Econometrics

Lecture 4

Nathaniel Higgins

ERS and JHU

Outline
Plan for the lecture

- Jump right in to where we left off
- Talk about multiple regression analysis
- Exercise to develop intuition

- A quick sidebar on linearity
- We frequently say that the function

$$y = \beta_0 + \beta_1 x + u$$

is linear. What do we mean?

- A quick sidebar on linearity
- We frequently say that the function

$$y = \beta_0 + \beta_1 x + u$$

  is linear. What do we mean?
- We mean that it is linear *in the parameters* — this does not mean that we cannot represent non-linear relationships between y and x with this model
- It means that each parameter is separate from the other parameters, and multiplies a unique term

- To understand what we mean when we say that a model is linear, it might be helpful to look at a model that is *not* linear
- What would a *nonlinear* model look like?

$$y = \left(\frac{\beta_1 x}{\beta_0 + x} \right)^u$$

- I just made that up. It doesn't mean anything, but it is *nonlinear*
- We'll talk more about functional form later — for now I just wanted to clear up what we mean when we say "linear"

Regression modeling

- Sidebar over. Time to start trying this stuff.
- To get us ready for doing (multivariate) regressions in R, let's start playing with data.

Try it

- Load up some data from Wooldridge

R code

```
dta <- read.csv('ceosal1.csv',
stringsAsFactors=F)
# What is in the data?
# What are the basic statistics of the data?
```

Try it

- Load up some data from Wooldridge

R code

```
dta <- read.csv('ceosal1.csv',
stringsAsFactors=F)
# What is in the data?
# What are the basic statistics of the data?
summary(dta)
```

Try it

R code

```
# We are interested in the relationship
# between CEO salary (salary) and return on
# investment (roe)
# Let's look at how they appear to be related
# by examining a simple scatter of the data
plot(dta$roe, dta$salary)
# What is the first thing that you notice?
```

Try it

R code

```
# Eliminate some of the outliers
# so that we can get a better look
condition <- (dta$salary < 5000)
plot(dta$roe[condition],
dta$salary[condition])
```

Try it

R code

```
# Now that we have a rough idea of what
# the data look like, let's run a model
m <- lm(salary ~ roe, data=dta)
summary(m)
```

Try it

R code

```
# Let's draw the model on top of the data
# First, let's get the predictions of the
# model
yhat <- m$fitted.values
# Now plot yhat against roe
```

Try it

R code

```
# Let's draw the model on top of the data
# First, let's get the predictions of the
# model
yhat <- m$fitted.values
# Now plot yhat against roe
plot(dta$roe[condition],
dta$salary[condition])
lines(dta$roe[condition],yhat[condition])
```

Blank

- So far have been doing this:

$$y = \beta_0 + \beta_1 x + u$$

- we are bored with it now
- Almost never do bivariate regression in practical work

- So far have been doing this:

$$y = \beta_0 + \beta_1 x + u$$

- we are bored with it now
- Almost never do bivariate regression in practical work
- Now we are doing this:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + u$$

- or this:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + u$$

Multivariate regression

- So *doing* multivariate regression in R is easy (this is true)
- But it brings complications
- First things first: If our ONLY goal was to predict y (and not to interpret relationships between x and y), then you should throw in the kitchen sink, and forget about interpreting the coefficient estimates

Multivariate regression

- So *doing* multivariate regression in R is easy (this is true)
- But it brings complications
- First things first: If our ONLY goal was to predict y (and not to interpret relationships between x and y), then you should throw in the kitchen sink, and forget about interpreting the coefficient estimates
- So why include multiple x's at the same time? And why be careful about what x's we include/exclude in a model?

Multivariate regression

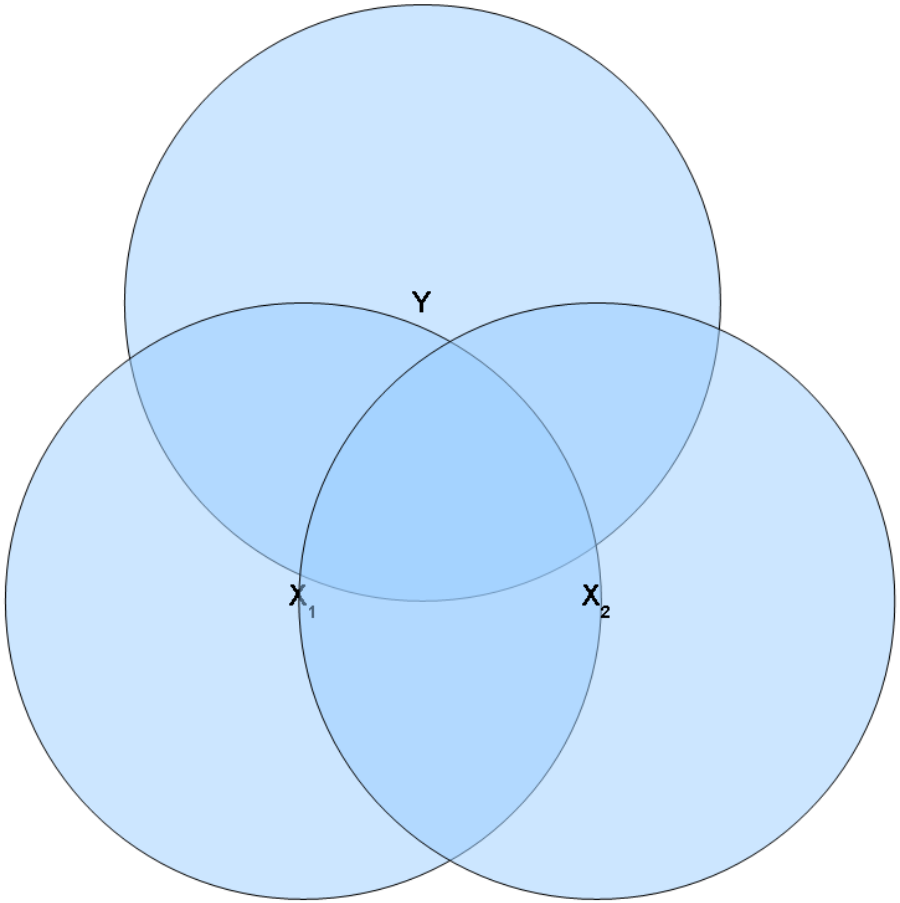
- So *doing* multivariate regression in R is easy (this is true)
- But it brings complications
- First things first: If our ONLY goal was to predict y (and not to interpret relationships between x and y), then you should throw in the kitchen sink, and forget about interpreting the coefficient estimates
- So why include multiple x's at the same time? And why be careful about what x's we include/exclude in a model?
- Because:
- We would like to be able to interpret $\widehat{\beta_k}$ as our best estimate of the relationship between y and x_k (we don't just want to *predict* poverty — we want to be able to say how poverty *changes*). Policy is all about *changes*.
- We would like to know how each x-variable *individually* relates to the single y-variable

- Another way of saying the same thing ...
- We would like to be able to interpret $\widehat{\beta_k}$ as our best estimate of the relationship between y and x_k, *holding other relevant factors constant*
- Including relevant (related) variables allows us to *hold constant* these other factors, so we can focus on the relationship between y and x_k

Multivariate OLS, graphically

- Ballantine time!

Multivariate OLS, graphically
Why is it called the Ballantine?

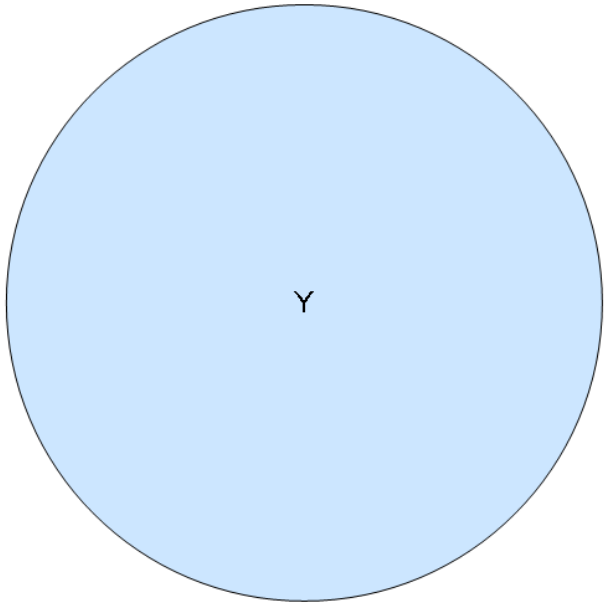


Multivariate OLS, graphically
Why is it called the Ballantine?

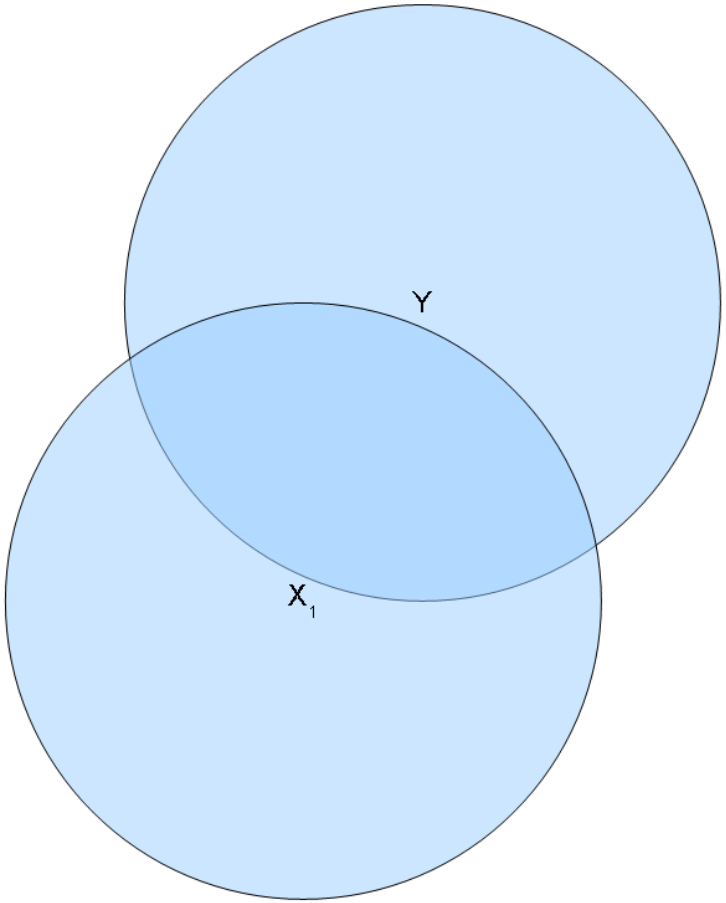
Multivariate OLS, graphically
What the Ballantine is for

- The Ballantine (which we are about to introduce) is useful for thinking about statistical relationships
- I find the Ballantine to be one of the best representations of variance, covariance, and statistical relation between a group of variables
- But it is important to realize that the Ballantine has limits. The Ballantine is a diagram that makes new concepts easier to digest. The Ballantine cannot be used to *prove* anything, and thinking too much using the Ballantine as your only source of intuition is dangerous
- The Ballantine is for: thinking about statistical co-movements
- The Ballantine is NOT for: thinking about the magnitude of systematic relationships

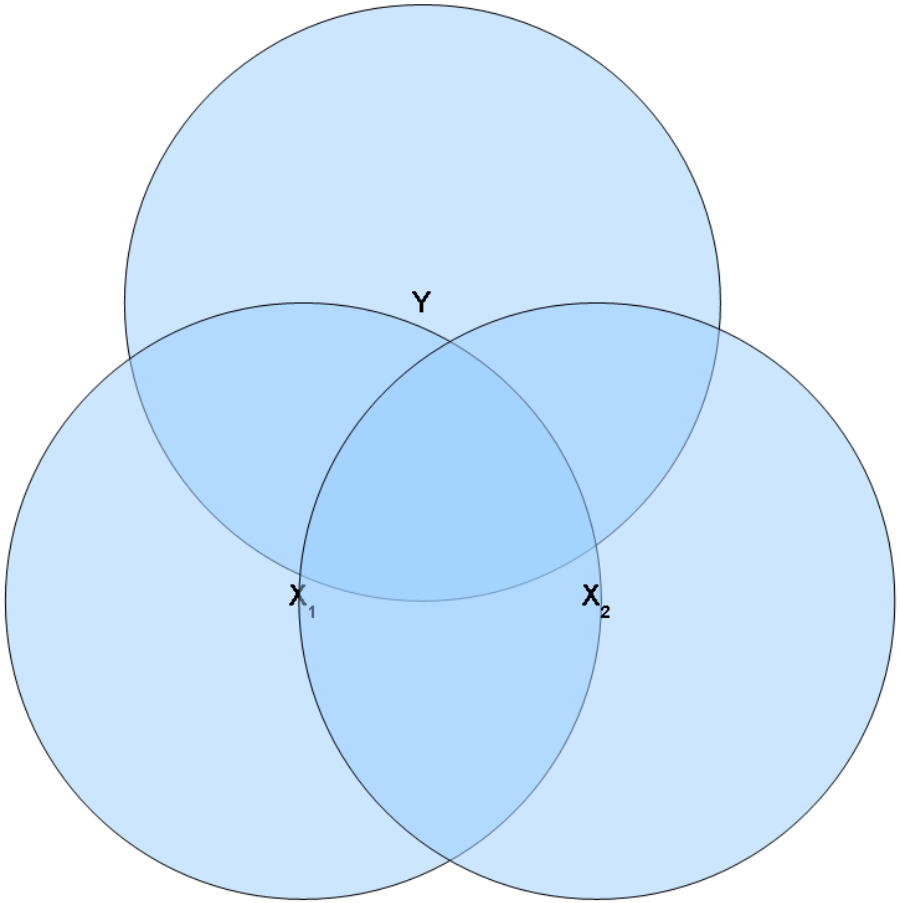
Multivariate OLS estimator
Graphically



Multivariate OLS estimator
Graphically



Multivariate OLS estimator
Graphically



Let's play with data

- I have created some fake data
- We're going to run some regressions on this data
- Why make fake data (why not use real data?)
- Because we can create fake data with properties that we are sure about
- We run regressions under a bunch of different conditions (where we create the conditions) and see what happens
- Then, when we are using real data (i.e. when it counts), we have some intuition about what the results are telling us about how the real variables are related

Let's play with data
What data did I create?

- I created two x-variables (well, really three, but we'll only use two for now . . .) and a y-variable
  - The x-variables are a little bit correlated
  - This is like saying that they overlap a little bit in the Ballantine drawing
- The y-variable is going to be determined by the x-variables and some other random (unobservable) data
- So the y-variable has a Data Generating Process (DGP) that fits the way that we normally model data

Playing with data

- Download “lecture-04-dset-01.RData” from nathanielhiggins.com/fall-2015.html
- Load it into R
- Look at the dset and characterize it
  1. See what’s in it
  2. Look at the variables and think about them in terms of Ballantine bubbles (how big are they? do they overlap? what would a Ballantine of the variables look like?)
  3. Draw the Ballantine!

OLS estimator
Try it

R code

```
# Regress y on x1 (by itself)
m1 <- lm(y ~ x1, data=dta1)
```

R code

```
# Regress y on x1 (by itself)
m1 <- lm(y ~ x1, data=dta1)
# Regress y on x2 (by itself)
m2 <- lm(y ~ x2, data=dta1)
```

OLS estimator
Try it

R code

```
# Regress y on x1 (by itself)
m1 <- lm(y ~ x1, data=dta1)
# Regress y on x2 (by itself)
m2 <- lm(y ~ x2, data=dta1)
# Regress y on x1 and x2
m3 <- lm(y ~ x1 + x2, data=dta1)
```

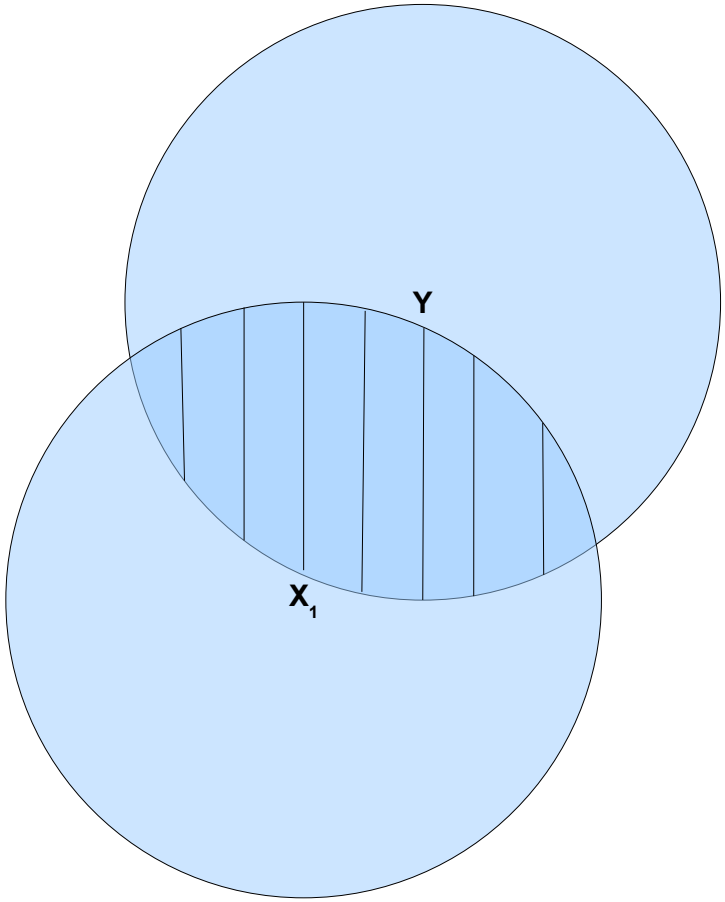
- What do you notice?
- Suppose that you were most interested in the relationship between y and x_1. Does it matter whether or not you include x_2?

- What do you notice?
- Suppose that you were most interested in the relationship between y and x_1. Does it matter whether or not you include x_2?
- Yes!! It matters a lot. Why?

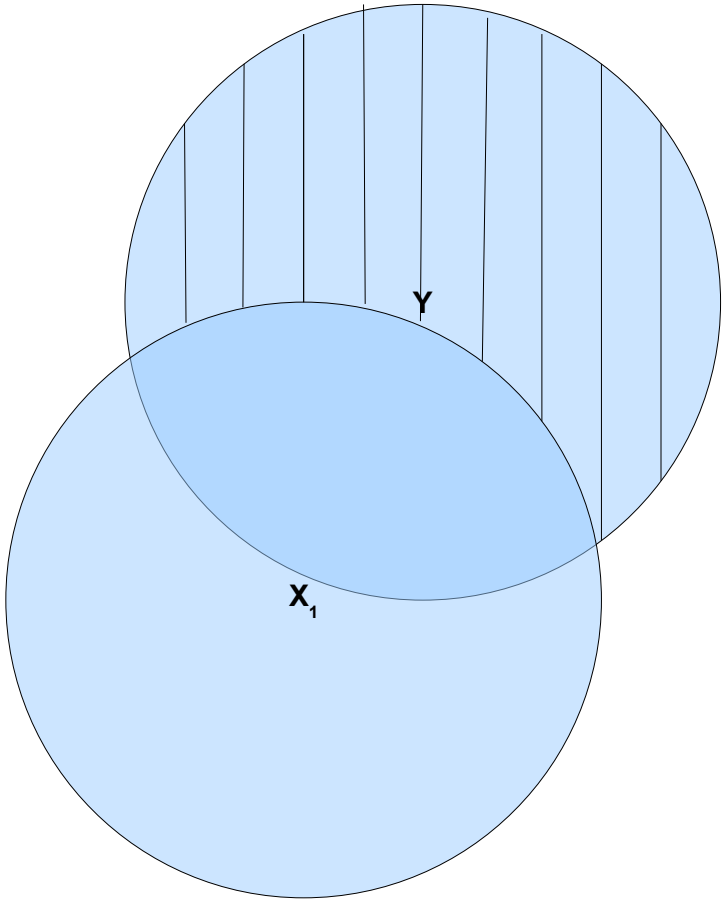
- What do you notice?
- Suppose that you were most interested in the relationship between y and x_1. Does it matter whether or not you include x_2?
- Yes!! It matters a lot. Why?
- Back to the Ballantine . . .

- What do you notice?
- Suppose that you were most interested in the relationship between y and x_1. Does it matter whether or not you include x_2?
- Yes!! It matters a lot. Why?
- Back to the Ballantine ...
- See?

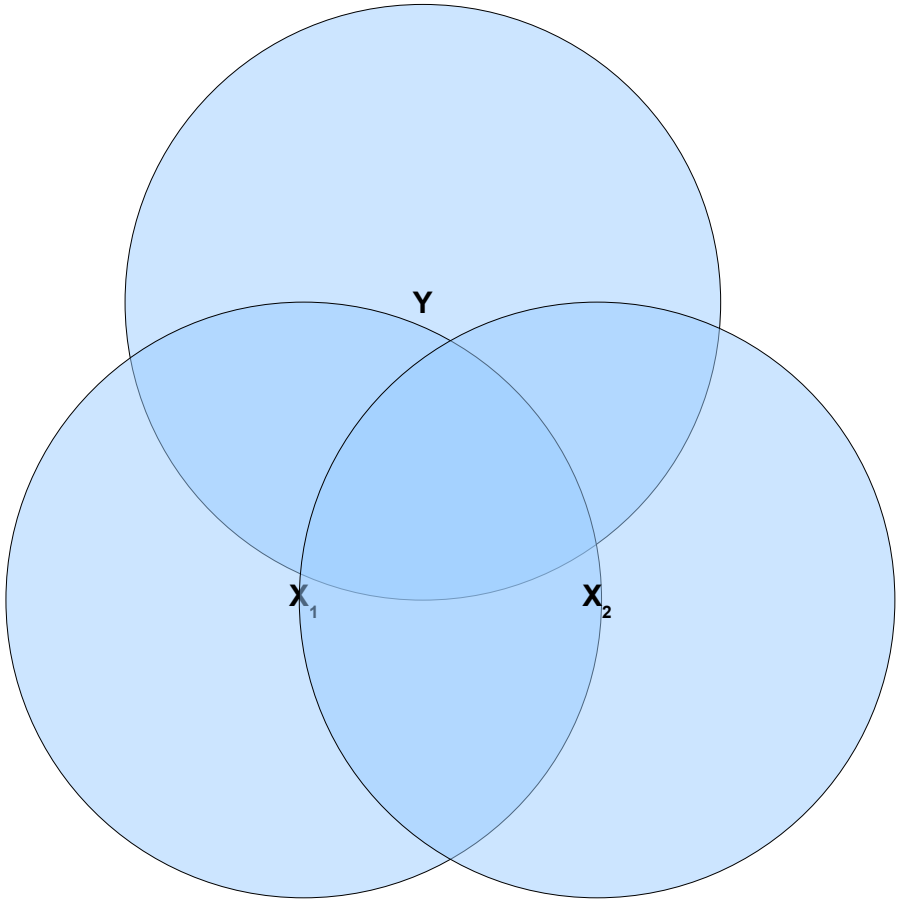
What the OLS estimator does



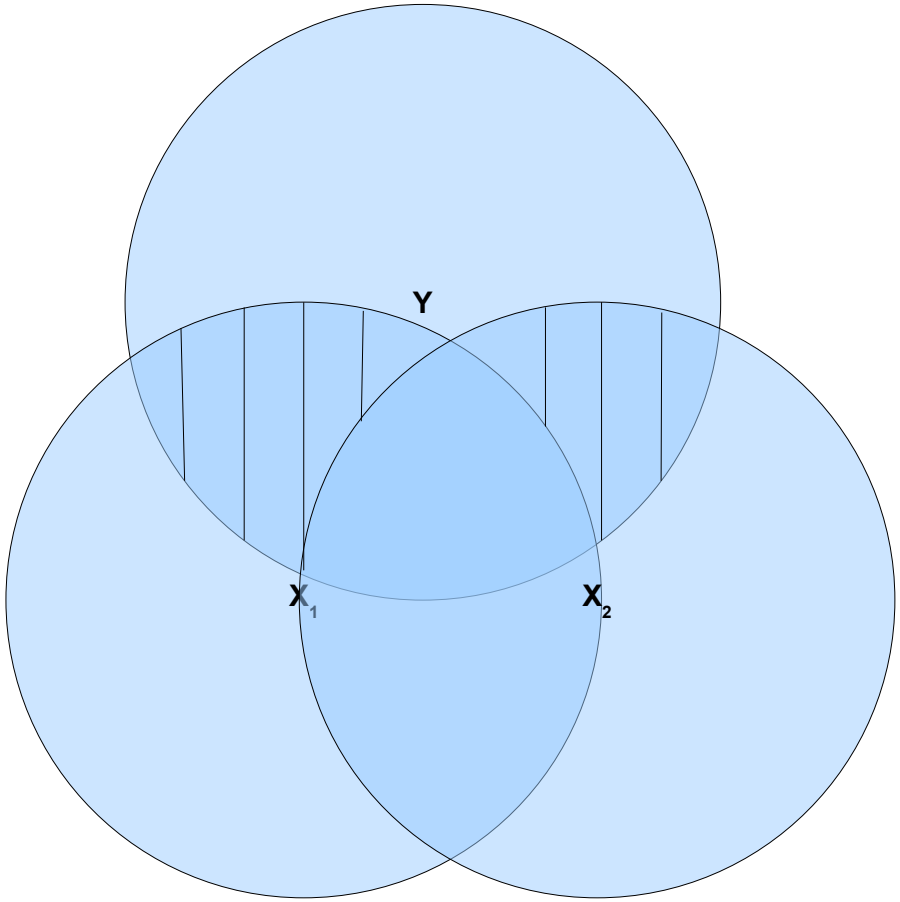
What the OLS estimator does



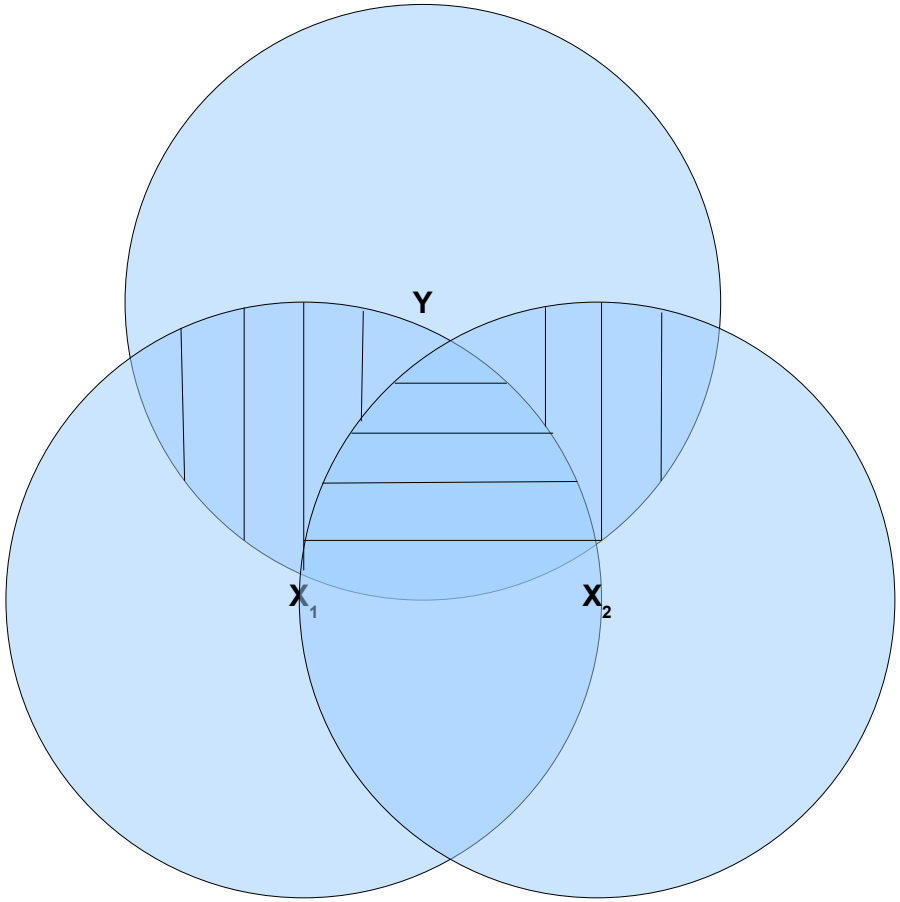
What the OLS estimator does



What the OLS estimator does



What the OLS estimator does



- Switch it up a little bit
- Download “lecture-04-dset-02.RData” from nathanielhiggins.com/fall-2015.html
- Load it into R
- Look at the dset and characterize it
  1. See what’s in it
  2. Look at the variables and think about them in terms of Ballantine bubbles (how big are they? do they overlap? what would a Ballantine of the variables look like?)
- . . . Then repeat the previous exercise

R code

```
# Regress y on x1 (by itself)
m1 <- lm(y ~ x1, data=dta2)
```

OLS estimator
Try it

R code

```
# Regress y on x1 (by itself)
m1 <- lm(y ~ x1, data=dta2)
# Regress y on x2 (by itself)
m2 <- lm(y ~ x2, data=dta2)
```

OLS estimator
Try it

R code

```
# Regress y on x1 (by itself)
m1 <- lm(y ~ x1, data=dta2)
# Regress y on x2 (by itself)
m2 <- lm(y ~ x2, data=dta2)
# Regress y on x1 and x2
m3 <- lm(y ~ x1 + x2, data=dta2)
```

OLS estimator

- What's different?

OLS estimator

- What's different?
- This time, x_1 and x_2 don't overlap much – they aren't very correlated

Tricks with OLS

- See the difference?

Tricks with OLS

- See the difference?
- Lessons:
  1. Omitting variables that *should* be in the regression is potentially harmful. That harm is minimized when the omitted variable is independent of the other variable(s).
  2. It's variation that drives the bus: The more variation there is in an x-variable, the easier it is to estimate the coefficient associated with that variable (the easier it is to *detect* the x-variable's effect on y)
  3. When x-variables are related (when they co-move) it is harder to estimate their independent effects on the y-variable (this is multicollinearity)

Tricks with OLS

- See the difference?
- Lessons:
  1. Omitting variables that *should* be in the regression is potentially harmful. That harm is minimized when the omitted variable is independent of the other variable(s).
  2. It's variation that drives the bus: The more variation there is in an x-variable, the easier it is to estimate the coefficient associated with that variable (the easier it is to *detect* the x-variable's effect on y)
  3. When x-variables are related (when they co-move) it is harder to estimate their independent effects on the y-variable (this is multicollinearity)
- Back to the correlated case
- I hope you're saving your work in a .R-file so that you don't have to type everything again!

Tricks with OLS

- Now we're going to focus on what it means to *hold variables constant*
- We're going to learn how to *isolate* the effect of x_1 on y, even if we don't include x_2 in the regression

Tricks with OLS

- Now we're going to focus on what it means to *hold variables constant*
- We're going to learn how to *isolate* the effect of x_1 on y, even if we don't include x_2 in the regression
- We're sort of going to cheat, but that's OK
- This particular type of cheating is going to provide us with lots of helpful intuition (particularly helpful when we get to instrumental variables)

Tricks with OLS

- I want you to do something that may seem strange at first:

Tricks with OLS

- I want you to do something that may seem strange at first:
- I want you to regress x_1 on x_2
- Do it now

Tricks with OLS

- I want you to do something that may seem strange at first:
- I want you to regress x_1 on x_2
- Do it now
- You have just separated the variation in x_1 into two parts:
  1. The variation in x_1 that can be explained by x_2
  2. The variation in x_1 that cannot be explained by x_2

Tricks with OLS

- I want you to do something that may seem strange at first:
- I want you to regress x_1 on x_2
- Do it now
- You have just separated the variation in x_1 into two parts:
  1. The variation in x_1 that can be explained by x_2
  2. The variation in x_1 that cannot be explained by x_2
- Which part is used to estimate the true β_1?
- Think back to the Ballantine if you're not sure

R code

```
# Regress x1 on x2
m1 <- lm(x1 ~ x2, data=dta1)
# Obtain predictions of x1

# Use those predictions to obtain the
unexplained variation in x1

# Regress y on x1 and x2, so we know
what we're shooting for

# Regress y on x1hat

* Ta-da!
```

R code

```
# Regress x1 on x2
m1 <- lm(x1 ~ x2, data=dta1)
# Obtain predictions of x1
x1hat <- m1$fitted.values
# Use those predictions to obtain the
unexplained variation in x1

# Regress y on x1 and x2, so we know
what we're shooting for

# Regress y on x1hat

* Ta-da!
```

R code

```
# Regress x1 on x2
m1 <- lm(x1 ~ x2, data=dta1)
# Obtain predictions of x1
x1hat <- m1$fitted.values
# Use those predictions to obtain the
unexplained variation in x1
x1u <- dta$x1 - x1hat
# Regress y on x1 and x2, so we know
what we're shooting for

# Regress y on x1hat

* Ta-da!
```

R code

```
# Regress x1 on x2
m1 <- lm(x1 ~ x2, data=dta1)
# Obtain predictions of x1
x1hat <- m1$fitted.values
# Use those predictions to obtain the
unexplained variation in x1
x1u <- dta$x1 - x1hat
# Regress y on x1 and x2, so we know
what we're shooting for
m2 <- lm(y ~ x1 + x2, data=dta1)
# Regress y on x1hat

* Ta-da!
```

R code

```
# Regress x1 on x2
m1 <- lm(x1 ~ x2, data=dta1)
# Obtain predictions of x1
x1hat <- m1$fitted.values
# Use those predictions to obtain the
unexplained variation in x1
x1u <- dta$x1 - x1hat
# Regress y on x1 and x2, so we know
what we're shooting for
m2 <- lm(y ~ x1 + x2, data=dta1)
# Regress y on x1hat
m3 <- lm(dta$y ~ x1u)
* Ta-da!
```

- We have just learned how to isolate variation in our independent variables
- Turns out this is pretty useful
- Lesson
  1. The *only* variation used to estimate the ceteris paribus relationship between an x-variable and y is the variation in x that is independent (not explained) by the other x-variables
- If we want to run the regression

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + u$$

- . . . then the estimate for β_1 (i.e. $\widehat{\beta_1}$) is obtained by "holding constant" the effects of x_2 and x_3 on y
- The OLS estimator does this (does the "holding constant") via a formula that mimics the following procedure

1. Regress x_1 on x_2 and x_3. That is, run the regression

$$x_1 = \alpha_0 + \alpha_1 x_2 + \alpha_2 x_3 + w$$

2. Obtain the predicted x_1 from the regression above; call it $\widehat{x_1}$. (note: don't be confused by the α's — they are just like β's. I am using α's just so we don't confuse the coefficients in this regression with the coefficients in the main regression that has y as the left-hand-side variable)
3. Use the predictions $\widehat{x_1}$ to obtain the unexplained variation in x_1 (the variation in x_1 not explained by variation in x_2 or x_3): $x_1 u = x_1 - \widehat{x_1}$ ($x_1 u$ means "x_1-unexplained")
4. Finally, regress y on $x_1 u$. This gives us the effect of x_1 on y, *independent* of the effect of x_2 and x_3 on y.

Let's play with data
What did I create??

- `x1` and `x2` are a bit correlated
- Represent this with a correlation matrix

$$\texttt{cov.matrix} \leftarrow \begin{pmatrix} 1 & 0.2 \\ 0.2 & 1 \end{pmatrix}$$

- If `x1` and `x2` were not correlated at all, `cov.matrix` would look like this instead

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

- If `x1` and `x2` were really correlated, `cov.matrix` would look like this

$$\begin{pmatrix} 1 & 0.9 \\ 0.9 & 1 \end{pmatrix}$$

R code

```
# Set seed
set.seed(12)
# Create a matrix of correlations
cov.matrix <- matrix( c(1, 0.2, 0.2 , 0.2, 1,
0.2 , 0.2, 0.2, 1), 3,3)
# Create a list of means
means <- (3,2,2)
# Draw three random variable from a
# multivariate distribution
x <- mvrnorm(n=100, means, cov.matrix)
# Draw some "unobservable" stuff
u1 <- mvrnorm(n=100,0,1)
```

R code

```
# Now create y
y <- 5 + 2*x[,1] - 3*x[,2] + u1
# Create a dset

# Name the variables in the dset
```

R code

```
# Now create y
y <- 5 + 2*x[,1] - 3*x[,2] + u1
# Create a dset
dta1 <- as.data.frame(cbind(y,x))
# Name the variables in the dset
```

R code

```
# Now create y
y <- 5 + 2*x[,1] - 3*x[,2] + u1
# Create a dset
dta1 <- as.data.frame(cbind(y,x))
# Name the variables in the dset
names(dta1) <- c("y","x1","x2","x3")
```

OLS estimator
one more thing . . .

- There is one other important thing that we can develop intuition for using this exercise

OLS estimator
one more thing . . .

- There is one other important thing that we can develop intuition for using this exercise
- When we do econometrics in the real world (when we run regressions on real data), the estimates we get are themselves random variables

OLS estimator
one more thing . . .

- There is one other important thing that we can develop intuition for using this exercise
- When we do econometrics in the real world (when we run regressions on real data), the estimates we get are themselves random variables
- The values of the $\widehat{\beta}$s that we get aren't going to be the same if we get new data – every dset we get is one chance to estimate the $\widehat{\beta}$s

OLS estimator
one more thing . . .

- There is one other important thing that we can develop intuition for using this exercise
- When we do econometrics in the real world (when we run regressions on real data), the estimates we get are themselves random variables
- The values of the $\widehat{\beta}$s that we get aren't going to be the same if we get new data – every dset we get is one chance to estimate the $\widehat{\beta}$s
- Every time we collect more data, the DGP is running in the background of the real world. New unobservable stuff is being created all the time. New *y* data is being created all the time.

OLS estimator
one more thing . . .

- There is one other important thing that we can develop intuition for using this exercise
- When we do econometrics in the real world (when we run regressions on real data), the estimates we get are themselves random variables
- The values of the $\widehat{\beta}$s that we get aren't going to be the same if we get new data – every dset we get is one chance to estimate the $\widehat{\beta}$s
- Every time we collect more data, the DGP is running in the background of the real world. New unobservable stuff is being created all the time. New *y* data is being created all the time.
- To the R machine!