Classwork 17: Omitted Variable Bias (part 2: R)

The last thing I want you to know about Econometrics is that simulations can be a really powerful tool to learn more about properties of your estimators and when to expect biased results. In this classwork, you will do a number of simulations considering the relationships between these 3 variables: `earnings`, `education`, and `ability`. You will simulate the data generating process and then you'll consider, out of all possible regressions you could run, which would give you unbiased estimates of causal effects, and which would not.

Hints:

- When your dependent variable has an effect on any of your explanatory variables, you're in trouble. It should be the other way around.
- Look for omitted variable bias in at least one of these regressions. You'll find OVB when X -> Y but X <- U -> Y.
- When X -> Y and X -> U -> Y, you'll get unbiased estimates by estimating Y ~ X. The thing that's going on here is U is a "mediator". X has a direct effect on Y (X -> Y), and X also has an indirect effect on Y (X -> U -> Y). The entire effect of X on Y is the direct effect + the indirect effect, and OLS estimates it.
- 3/9 of these models yield unbiased coefficient estimates.
- Your simulation should start with this: generate a synthetic dataset with 3 variables: ability, education, and earnings. Ability could be random uniform `runif`. Education should be a linear function of ability with some random noise: perhaps generated with `rnorm`. Then earnings should be a linear function of both education and ability, also with some random noise added on.
- After you generate a synthetic dataset, use `lm` to estimate a model and use broom::tidy, slice, and select to find the coefficient estimate you're interested in ($\hat{\beta}_1$ will suffice).
- Run the simulation perhaps 100 times, saving the coefficient estimate for each iteration. You might use `map_dfr(.x, .f)` for this task. Draw a `geom_density()` plot with a vertical line for the true effect of x_1 on y.

For each of these models:

- Run a simulation
- Draw a density plot of your simulation results (coefficient estimates)
- Interpret your simulation results by discussing whether or not OLS seems to be unbiased and why/why not.

1. $Earnings_i = \beta_0 + \beta_1 Education_i + \beta_2 Ability_i + u_i$
2. $Earnings_i = \beta_0 + \beta_1 Education_i + u_i$
3. $Earnings_i = \beta_0 + \beta_1 Ability_i + u_i$
4. $Education_i = \beta_0 + \beta_1 Ability_i + \beta_2 Earnings_i + u_i$
5. $Education_i = \beta_0 + \beta_1 Ability_i + u_i$
6. $Education_i = \beta_0 + \beta_1 Earnings_i + u_i$
7. $Ability_i = \beta_0 + \beta_1 Education_i + \beta_2 Earnings_i + u_i$
8. $Ability_i = \beta_0 + \beta_1 Education_i + u_i$
9. $Ability_i = \beta_0 + \beta_1 Earnings_i + u_i$

Extra Credit: One thing programmers like to say is DRY: "Don't Repeat Yourself".

You may notice that your simulations contain a lot of repeated code. Why is this problematic? Recall that one of our main goals when it comes to programming is to write code that's **clear and readable**. When you have a lot of code that's copy-pasted over and over, it's less clear and harder to read. Code that's hard to read also has the problem that if there's a mistake, it's harder to catch. So your challenge is to write a function `simulation` that takes a model as a character string and a true_beta1 value to draw the vertical line. You should be able to call your function more or less like this:

```
# simulation(model = "earnings ~ education + ability", true_beta1 = 1)
```

And your function should return a density plot of simulation results. After writing `simulation`, hopefully you can go back through and remove all your repeated code.