

Topic 1 Simulation: Introduction to Simulation

Simulation is a very useful tool in econometrics that often helps us to understand a little more about estimators and certain issues. Due to this it can be a very useful tool in teaching econometrics too and this selection of notes is aimed to compliment the materials we discuss in lectures and those provided on Blackboard. Some of the materials may be slightly more advanced, while other sections just recap things we will have already covered. The simulation materials are just to give you another view of what is going on, and if you follow along on STATA they will hopefully make you a more rounded econometrician and more at home on the software.

Now that is out the way, what do I mean when I say simulation? I mean creating your own dataset so that you know what the result should be before we start estimating. Our usual process is to use an estimator, such as OLS, on a subsample of the population to try and predict population estimates. For example if we want to know about the impact of education on wages we might think about the following specification, where wage is determined by an individuals (i) education, experience, gender and if they work full time or part time:

$$Wage_i = Education_i + Experience_i + Gender_i + Full Time_i$$

Given a subsample of the population of workers in the nation, we might try to estimate this equation to find out how important education is in determining wage, while controlling for all other factors. This would mean estimating the following:

$$Wage_i = \hat{\beta}_0 + \hat{\beta}_1 Education_i + \hat{\beta}_2 Experience_i + \hat{\beta}_3 Gender_i + \hat{\beta}_4 Full Time_i + \hat{\varepsilon}_i$$

Where we now have a constant term ($\hat{\beta}_0$), an error term ($\hat{\varepsilon}_i$) and hats above our estimated parameters ($\hat{}$) to show that these values are estimates of the population and not necessarily true values.

The idea with simulation is that we start at the other end, and make a dataset where we already know the values of $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3$ and $\hat{\beta}_4$, then we can check to see if our estimation is providing us with these correct values. Using this we can check to ensure our estimator is not biased and we can define how much error there should be around these values in our data, to check that our standard errors and thus our p-values are correct.

In this set of notes we will look to simulate a simple OLS regression in STATA before using simulation to discuss one of OLS's key assumptions, multicollinearity. There will of course be some STATA commands you have not seen before, but I will try and explain things as well progress... If you are ever stuck with a command, just type help and then the command into STATA and it should display the help file for it, i.e. "help regress".

So to start with we need to tell STATA that we have some observations and I do this using the "set obs" command below, outlining that we want 1000 observations to work with.

set obs 1000

We are going to start on the simple regression equation where there is only one control variable (x):

$$y_i = \hat{\beta}_0 + \hat{\beta}_1 x_i + \hat{\varepsilon}_i$$

So we need to generate a value for x and a value for ε . This can be done using the generate

command and then defining that we want x to be a random draw from the normal distribution (rnormal). X should therefore have a mean of 0 and a standard deviation of 1, although each observation is randomly picked. We can do the same thing to create our error term epsilon too. The line “set seed” just sets the starting point of the random number process so that I can replicate this result again. If you do not set the seed then every time you do this you should end up with entirely different data. Thus while everything should work in a similar fashion you will not see EXACTLY what I document here if you follow along and do not set the seed. If you are following along, once you have generated these variables it might be a good point to browse your created dataset and check that everything looks about right.

```
set seed 101
gen x = rnormal()
gen u = rnormal()
```

Next all we have to do is create our dependent variable y. Here we are defining the relationship between y and x, and thus the values that we insert here should be found by our OLS regression. Here I have set the constant (β_0) to equal 1, and β_1 (the relationship with x) to be 2. So $y=1 + 2x$. We also make y depend on the random error term so that it does not perfectly predict our result giving us no standard errors.

```
gen y = 1 + 2*x + u*5
```

The next step then is to run our regression and see if our OLS is finding the correct values.

```
reg y x
```

This gives me the following:

Figure 1: Regression Output for $y=1+2x$

Source	SS	df	MS	Number of obs	=	1,000
Model	4630.16836	1	4630.16836	F(1, 998)	=	195.85
Residual	23594.3203	998	23.6416035	Prob > F	=	0.0000
				R-squared	=	0.1640
				Adj R-squared	=	0.1632
Total	28224.4886	999	28.2527414	Root MSE	=	4.8623

y	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
x	2.221807	.1587619	13.99	0.000	1.910262	2.533353
_cons	.8071897	.1538024	5.25	0.000	.5053766	1.109003

I have a constant of 0.8, and a value for β_1 of 2.22. Note that our R-squared is not very high here as well. This is mainly due to how large the relationship with the unobserved error term was when we defined y. If you remember the relationship with u was over twice as strong as that with x. Let's change that and see what difference it makes....

```
drop y
gen y = 2 + 4*x + u
```

This time the constant should be 2, with β_1 being equal to 4 and a much smaller role for the unobserved error term. Rather than five times the error term we have only one times the error term added to our y variable. Running our regression again now gives:

Figure 2: Regression Output for $y=2+4x$

Source	SS	df	MS	Number of obs	=	1,000
Model	15342.0719	1	15342.0719	F(1, 998)	=	16223.59
Residual	943.772812	998	.94566414	Prob > F	=	0.0000
				R-squared	=	0.9420
				Adj R-squared	=	0.9420
Total	16285.8447	999	16.3021468	Root MSE	=	.97245

y	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
x	4.044361	.0317524	127.37	0.000	3.982052	4.106671
_cons	1.961438	.0307605	63.76	0.000	1.901075	2.021801

This looks much better. We have tighter estimates of x and the constant term, and the R-squared is much higher too. We are now capturing much more of the relationship as the relationship is now determined more by the observable factors β_0 and β_1 .

How else could we have improved the estimates that our regression provided? One approach could have been to increase the power of the regression by running this on a larger sample size. If you go back to the start of the work and set the number of observations in the data set to a higher number such as 100,000 then you should see the point estimates of β_0 and β_1 become closer to the true values that we set and the standard errors getting smaller. This shows the importance of sample size, and also highlights another key bit of econometrics terminology “consistency”. A consistent estimator is one that approaches the true value of the parameter in the population as the size of the sample, n, increases. OLS is consistent and thus as we increase the sample size the estimates of our parameters (β_0 and β_1) become closer to their true values.

What can we learn from the fact that the standard errors change as the sample size changes? If our standard errors are changing then our p-values are also changing (you cannot see this here as they are already small!) and thus the significance of the coefficient is changing. Why am I raising this as important? We often talk about coefficients being significant and thus an independent variable being a determinant of the dependent variable as a very clear-cut matter. If the p-value is 0.05 or less then the coefficient is significant and the variable is important. The truth, as always, is never so clear. Given the amount of data we have, we set a critical value of 5% and test to see if the coefficient is significant. Setting a higher or lower critical value may change our interpretation of whether a coefficient is significant. Also, changing the sample size will do this.

Going back to our first example (Figure 1) where there was a much larger unobserved error term u we can see this. If you clear the data and generate a dataset with just 10 observations but the same relationship between x and y, you will see that x has a high p-value and thus would not be determined as significant.

```
clear
set obs 10
set seed 101
```

```
gen x = rnormal()  
gen u = rnormal()  
gen y = 1 + 2*x + 5*u  
reg y x
```

If we keep the same relationship but just increase the sample size up to 50 then we suddenly find that the coefficient on x is significantly significant and thus x plays a role in determining y.

```
clear  
set obs 50  
set seed 101  
gen x = rnormal()  
gen u = rnormal()  
gen y = 1 + 2*x + 5*u  
reg y x
```

So is x related to y or not?! The answer is yes and it always was, the underlying relationship between y and x never changed, it was always: $y = 1 + 2x + 5u$.

In the first example however, we did not have enough data to prove this relationship, and that is why our p-value was so large. It is always worth considering that a null result (i.e. where a coefficient is not significant) may just be because you do not have enough data to prove a relationship. It may also be because there is no relationship to find... or we are incorrectly specifying the relationship but these are issues for another day!

Part 2: Multicollinearity and its implications

For OLS to be the best unbiased linear estimator (BLUE) numerous assumptions have to hold¹. You have discussed most of these in detail in term 1, and we will cover heteroskedasticity next week. Just to get used to simulation I want to work through and recap the issue of multicollinearity today in these notes.

Why should we care about multicollinearity and what impact does it have? Well the first reason to care is some form of multicollinearity is pretty much certain in all regressions. However, even extreme multicollinearity (so long as it is not perfect) does not violate the OLS assumptions and thus OLS estimates will still be BLUE. However, the greater the multicollinearity, the greater the standard errors will be in our regression results. Thus when high multicollinearity is present, confidence intervals for coefficients tend to be very wide and t-statistics tend to be very small. Thus coefficients will have to be larger in order to be statistically significant, i.e. it will be harder to reject the null hypothesis that beta equal zero when multicollinearity is present. We might miss finding a relationship of importance due to having a large degree of multicollinearity in our regression.

Okay, let's start with a simple application where there are no issues with multicollinearity (i.e. our independent variables do not overlap at all). We are going to need two independent variables here rather than just one as we used earlier but this should not cause an issue. The following code should now make sense to you, with x being our first independent variable and x2 being our second. Our dependent variable (y) is now a function of a constant, x, x2 and the error term u.

¹ For a recap of the Gauss Markov assumptions this is a pretty nicely written piece:
<https://www.albert.io/blog/key-assumptions-of-ols-econometrics-review/>

```

clear
set obs 1000
set seed 101
gen x = rnormal()
gen x2 = rnormal()
gen u = rnormal()
gen y = 1 + 3*x + 5*x2 + 2*u
reg y x x2

```

Our results look good, the estimator seems to be pulling out the correct value for the constant term, x and x2, with small standard errors.

Figure 3: OLS Results for $y=1+3x+5x_2$

Source	SS	df	MS	Number of obs	=	1,000
Model	32946.3318	2	16473.1659	F(2, 997)	=	3934.84
Residual	4173.93202	997	4.1864915	Prob > F	=	0.0000
				R-squared	=	0.8876
				Adj R-squared	=	0.8873
Total	37120.2638	999	37.1574213	Root MSE	=	2.0461

y	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
x	2.895934	.066874	43.30	0.000	2.764704	3.027164
x2	5.024152	.0666026	75.43	0.000	4.893454	5.154849
_cons	.9912961	.0647726	15.30	0.000	.8641899	1.118402

The next step then is to try and build some multicollinearity in between x and x2, i.e. have them be related in some way. While doing this I want to keep the same underlying relationship between the independent variables and y so that we are still testing the same relationship so I will keep $y = 1 + 3x + 5x_2 + 2u$. First I will drop x2 as we want to make a new x2 that is related to x in some way. Then I will create x2 to be x and a random term (rnormal). If I create x2 to just be equal to x, or a simple linear transformation of x, then STATA will automatically remove it from the regression as it will be perfectly correlated. It is worth a trying this yourself so you can check it, but it does not illustrate the impacts of multicollinearity that I am interested in showing today. To check that our two independent variables are related I will just run a quick correlation between x and x2 (**corr x x2**). This shows what we expect/created, that the two independent variables are highly correlated, with a correlation coefficient of 0.7 (with 0 being no correlation and 1 being perfectly correlated).

```

drop x2
gen x2= x+rnormal()
corr x x2

```

Figure 4: Correlation coefficients between x and x2 where x2 is x+rnormal(0,1)

```
. corr x x2
(obs=1,000)
```

	x	x2
x	1.0000	
x2	0.7002	1.0000

We need to update our y term so that it is built on this new value of x2 rather than the old one, so we will drop it and create it again using the same line of code, then we can retest our regression.

```
drop y
gen y = 1 + 3*x + 5*x2 + 2*u
reg y x x2
```

Figure 5: Regression Output for $y=1+3x+5x_2$ where there is reasonably high correlation between x and x2

Source	SS	df	MS	Number of obs	=	1,000
Model	80416.2818	2	40208.1409	F(2, 997)	=	9628.17
Residual	4163.56489	997	4.17609317	Prob > F	=	0.0000
				R-squared	=	0.9508
				Adj R-squared	=	0.9507
Total	84579.8467	999	84.6645112	Root MSE	=	2.0435

y	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
x	3.002816	.0934603	32.13	0.000	2.819414	3.186217
x2	4.892922	.066225	73.88	0.000	4.762965	5.022878
_cons	.9896901	.0646426	15.31	0.000	.862839	1.116541

Our results don't actually look too bad at this point. This might be because there is not too much multicollinearity between our variables at the moment. We can check this quickly by running one of the common tests for multicollinearity, the VIF test. As you can see below the mean from the VIF is 1.96. With VIFs the rule of thumb is that a value over 10 is an issue and here you can see we are not even close to this.

```
vif
```

Figure 6: VIF results for previous model where x2 is x+rnormal(0,1)

```
. vif
```

Variable	VIF	1/VIF
x	1.96	0.509720
x2	1.96	0.509720
Mean VIF	1.96	

So we can see with multicollinearity it is not much of an issue unless it is rather extreme. This is more likely with more independent variables as they can all be highly related to each other, whereas here we only have two controls and thus the possibility of multicollinearity is smaller. Let's try to adjust our x2 value again to create stronger multicollinearity and see what impact this has on our results. I will follow the same steps as last time but rather than create x2 to be x plus a random normal element (mean zero and standard deviation 1), I will reduce the random element. By writing rnormal(0,0.1) I have defined that I want a mean of zero still, and the standard deviation to be 0.1, much lower than the previous value of 1. Thus my new x2 term should have a much higher correlation with x (as the random element of the relationship is smaller). When we check our correlation of x and x2 now we do see much higher correlation, this time at 0.99.

```
drop x2
gen x2= x+rnormal(0,0.1)
corr x x2
```

Figure 7: Correlation coefficients between x and x2 when x2 is x+rnormal(0,0.1)

```
. corr x x2
(obs=1,000)
```

	x	x2
x	1.0000	
x2	0.9949	1.0000

```
drop y
gen y = 1 + 3*x + 5*x2 + 2*u
reg y x x2
```

Looking at the regression results with this new updated x2 variable we can see that multicollinearity is starting to play a much more significant role. Our point estimates of x and x2 are now much further away from their true values than our first regression where x was estimated to be 2.89 and x2 was estimated to be 5.02. The difference in the standard errors is huge too, in the first regression we had standard errors of around 0.06, and in this regression with high multicollinearity this is up to 0.66, over ten times higher. This has not impacted on our interpretation of whether the coefficient is significantly different to zero in this particular case (due to other factors such as the sample size and the strength of the relationship we are testing), but in other case multicollinearity could both lead to poor coefficient estimates and much larger standard errors. Looking at the VIF results from this regression we see that the mean is way above our rule of thumb value 10 and thus the test is suggesting we should be

worried about multicollinearity here. This is supported by our regression results where we know the estimates are not the best.

Figure 8: Output for $y=1+3x+5x^2$ where there is very high correlation between x and x^2

```
. reg y x x2
```

Source	SS	df	MS	Number of obs	=	1,000
Model	59230.2941	2	29615.147	F(2, 997)	=	7078.75
Residual	4171.1186	997	4.18366961	Prob > F	=	0.0000
Total	63401.4127	999	63.4648775	R-squared	=	0.9342
				Adj R-squared	=	0.9341
				Root MSE	=	2.0454

y	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
x	2.303956	.6647315	3.47	0.001	.9995225 3.608389
x2	5.589599	.65752	8.50	0.000	4.299317 6.879881
_cons	.9906434	.0647006	15.31	0.000	.8636784 1.117608

Figure 9: VIF results for previous model where x^2 is $x+\text{rnormal}(0,0.1)$

Variable	VIF	1/VIF
x	99.06	0.010094
x2	99.06	0.010094
Mean VIF	99.06	

What can we learn from all of this? Firstly, multicollinearity does not seem to be much of an issue if it is of a low level. With more independent variables it is more likely than in this example however. Secondly, running quick correlation tests between independent variables and a VIF after a regression are very quick methods of checking for any potential issues when you do not know the true values and therefore do not know in advance if there is multicollinearity. Finally, it is nice to know that the VIF test threshold of 10 seems to appear sensible from these two examples. Playing around a little more we might be able to test if issues with standard errors and the estimates of our coefficients become an issue before this value, but generally it seems about right and a good rule of thumb to stick to!

I hope this did provide some insights and an alternative perspective on some of the topics we are covering, if you have any problems working through sections do let me know and I will be happy to help!

See you all next week to discuss heteroskedasticity!

David Morris