ECOMAss2

NathanAung

April 18, 2017

```
knitr::opts_chunk$set(echo = TRUE)
set.seed(42)
library(R6) #using R6 oop system
library(AER)
```

```
## Loading required package: car
```

```
## 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
```

```
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2015). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2. http://CRAN.R-project.org/package=stargazer
```

Q1

Define two classes that generates results:

```
#Class that generates all combinations in a vector
regMCloop = R6Class(
  public = list(
    nrange=NULL,
    sigmarange=NULL,
    Urange=NULL,
    a1range=NULL,
    a2range=NULL,
    results = NULL,

    initialize = function(nrange,sigmarange,Urange,a2range=0,a1range=1)
    {
      self$nrange = nrange
      self$sigmarange = sigmarange
      self$Urange = Urange
      self$a2range = a2range
```

```
      self$a1range = a1range
      self$results = vector("list",length(nrange)*length(sigmarange)*length(Urange)*length(a2range))#li
    },

    loop = function( a3=1, b1=1, b2=1, b3=0,IV=FALSE)
    {
      i=1#elements of result vector
      for(n in self$nrange)
      {
        for(sigma in self$sigmarange)
        {
          for(dist in self$Urange)
          {
            for(a2 in self$a2range)
            {
              for(a1 in self$a1range)
                {
                  self$results[[i]] = regMClass$new(ns=n,sigma=sigma,dist=dist, a2=a2, a3=a3, b1=b1, b2=
                  i=i+1#next result
                }
            }
          }
        }
      }
    }


  )
)
```

```
regMClass = R6Class(
  public = list(
    bias = NULL,
    sd = NULL,
    CovRate=NULL,
    CovLength=NULL,
    bias2 =NULL,
    sd2 = NULL,
    CovRate2 = NULL,
    CovLength2 = NULL,
    ns = NULL,
    sigma= NULL,
    dist= NULL,
    a2 =NULL,
    a1 =NULL,
    fittedvalues = NULL,


    initialize = function(ns = 20, n=1000, b1=1, b2=1, b3=0, a1=1, a2=0, a3=1, sigma=1,dist="norm", IV=
    {
      intervalrange =vector(mode="numeric",length=0)
      intervalrange2 =vector(mode="numeric",length=0)
      b1list = vector(mode="numeric",length=0)
      b2list = vector(mode="numeric",length=0)
```

```
inInterval=vector(mode="logical",length=0)
inInterval2=vector(mode="logical",length=0)#declare vectors so that append method works
for(j in 1:n)
{
  #__init__
  #we dont use a multivariate normal to generate it as this is equivalent.(independant and jointly
  Z1 = rnorm(n=ns,mean=0,sd=1)
  X2 = rnorm(n=ns,mean=0,sd=1)
  X3 = rnorm(n=ns,mean=0,sd=1)
  V = rnorm(n=ns,mean=0,sd=1)

  if (dist == "norm")
  {
    U = rnorm(n=ns,mean=0,sd=1)
  }
  else
  {
    U = rlnorm(n=ns,mean=0,sd=1)
    U=(U-mean(U))/sd(U)#z score rescaled
  }
  #Generate X1 and Y
  X1 = a1*Z1 + a2*X2 + a3*X3 +V
  X1 = X1/sd(X1) #renormalising so all values have sd of 1
  Y = b1*X1 + b2*X2 +b3*X3 + sigma*U
  #Estimates
  if(IV==FALSE){
  hat = lm(Y ~ X1+X2)
  }

  else{
  hat = ivreg(Y~X1+X2|X2+Z1)
  }
  b1list=append(b1list,hat$coefficients["X1"])
  b2list=append(b2list,hat$coefficients["X1"])
  #In confidence interval?
  #b1
  interval = confint(hat,parm = "X1",interval="confidence")
  inInterval = append(inInterval,b1<interval[2]&b1>interval[1])
  intervalrange = append(intervalrange,interval[2]-interval[1])
  #b2
  interval2 = confint(hat,parm = "X2",interval="confidence")
  inInterval2 = append(inInterval2,b2<interval2[2]&b2>interval[1])
  intervalrange2 = append(intervalrange2,interval2[2]-interval2[1])
  fittedvalues = hat$fitted.values

  #note that this could be probably be optimised by a running mean,sd tally . This is however a l

}
self$ns = ns # so we can sort by values used
self$sigma = sigma
self$dist = dist
self$a2 = a2
self$a1 = a1
```

```
      self$bias = mean(b1list)-1
      self$sd = sd(b1list)
      self$CovRate = mean(inInterval)
      self$CovLength = mean(intervalrange)

      self$bias2 = mean(b2list)-1
      self$sd2 = sd(b2list)
      self$CovRate2 = mean(inInterval2)
      self$CovLength2 = mean(intervalrange2)

      self$fittedvalues = fittedvalues
    }
  )
)
```

We use a nested loop to generate all combinations needed and store them in a vector of results. These results are then stored within an object. It is possible that a recursive function approach could have been more elegant but for the number of combinations here, nested loop should suffice.

```
dataQ1 = regMCloop$new(nrange=c(20,200,2000),sigmarange=c(1,2),Urange=c("norm","lognorm"))
dataQ1$loop()
```

Here we create an instance of the class and use the loop methods to generate the results. This does take some time(~20s) as we are looping over quite a large number of iterations. We shall show the code here for this instance but as the code is largely similar, we will not show it for the sake of cluttering in following sections. This probably could have been a method for the regMCloop class.

```
    Bias = vector(length=12)
    SD = vector(length=12)
    CovRate = vector(length=12)
    CovLength = vector(length=12)
    for(i in 1:12){
    Bias[i] = (dataQ1$results[[i]]$bias)
    SD[i] =   (dataQ1$results[[i]]$sd)
    CovRate[i] = (dataQ1$results[[i]]$CovRate)
    CovLength[i]=(dataQ1$results[[i]]$CovLength)
    }
    ndf=c(rep(20,4),rep(200,4),rep(2000,4))
    sigma  = c(rep(1,2),rep(2,2),rep(1,2),rep(2,2),rep(1,2),rep(2,2))
    Distribution = rep(c("Normal","LogNormal"),6)
    Q1=data.frame(ndf,sigma,Distribution,Bias,SD,CovRate,CovLength)
    stargazer(Q1,header=FALSE,type="latex",summary=FALSE)
```

a)

In all combinations, |bias| is unanimously within $\pm$ 0.005 of 0. Considering that even under all combinations in which the number of samples is 20 that this is present, these results strongly indicate that the estimator is or very close to unbiased. It is impossible to say definitively due to the stochastic nature of the simulation, however the weak law of large numbers and central limit theorem point to this conclusion. This seems appropriate considering the theoretical aspects of the estimator. This is explained by the Gauss Markov Theorem, which guarantees that the OLS is the best unbiased linear estimator.

Table 1:

	ndf	sigma	Distribution	Bias	SD	CovRate	CovLength
1	20	1	Normal	0.010	0.242	0.949	0.987
2	20	1	LogNormal	-0.003	0.235	0.956	0.997
3	20	2	Normal	0.003	0.477	0.958	1.925
4	20	2	LogNormal	0.018	0.477	0.946	1.996
5	200	1	Normal	-0.005	0.068	0.957	0.280
6	200	1	LogNormal	0.003	0.071	0.954	0.280
7	200	2	Normal	-0.002	0.146	0.948	0.560
8	200	2	LogNormal	0.008	0.148	0.939	0.560
9	2,000	1	Normal	0.001	0.022	0.946	0.088
10	2,000	1	LogNormal	0.001	0.022	0.953	0.088
11	2,000	2	Normal	0.001	0.045	0.944	0.175
12	2,000	2	LogNormal	0.001	0.044	0.950	0.176

b)

For each combination, of σ and distribution, we vary the number of samples and observe the results. In nearly all cases, bias decreases monotonically throughout. In the few cases where it does not, bias is very close to zero and is still within the same order of magnitude of the previous estimate. This is to be expected even with a consistent estimator as the estimator becomes closer and closer to the true value of the variable. Perhaps with a higher number of simulations, this error would be reduced. Further, testing across a larger number of sample sizes may have graphically indicated the monotonicity of the bias as sample size increased. With only three data points, there could be any number of small changes in between.

c)

Coverage rate is consistently within 0.15 of the expected 95% coverage. Coverage rates between normal and log normal distributions appear to be inconclusive. Similarly, σ appears to have limited effect on coverage rate. The number of samples also inconclusive. One could of course perform an anova test to show these and reveal possible interaction effects.

d)

For all combinations, log normal would have generated much large confidence intervals than its normal counterpart.

This can be seen by examining the variance of normal and log normal distributions. With standard deviation σ, the variance for normal and lognormal respectively is:

$$\sigma^2$$

$$(e^{\sigma^2} - 1)(e^{2\mu+\sigma^2})$$

However because we z-score standardised the data in some cases it may be more accurate than the normal case - z-score scaling skews the data as every lognormal error has mean 0 and standard deviation 1. Obviously, if one draws from a distribution, not every sample will have this present.

One can verify that for all values the variance will be greater for the log normal case . Therefore the estimated standard deviation will be much larger for the distribution and thus parameter estimates will have a larger confidence interval for the same coverage rate.

This can be verified again as for unitary σ, the coverage rate is far smaller than their $\sigma = 2$ counterparts.

Sample size reduces interval length drastically for all combinations.

2a)

```
dataQ2a = regMCloop$new(nrange=c(20,200,2000),sigmarange=c(1,2),Urange=c("norm","lognorm"))
dataQ2a$loop(b3=1,a3=1)
```

Table 2:

	ndf	sigma	Distribution	Bias	SD	CovRate	CovLength
1	20	1	Normal	0.555	0.323	0.594	1.279
2	20	1	LogNormal	0.583	0.322	0.579	1.281
3	20	2	Normal	0.549	0.514	0.813	2.129
4	20	2	LogNormal	0.576	0.515	0.834	2.150
5	200	1	Normal	0.572	0.099	0	0.361
6	200	1	LogNormal	0.576	0.094	0	0.361
7	200	2	Normal	0.580	0.158	0.035	0.604
8	200	2	LogNormal	0.574	0.153	0.035	0.606
9	2,000	1	Normal	0.577	0.029	0	0.113
10	2,000	1	LogNormal	0.579	0.030	0	0.113
11	2,000	2	Normal	0.578	0.049	0	0.190
12	2,000	2	LogNormal	0.578	0.049	0	0.189

Calculate bias here: It is clear that omitted variable bias is present here with an apparent consistent bias of 1. This would be ~0.33 if X1 was not rescaled.. We can calculate the theoretical bias here:

$$y = X_1\beta_1 + X_2\beta_2 + X_3\beta_3 + U$$

$$X_1 = \alpha_1 Z_1 + \alpha_3 X_3 + V_i$$

OLS estimates:

$$E(\hat{\beta_1}) = E\frac{\text{Cov}(X_1, y)}{\text{Var}(X_1)}$$

$$E(\hat{\beta_1}) = E\frac{\text{Cov}(X_1, X_1\beta_1 + X_2\beta_2 + X_3\beta_3 + U)}{\text{Var}(Z_1 + X_3 + U)}$$

$$E(\hat{\beta_1}) = E\frac{\text{Cov}(X_1, X_1\beta_1) + \text{Cov}(X_1, X_2\beta_2) + \text{Cov}(X_1, X_3\beta_3) + \text{Cov}(X_1, U)}{\text{Var}(Z_1 + X_3 + U)}$$

$$E(\hat{\beta_1}) = E\frac{\beta_1 Var(X_1) + \beta_2 0 + \text{Cov}(Z_1 + X_3 + U, X_3\beta_3) + 0}{\text{Var}(Z_1 + X_3 + U)}$$

$$E(\hat{\beta_1}) = E(\beta_1 + \frac{\text{Cov}(Z_1 + X_3 + U, X_3\beta_3)}{\text{Var}(Z_1 + X_3 + U)})$$

$$E(\hat{\beta_1}) = E(\beta_1 + \frac{\beta_3\text{Cov}(Z_1, X_3) + \beta_3\text{Cov}(X_3, X_3) + \text{Cov}(U, X_3)}{1 + 1 + 1})$$

$$E(\hat{\beta_1}) = \beta_1 + \frac{1}{3}$$

$$\text{Bias} = \frac{1}{3}$$

When X1 is rescaled,the bottom term is simply 1, and thus bias becomes 1.

Standard deviation is also uniformly higher compared to a) and as such confidence intervals are larger. Coverage rate is especially poor here indicating a severe deviation from the requirements and assumptions of linear regression.

Whilst X_3 determines X_1 in both 1) and 2a), it does not determine Y in 1). As such, it does not fulfill the requirements for an omitted variable bias as endogeniety does not occur:

1) The omitted variable must be a determinant of the dependant variable(ie. regression coefficient is non-zero)
2) Omitted variable is correlated with the explanatory variable.

Whilst condition 2) is present through both questions, condition 1 only holds for question 2a).

2b)

```
dataQ2b = regMCloop$new(nrange=c(20,200,2000),sigmarange=c(1,2),Urange=c("norm","lognorm"))
dataQ2b$loop(b3=1,a3=0)
```

Table 3:

	ndf	sigma	Distribution	Bias	SD	CovRate	CovLength
1	20	1	Normal	-0.012	0.337	0.954	1.392
2	20	1	LogNormal	-0.001	0.326	0.957	1.390
3	20	2	Normal	0.004	0.524	0.955	2.228
4	20	2	LogNormal	0.005	0.532	0.946	2.217
5	200	1	Normal	-0.004	0.102	0.949	0.396
6	200	1	LogNormal	-0.0003	0.099	0.955	0.396
7	200	2	Normal	0.001	0.165	0.943	0.625
8	200	2	LogNormal	-0.003	0.158	0.951	0.626
9	2,000	1	Normal	0.001	0.032	0.955	0.124
10	2,000	1	LogNormal	-0.0004	0.031	0.943	0.124
11	2,000	2	Normal	-0.001	0.049	0.956	0.196
12	2,000	2	LogNormal	-0.001	0.048	0.958	0.196

As expected bias disappears as condition 1 is removed. The another anomalies can be explained as follows: Insert math here

3a) Yes, it satisfies the two conditions required for an IV.

1) Z is not correlated with Y other than through X_1
2) Z is correlated relatively strongly with X_1

We know Z is not correlated with Y as it is generated independantly.

3b)

```
dataQ3 = regMCloop$new(nrange=c(20,200,2000),sigmarange=c(1,2),Urange=c("norm","lognorm"))
dataQ3$loop(b3=1,IV=TRUE)
```

Asymptotic convergence is slower but bias is reduced eventually. Standard deviations are higher due to higher presence of multicollinearity and use of IV estimates.

4)

Table 4:

	ndf	sigma	Distribution	Bias	SD	CovRate	CovLength
1	20	1	Normal	-0.265	2.868	0.949	11.274
2	20	1	LogNormal	-0.208	3.293	0.957	14.570
3	20	2	Normal	-0.190	2.635	0.961	10.513
4	20	2	LogNormal	-0.103	2.139	0.965	7.268
5	200	1	Normal	-0.003	0.180	0.945	0.694
6	200	1	LogNormal	-0.009	0.174	0.947	0.691
7	200	2	Normal	0.011	0.279	0.956	1.090
8	200	2	LogNormal	-0.013	0.281	0.949	1.099
9	2,000	1	Normal	-0.001	0.056	0.944	0.215
10	2,000	1	LogNormal	-0.0002	0.056	0.944	0.215
11	2,000	2	Normal	0.001	0.085	0.954	0.340
12	2,000	2	LogNormal	0.002	0.087	0.956	0.340

```
dataQ4 = regMCloop$new(nrange=200,sigmarange=1,Urange="norm",a1range=c(0.8,seq(0.6,0,by=-0.1)))
dataQ4$loop(b3=1,IV=TRUE)
```

Table 5:

	ndf	sigma	Distribution	Bias	SD	CovRate	CovLength
1	200	1	Normal	-0.013	0.213	0.954	0.824
2	200	1	Normal	-0.017	0.279	0.941	1.062
3	200	1	Normal	-0.023	0.329	0.954	1.297
4	200	1	Normal	-0.284	7.284	0.957	59.131
5	200	1	Normal	-0.168	1.951	0.951	8.340
6	200	1	Normal	0.070	10.446	0.951	206.402
7	200	1	Normal	-1.208	70.101	0.969	15,651.010
8	200	1	Normal	0.696	25.970	0.983	1,985.125

As the correlation between Z and X decreases, Z becomes takes on more and more of the characteristics of a weak instrument until it becomes an invalid instrument at 0. Bias increases accordingly, and the convergence of the estimator is slower. Because convergence for every given sample size increases with the strength of the IV, standard deviation at every sample size is decreases accordingly. This has the follow over effect of increasing confidence interval lengths. Bias is also higher due to finite sample bias in Iv regression.

Obviously when α_1 becomes zero it is longer an instrument and therefore the IVreg method is invalid. However when the instrumental variable is invalid, the bias and standard deviation curiously drops. This can be explained for a number of reasons which were outlined in the previous assignment. We again cite Morgan and Winship. Firstly, IV estimates can always be estimated as sample covariances are never exactly equal to zero (cannot draw an infinite sample). Therefore even for invalid instruments, Iv estimates can be computed. In fact, estimators for the standard errors are constructed under the assumption that the IV is valid and thereby generate "artificially small standard errors" (Morgan and Winship, 2015).

5)

```
dataQ5 = regMCloop$new(nrange=200,sigmarange=1,Urange="norm",a2range=seq(0,10,by=2))
dataQ5$loop()
```

β_1 does not appear to change with α_2. At first this may seem surprising - common literature holds that multicollinearity increases standard errors.

Table 6:

	ndf	sigma	Distribution	Bias	SD	CovRate	CovLength
1	200	1	Normal	-0.0002	0.072	0.954	0.280
2	200	1	Normal	0.001	0.107	0.959	0.426
3	200	1	Normal	0.006	0.181	0.947	0.707
4	200	1	Normal	-0.006	0.265	0.938	1.010
5	200	1	Normal	0.011	0.335	0.953	1.321

However we see that this is only the case for β_2:

Table 7:

	ndf	sigma	Distribution	Bias	SD	CovRate	CovLength
1	200	1	Normal	-0.0002	0.072	0.955	0.281
2	200	1	Normal	0.001	0.107	0.950	0.428
3	200	1	Normal	0.006	0.181	0.958	0.708
4	200	1	Normal	-0.006	0.265	0.961	1.013
5	200	1	Normal	0.011	0.335	0.968	1.326

The intuitive reasoning can be explained by thinking of the uncorrelated model(a2=0). In this case we have n points, in p dimensional space. ie. the column space of X. We seek to find the linear combination of these vectors which best explain y. A more precise way of defining this is we wish to find a project the data onto a hyperplane H such that the projection has smallest distance with y. In this case, its a little simpler - we have a column space with rank 2.

In the correlated case, one of the columns is nearly linearly dependant on the other.

eg.

$$X_{1,new} = Z_1 + 5X_2 + U$$

In this case:

$$X1_{new} \sim X1_{old} + 4X_2$$

Then:

$$\hat{Y} = \beta_1 X_{1,new} + \beta_2 X_2$$

$$\hat{Y} = \beta_1 X_{1,old} + \beta_{1,old} 4X_2 + \beta_2 X_2$$

If we set

$$\beta_{2,new} = 1 - 4\beta_{1,old}$$

Then our estimate for $\hat{Y}$ should be similar to our old one whilst keeping β_1 the same. Then β_1 would only vary in so much as X_1 would vary normally. However as β_2 has to compensate for the increasing value of α_2, any variation in X_2 is magnified by the value of α_2. Therefore as α_2 increases, β_2 continuously changes and the standard deviation also increases.

Ths reason why the old Y_hat is roughly the same value as the new one is because no new dimensions have really been added to the column space - X_2 is simply added again and as a projection is a linear combination, this can be compensated for very precisely.