Lab #11 - Instrumental Variables Part I

Econ 224

October 16th, 2018

Generating Correlated Normal Draws

The function `mvrnorm` from the package `MASS` is used to generate draws from a multivariate normal distribution. It's not important that you know the formal definition of a multivariate normal for this course. All that you need to know is that we can use this distribution to make random normal draws that are *correlated* with one another. The pacakge `MASS` is automatically installed as part of R, so you don't need to install it manually. Here's an example of a *bivariate* normal. This means that each draw is a vector with two elements. See if you can figure out how this example works, consulting the help files as necessary:

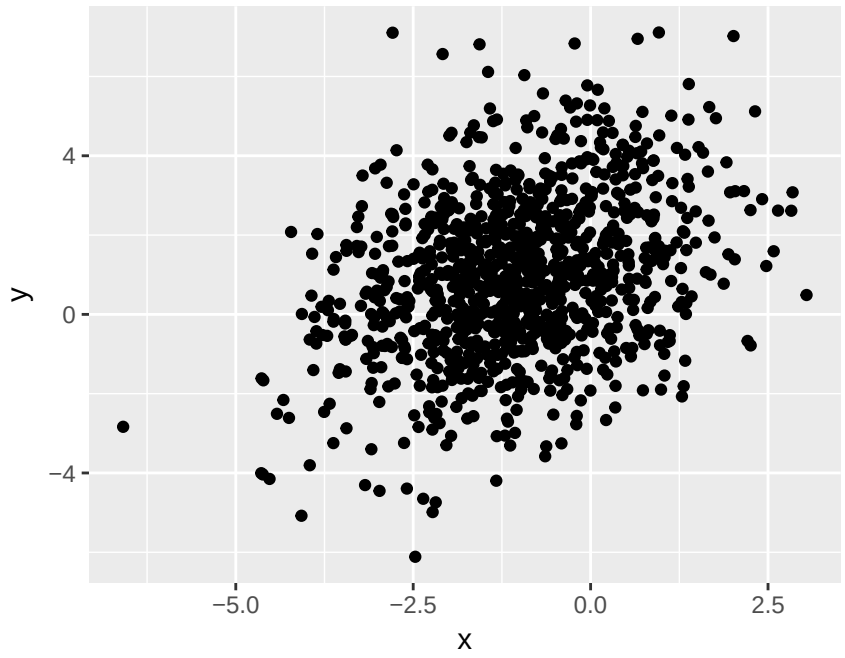
```
library(MASS)
m <- c(-1, 1)
S <- matrix(c(2, 1, 1, 4), 2, 2, byrow = TRUE)
set.seed(1234)
sims <- mvrnorm(1000, m, S)
head(sims)
```

```
           [,1]      [,2]
[1,] -0.5681911 -1.923861
[2,] -1.1276750  1.683789
[3,]  1.6625885  2.363257
[4,] -3.6251906 -3.246983
[5,] -1.4728059  2.171719
[6,]  1.6242268  1.063836
```

Exercise

1. Store the first column of `sims` in a vector called `x` and the second in a vector called `y`. Plot `x` against `y`.
2. Calculate the length of `x` and `y`.
3. Calculate the sample mean of `x` and `y`. How does your result compare to the vector `m`?
4. Calculate the sample variance of `x` and `y`.
5. Calculate the sample covariance of `x` and `y`.
6. What result do you get if you run `var(sims)`? How does your result compare to the matrix `S`?
7. Based on your answers to the above and the R helpfiles, figure out how to use `mvrnorm` to generate 1000 draws from a bivariate normal distribution where each of the two elements is a *standard normal* and the correlation between them is 0.5. Check that your code works as expected using `cor`, `mean`, etc.

```
x <- sims[,1]
y <- sims[,2]
library(ggplot2)
ggplot() + aes(x, y) + geom_point()
```




```
colMeans(sims)
```

```
[1] -1.0382639  0.9553645
```

```
var(sims)
```

```
          [,1]      [,2]
[1,] 1.8424523 0.9089161
[2,] 0.9089161 4.0749877
```

The Instrumental Variables Equations

There are two equations: one for `y` and one for `x`:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$
$$x_i = \pi_0 + \pi_1 z_i + v_i$$

The equation for `y` is called the *structural equation*. It shows how the regressor `x` causes the outcome `y`. The coefficient β_1 is the causal effect of `x` on `y`. This may *not* be the same thing as the slope from a regression of `y` on `x` because ϵ_i is a *structural error* rather than a *regression error*. The equation for `x` is called the *first-stage*. This equation shows the relationship between the instrument `z` and the regressor `x`. The coefficients π_0 and π_1 are simply defined as the intercept and slope from a regression of x on z. The error v is a regression error term so it is uncorrelated with z. In the following exercise, you will explore what this means.

Exercise

1. Use what you know about linear regression to express π_1 in terms of variances and covariances.
2. Suppose we *define* $v_i = x_i - \pi_0 - \pi_1 z_i$. Calculate $\text{Cov}(v_i, z_i)$ using your answer to part 1.

3. Substitute the *first-stage* equation (the equation for x_i) into the *structural equation* (the equation for y_i) to produce a linear equation relating z_i to y_i. This is called the *reduced form*. What is the slope coefficient of the reduced form?
4. Suppose that ϵ_i and v_i are correlated with one another. If we run a linear regression of y on x, what slope coefficient will we obtain?

Simulating Data for IV Regression

We will now generate some data to use for IV estimation, using what you learned in the proceeding two exercises. The precise steps you will need are listed in the exercise below.

Exercise

Simulate data for two-stage least squares estimation using the following procedure:

1. Set the seed of the random number generator to `1234` so you can replicate your results later.
2. Define a variable called `n` to use as your sample size. Set it equal to `1000`.
3. Create a matrix called `Rho` that will serve as the variance covariance matrix of the error terms ϵ and v in the simulation. Set the variance of each to `1` and the correlation between them to `0.5`.
4. Make `n` bivariate normal draws with mean zero and variance-covariance matrix `Rho`. Store the as `sims`.
5. Extract the first column of `sims` and store it as a vector called `e`. Extract the second column and store it as a vector called `v`.
6. Make `n` iid Uniform(0,1) random draws and store the result in a vector called `z`.
7. Generate a vector called `x` using the IV first-stage equation with $\pi_0 = 0.5$ and $\pi_1 = 0.8$.
8. Generate a vector called `y` using the IV structural equation with $\beta_0 = -0.3$ and $\beta_1 = 1$.

```
set.seed(1234)
n <- 1000
Rho <- matrix(c(1, 0.5, 0.5, 1), 2, 2, byrow = TRUE)
errors <- mvrnorm(n, c(0,0), Rho)
e <- errors[,1]
v <- errors[,2]
z <- runif(n)
x <- 0.5 + 0.8 * z + v
y <- -0.3 + x + e
```

OLS Estimation

In the simulation from the previous section, x is *endogenous*: in other words it is *correlated* with the error term ϵ in the structural equation. We can think of this as a situation where ϵ_i contains an important *omitted variable* that is correlated with x.

Exercise

1. Run an OLS regression of `y` on `x`. Do your results give the causal effect of `x` on `y`? Why or why not?
2. Using the formulas you worked out above, how would your results change if the correlation between ϵ and v were -0.5 rather than 0.5?

```r
coef(lm(y ~ x)) # Slope is much too large!
```

```
(Intercept)           x
 -0.7397178   1.4606086
```

IV Estimation "By Hand"

We will now carry out two-stage least squares estimation "by hand." In this particular example, this is overkill, but it is helpful to do it anyway just to make sure that you understand what's going on.

1. Run an OLS regression of `x` on `z` and store the result as `first_stage`.
2. Run an OLS regression of `y` on `z` and store the result as `reduced_form`. How does the estimated slope agree with your mathematical calculation of the reduced form slope from above?
3. Divide the slope from `reduced_form` by the slope from `first_stage`. How does your result compare to `cov(y,z) / cov(x,z)`. Compare both to the true causal effect of `x` on `y`

```r
first_stage <- lm(x ~ z)
reduced_form <- lm(y ~ z)
coef(reduced_form)
```

```
(Intercept)           z
  0.1615314   0.7849772
```

```r
coef(reduced_form)[2] / coef(first_stage)[2]
```

```
        z
0.9582441
```

```r
cov(y,z) / cov(x,z)
```

```
[1] 0.9582441
```

IV Estimation using `ivreg`

Install the package `AER`. This package contains a number of useful functions for econometrics, including `ivreg` which we'll use to carry out IV regression. Before proceeding, load `AER` and read the help file for `ivreg`.

Exercise

1. Use `ivreg` to carry out IV regression that you did by hand in the preceding exercise. Store the results as `iv_results`.
2. Display the IV results using `summary`.
3. Calculate an approximate 95% confidence interval for the causal effect of `x` on `y` using `iv_results`.

```r
library(AER)
```

```
Loading required package: car
```

```
Loading required package: carData

Loading required package: lmtest

Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

Loading required package: sandwich

Loading required package: survival
```

```r
iv_results <- ivreg(y ~ x | z)
summary(iv_results)
```

```
Call:
ivreg(formula = y ~ x | z)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8096 -0.6405 -0.0326  0.6351  3.0896

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.2932     0.1206  -2.431   0.0152 *
x             0.9582     0.1310   7.313 5.34e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9918 on 998 degrees of freedom
Multiple R-Squared: 0.6764, Adjusted R-squared: 0.6761
Wald test: 53.49 on 1 and 998 DF,  p-value: 5.34e-13
```