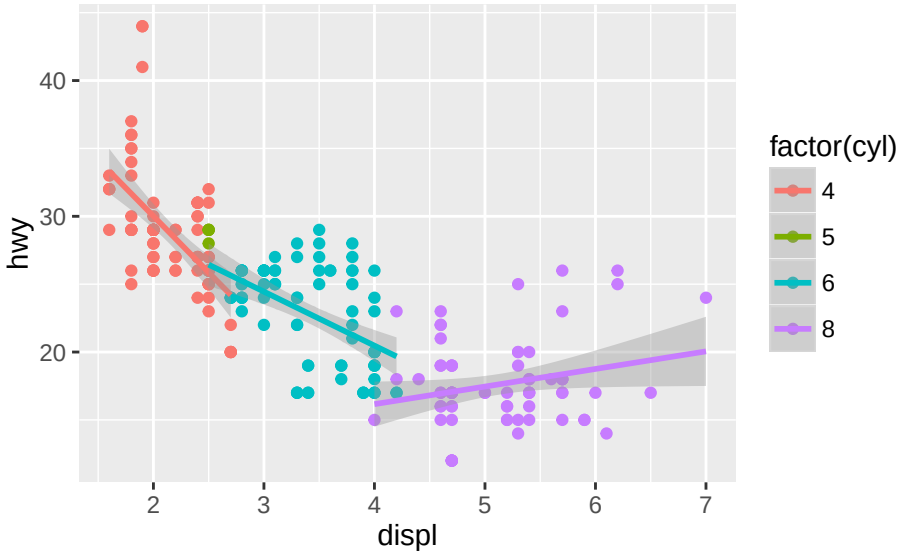
There are many **geom** built into ggplot2 see this doc. With this wide variety of selection, you can build all the common plot types.

Q (common plot): Think about how different **geom** can be useful:

- `geom_text` example
- `geom_vline` example
- `geom_errorbar` example
- `geom_polygon` example

But the magic is really in building any custom plot that you want, mixing and matching different **geoms**.

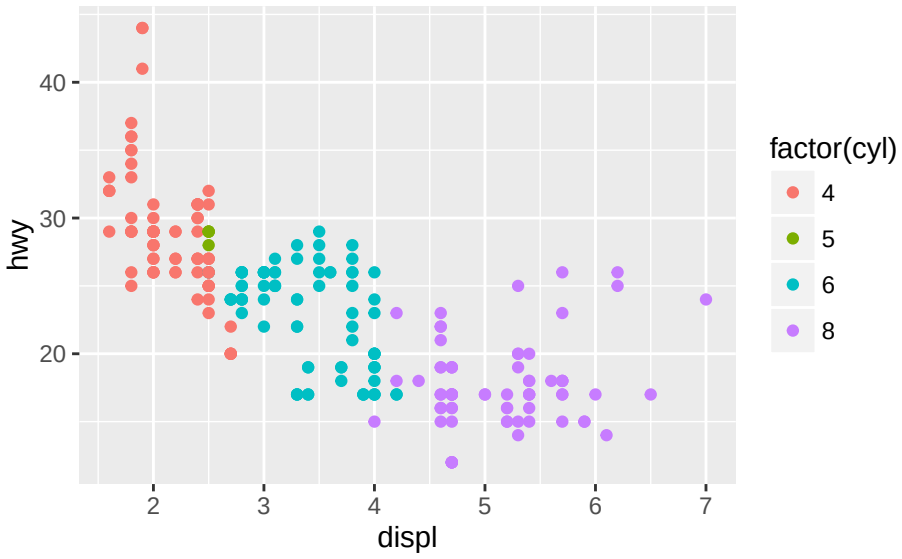
```
ggplot(data = mpg, aes(displ, hwy, color = factor(cyl))) +
  geom_point() +
  geom_smooth(data= subset(mpg, cyl != 5), method="lm")
```



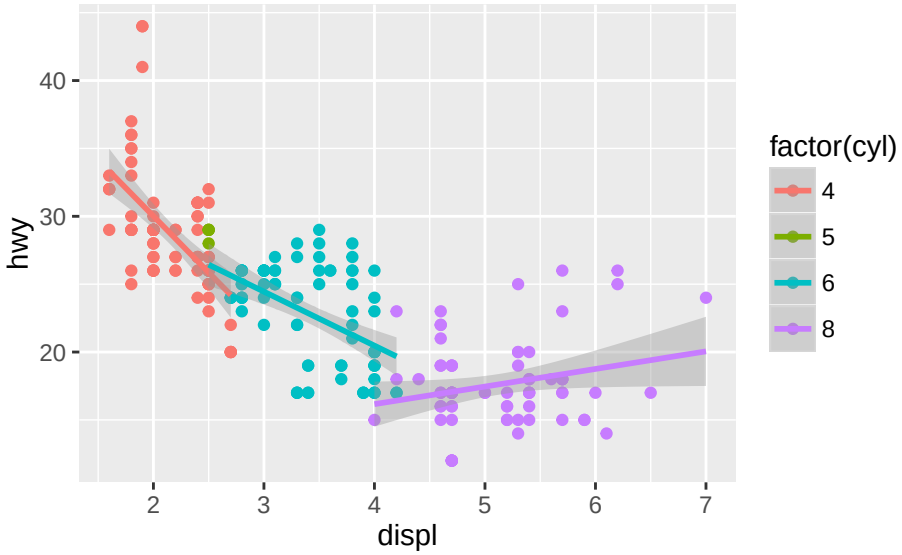
Building a plot layer by layer

As you may have noticed, the last plot has two geoms. In fact, we can add as many geoms as we want. This is thought of as building the plot layer by layer.

```
# First I add geom_point as the first layer
ggplot(data = mpg, aes(displ, hwy, color = factor(cyl))) +
  geom_point()
```




```
# Then I add geom_smooth as the second layer
ggplot(data = mpg, aes(displ, hwy, color = factor(cyl))) +
  geom_point() +
  geom_smooth(data= subset(mpg, cyl != 5), method="lm")
```



A rather extreme example of layering:

Overriding default aesthetics mapping

Now that we know how to build the plot layer by layer, sometimes the situation demands that we have different aesthetics mapping for each layer.

For example, consider the Oxford boy dataset. This is panel dataset, and I want to build 2 layers: a boxplot layer for each occasion (i.e. variation across boys at each time point), and a line layer for each boy (i.e. variation over time of each boy).

```
ggplot(data = Oxboys, aes(Occasion, height)) +
  geom_boxplot()
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour', which will replace the existing scale.
```

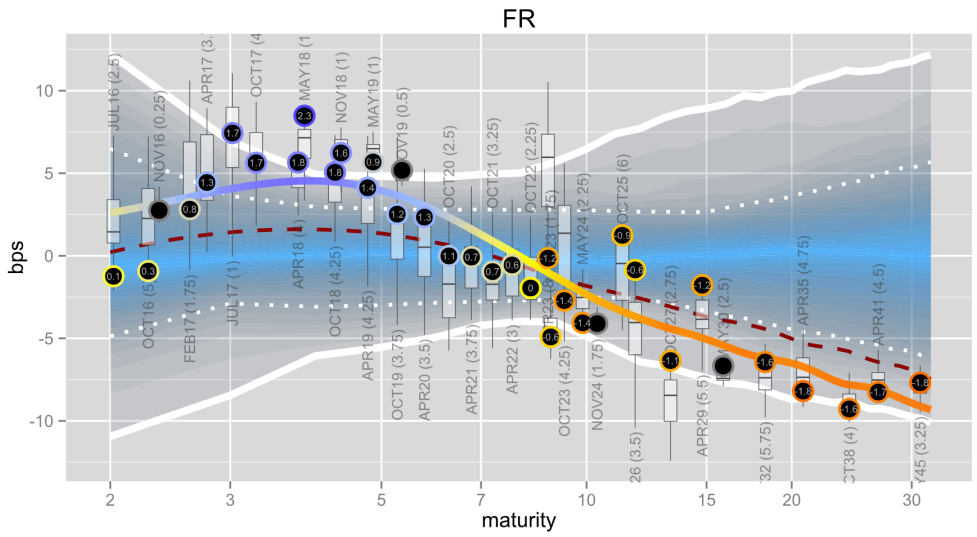
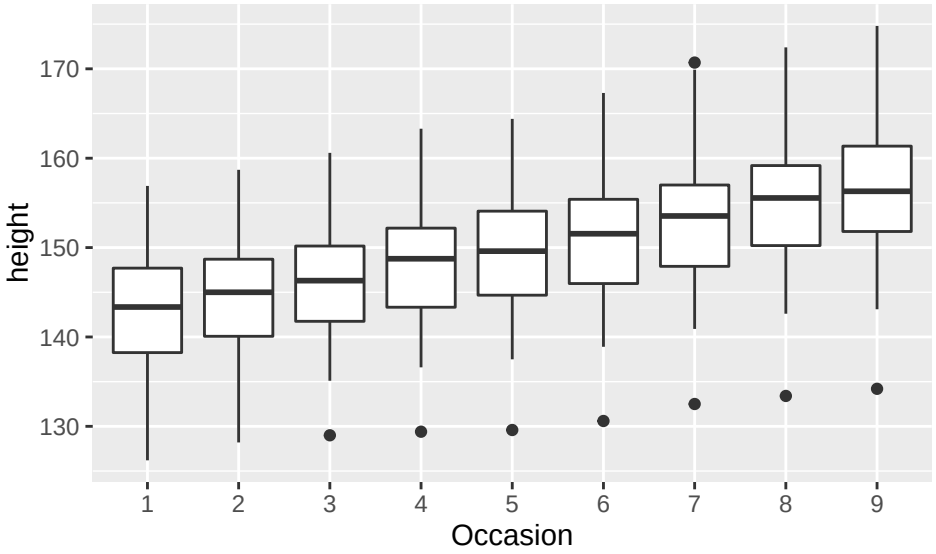
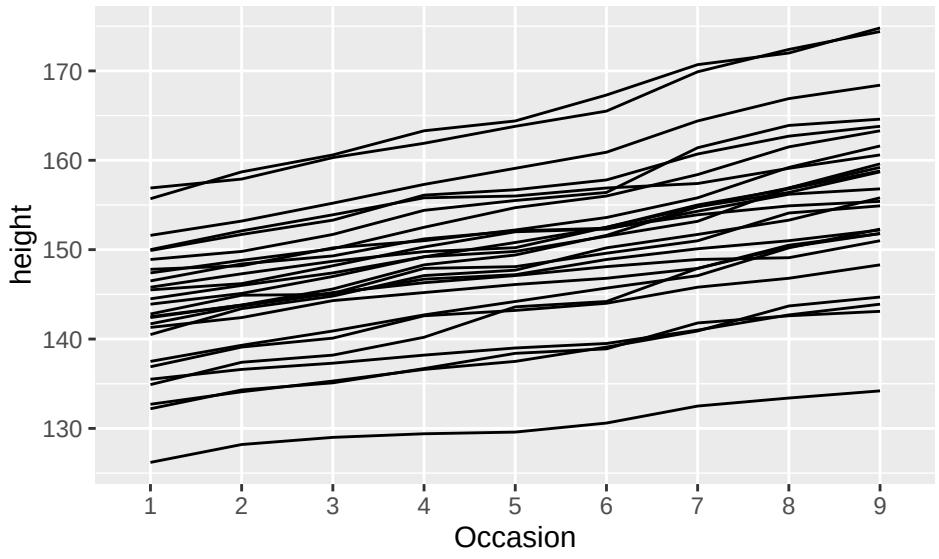


Figure 1: Here's a rather extreme example of layering.




```
ggplot(data = Oxboys, aes(Occasion, height, group = Subject)) +
  geom_line()
```



I can specify an aesthetic mapping specific to each layer as follows

```
ggplot(data = Oxboys, mapping = aes(Occasion, height)) +
  geom_boxplot() +
  geom_line(aes(group = Subject), color = 'blue')
```