

Lección 5

Marcos Bujosa

16 de octubre de 2023

Índice

1. Simulación del ejemplo del precio de las viviendas con tres regresores	2
1.1. Tareas	2
1.2. Estructura del guión	3
2. Simulación del ejemplo del precio de las viviendas con tres regresores que cambian	4
2.1. Estructura del guión	4
3. Simulación del ejemplo del precio de las viviendas con tres regresores que cambian pero U se mantiene	5
3.1. Estructura del guión	5
4. Simulación del ejemplo del precio de las viviendas con tres regresores ortogonales	6
4.1. Estructura del guión	6
5. Simulación del ejemplo del precio de las viviendas forzando a que la muestra de U cumpla los supuestos	8
5.1. Estructura del guión	8
6. Efectos del incumplimiento de algunos supuestos (perturbaciones sin esperanza nula)	10
6.1. Estructura del guión	10
7. Efectos del incumplimiento de algunos supuestos (perturbaciones no ortogonales a S)	11
7.1. Estructura del guión	11

1. Simulación del ejemplo del precio de las viviendas con tres regresores

Guión: [BucleSimuladorEjPvivienda.inp](#)

En este ejercicio con [Gretl](#) usaremos un bucle para generar muchas veces datos simulados con los que realizar muejas regresiones MCO de un mismo modelo y así observar el comportamiento del estimador MCO.

1.1. Tareas

Fijamos el tamaño muestral Simulamos series de datos de 500 observaciones. Indicamos la opción `--preserve` para que Gretl mantenga en memoria el scalar que fija el tamaño muestral `TM`

```
scalar TM = 500
nulldata --preserve TM
```

Simulamos S y D Generamos dos variables que serán los regresores no constantes: **S** con distribución uniforme (35, 120); **D** con distribución Chi cuadrado con 5 grados de libertad que vamos a multiplicar por 3

```
series S = randgen(U, 35, 120)
series D = randgen(X, 5) * 3
```

Parte sistemática La parte *sistemática* del modelo es $y_s = 100(1) + 3s - 130d$

```
series YS = 100 + 3*S - 130*D
```

Esperanza y desviación típica de U En cada iteración generaremos una nueva perturbación **U** con media cero $\mu=0$ y desviación típica $\sigma=100$ (por tanto la longitud o norma de **U** será 100 en cada iteración)

```
scalar mu = 0
scalar sigma = 100
```

Generación del vector de perturbaciones U

```
series U = randgen(N, mu ,sigma )
```

Comprobación de si U es ortogonal a los regresores S y D Dos vectores son perpendiculares si la media del producto Hadamard (componente a componente) entre ellos es nula. Si uno de los vectores tiene media nula, entonces ambos vectores son ortogonales si su correlación es nula. Vamos a calcular estos estadísticos

```
scalar mU = mean(U)
scalar mUS = mean(U*S)
scalar mUD = mean(U*D)
scalar cUS = corr(U,S)
scalar cUD = corr(U,D)
```

Parámetros beta estimados En cada iteración guardamos los betas estimados

```
scalar b1 = $coeff(const)
scalar b2 = $coeff(S)
scalar b3 = $coeff(D)
```

Visualización de resultados y guardado en disco Para estudiar los resultados, mostramos los estadísticos calculados y los guardamos en el fichero `coef.gdt`.

```
print mU mUS mUD cUS cUD b1 b2 b3
store "@workdir/coef.gdt" mU mUS mUD cUS cUD b1 b2 b3
```

Análisis gráfico de los resultados Una vez finalizado el bucle, pedimos a Gretl que abra el fichero de resultados guardado (se abrirá nueva sesión) y generamos unos diagramas donde vamos a ver cómo perturbaciones más alejadas de la perpendicularidad con los regresores afectan más a las estimaciones del parámetro beta asociado al regresor correspondiente.

```
open "@workdir/coef.gdt"
MediaU_b1 <- gnuplot b1 mU --output="display"
CorrUS_b2 <- gnuplot b2 cUS --output="display"
CorrUD_b3 <- gnuplot b3 cUD --output="display"
CorrUD_b2 <- gnuplot b2 cUD --output="display"
CorrUS_b3 <- gnuplot b3 cUS --output="display"
```

1.2. Estructura del guión

```
<<Fijamos el tamaño muestral>>
<<Simulamos S y D>>
<<Parte sistemática>>
<<Esperanza y desviación típica de U>>
loop 5000 --progressive --quiet
  <<Generación del vector de perturbaciones U>>
  series Y = YS + U
  <<Comprobación de si U es ortogonal a los regresores S y D>>
  ols Y 0 S D
  <<Parámetros beta estimados>>
  <<Visualización de resultados y guardado en disco>>
endloop
<<Análisis gráfico de los resultados>>
```

Código completo de la práctica BucleSimuladorEjPvivienda.inp

```
scalar TM = 500
nulldata --preserve TM
series S = randgen(U, 35, 120)
series D = randgen(X, 5) * 3
series YS = 100 + 3*S - 130*D
scalar mu = 0
scalar sigma = 100
loop 5000 --progressive --quiet
  series U = randgen(N, mu, sigma)
  series Y = YS + U
  scalar mU = mean(U)
  scalar mUS = mean(U*S)
  scalar mUD = mean(U*D)
  scalar cUS = corr(U, S)
  scalar cUD = corr(U, D)
  ols Y 0 S D
  scalar b1 = $coeff(const)
  scalar b2 = $coeff(S)
  scalar b3 = $coeff(D)
  print mU mUS mUD cUS cUD b1 b2 b3
  store "@workdir/coef.gdt" mU mUS mUD cUS cUD b1 b2 b3
endloop
open "@workdir/coef.gdt"
MediaU_b1 <- gnuplot b1 mU --output="display"
CorrUS_b2 <- gnuplot b2 cUS --output="display"
CorrUD_b3 <- gnuplot b3 cUD --output="display"
CorrUD_b2 <- gnuplot b2 cUD --output="display"
CorrUS_b3 <- gnuplot b3 cUS --output="display"
```

2. Simulación del ejemplo del precio de las viviendas con tres regresores que cambian

Guión: [BucleSimuladorEjPvivienda2.inp](#)

En este ejercicio con [Gretl](#) repetimos lo anterior, pero en cada iteración también cambian los regresores.

2.1. Estructura del guión

```
<<Fijamos el tamaño muestral>>
<<Esperanza y desviación típica de U>>
loop 5000 --progressive --quiet
  <<Simulamos S y D>>
  <<Parte sistemática>>
  <<Generación del vector de perturbaciones U>>
  series Y = YS + U
  <<Comprobación de si U es ortogonal a los regresores S y D>>
  ols Y 0 S D
  <<Parámetros beta estimados>>
  <<Visualización de resultados y guardado en disco>>
endloop
<<Análisis gráfico de los resultados>>
```

Código completo de la práctica [BucleSimuladorEjPvivienda2.inp](#)

```
scalar TM = 500
nulldata --preserve TM
scalar mu = 0
scalar sigma = 100
loop 5000 --progressive --quiet
  series S = randgen(U, 35, 120)
  series D = randgen(X, 5) * 3
  series YS = 100 + 3*S - 130*D
  series U = randgen(N, mu, sigma)
  series Y = YS + U
  scalar mU = mean(U)
  scalar mUS = mean(U*S)
  scalar mUD = mean(U*D)
  scalar cUS = corr(U, S)
  scalar cUD = corr(U, D)
  ols Y 0 S D
  scalar b1 = $coeff(const)
  scalar b2 = $coeff(S)
  scalar b3 = $coeff(D)
  print mU mUS mUD cUS cUD b1 b2 b3
  store "@workdir/coef.gdt" mU mUS mUD cUS cUD b1 b2 b3
endloop
open "@workdir/coef.gdt"
MediaU_b1 <- gnuplot b1 mU --output="display"
CorrUS_b2 <- gnuplot b2 cUS --output="display"
CorrUD_b3 <- gnuplot b3 cUD --output="display"
CorrUD_b2 <- gnuplot b2 cUD --output="display"
CorrUS_b3 <- gnuplot b3 cUS --output="display"
```

3. Simulación del ejemplo del precio de las viviendas con tres regresores que cambian pero U se mantiene

Guión: [BucleSimuladorEjPvivienda3.inp](#)

En este ejercicio con [Gretl](#) repetimos lo anterior, pero en cada iteración solo cambian los regresores.

3.1. Estructura del guión

```
<<Fijamos el tamaño muestral>>
<<Esperanza y desviación típica de U>>
<<Generación del vector de perturbaciones U>>
loop 5000 --progressive --quiet
  <<Simulamos S y D>>
  <<Parte sistemática>>
  series Y = YS + U
  <<Comprobación de si U es ortogonal a los regresores S y D>>
  ols Y 0 S D
  <<Parámetros beta estimados>>
  <<Visualización de resultados y guardado en disco>>
endloop
<<Análisis gráfico de los resultados>>
```

Código completo de la práctica [BucleSimuladorEjPvivienda3.inp](#)

```
scalar TM = 500
nulldata --preserve TM
scalar mu = 0
scalar sigma = 100
series U = randgen(N, mu, sigma)
loop 5000 --progressive --quiet
  series S = randgen(U, 35, 120)
  series D = randgen(X, 5) * 3
  series YS = 100 + 3*S - 130*D
  series Y = YS + U
  scalar mU = mean(U)
  scalar mUS = mean(U*S)
  scalar mUD = mean(U*D)
  scalar cUS = corr(U,S)
  scalar cUD = corr(U,D)
  ols Y 0 S D
  scalar b1 = $coeff(const)
  scalar b2 = $coeff(S)
  scalar b3 = $coeff(D)
  print mU mUS mUD cUS cUD b1 b2 b3
  store "@workdir/coef.gdt" mU mUS mUD cUS cUD b1 b2 b3
endloop
open "@workdir/coef.gdt"
MediaU_b1 <- gnuplot b1 mU --output="display"
CorrUS_b2 <- gnuplot b2 cUS --output="display"
CorrUD_b3 <- gnuplot b3 cUD --output="display"
CorrUD_b2 <- gnuplot b2 cUD --output="display"
CorrUS_b3 <- gnuplot b3 cUS --output="display"
```

4. Simulación del ejemplo del precio de las viviendas con tres regresores ortogonales

Guión: [BucleSimuladorEjPvivienda4.inp](#)

En este ejercicio con [Gretl](#) repetimos lo anterior, pero en cada iteración cambia la perturbación y cambian los regresores, pero los regresores son ortogonales entre si.

Simulamos S y D ortogonales Generamos dos variables que serán los regresores no constantes: S con distribución uniforme (35, 120); D con distribución Chi cuadrado con 5 grados de libertad que vamos a multiplicar por 3; pero luego ortogonalizamos los regresores.

```
series S0 = randgen(U, 35, 120)
series D0 = randgen(X, 5) * 3
ols S0 0
series S = $uhat
ols D0 0 S
series D = $uhat
```

4.1. Estructura del guión

```
<<Fijamos el tamaño muestral>>
<<Esperanza y desviación típica de U>>
loop 5000 --progressive --quiet
  <<Simulamos S y D ortogonales>>
  <<Parte sistemática>>
  <<Generación del vector de perturbaciones U>>
  series Y = YS + U
  <<Comprobación de si U es ortogonal a los regresores S y D>>
  ols Y 0 S D
  <<Parámetros beta estimados>>
  <<Visualización de resultados y guardado en disco>>
endloop
<<Análisis gráfico de los resultados>>
```

Código completo de la práctica BucleSimuladorEjPvivienda4.inp

```
scalar TM      = 500
nulldata --preserve TM
scalar mu      = 0
scalar sigma   = 100
loop 5000 --progressive --quiet
  series S0 = randgen(U, 35, 120)
  series D0 = randgen(X, 5) * 3
  ols S0 0
  series S = $uhat
  ols D0 0 S
  series D = $uhat
  series YS = 100 + 3*S - 130*D
  series U = randgen(N, mu ,sigma )
  series Y  = YS + U
  scalar mU  = mean(U)
  scalar mUS = mean(U*S)
  scalar mUD = mean(U*D)
  scalar cUS = corr(U,S)
  scalar cUD = corr(U,D)
  ols Y 0 S D
  scalar b1 = $coeff(const)
  scalar b2 = $coeff(S)
  scalar b3 = $coeff(D)
  print mU mUS mUD cUS cUD b1 b2 b3
  store "@workdir/coef.gdt" mU mUS mUD cUS cUD b1 b2 b3
endloop
open "@workdir/coef.gdt"
MediaU_b1 <- gnuplot b1 mU --output="display"
CorrUS_b2 <- gnuplot b2 cUS --output="display"
CorrUD_b3 <- gnuplot b3 cUD --output="display"
CorrUD_b2 <- gnuplot b2 cUD --output="display"
CorrUS_b3 <- gnuplot b3 cUS --output="display"
```

5. Simulación del ejemplo del precio de las viviendas forzando a que la muestra de U cumpla los supuestos

Guión: [BucleSimuladorEjPvivienda5.inp](#)

En este ejercicio con [Gretl](#) repetimos lo anterior, pero en cada iteración también cambian los regresores.

Generación del vector de perturbaciones U forzando que la muestra cumpla los supuestos

```
series U = randgen(N, 0, 1)
series U = U - mean(U)
series U = U / sqrt( var(U)*(TM-1)/TM )
series U = mu + sigma * U
ols U 0 S D
series U = $uhat
```

5.1. Estructura del guión

```
<<Fijamos el tamaño muestral>>
<<Esperanza y desviación típica de U>>
loop 5000 --progressive --quiet
  <<Simulamos S y D>>
  <<Parte sistemática>>
  <<Generación del vector de perturbaciones U forzando que la muestra cumpla los supuestos>>
  series Y = YS + U
  <<Comprobación de si U es ortogonal a los regresores S y D>>
  ols Y 0 S D
  <<Parámetros beta estimados>>
  <<Visualización de resultados y guardado en disco>>
endloop
<<Análisis gráfico de los resultados>>
```


Código completo de la práctica BucleSimuladorEjPvivienda5.inp

```
scalar TM = 500
nulldata --preserve TM
scalar mu = 0
scalar sigma = 100
loop 5000 --progressive --quiet
  series S = randgen(U, 35, 120)
  series D = randgen(X, 5) * 3
  series YS = 100 + 3*S - 130*D
  series U = randgen(N, 0, 1)
  series U = U - mean(U)
  series U = U / sqrt( var(U)*(TM-1)/TM )
  series U = mu + sigma * U
  ols U 0 S D
  series U = $uhat
  series Y = YS + U
  scalar mU = mean(U)
  scalar mUS = mean(U*S)
  scalar mUD = mean(U*D)
  scalar cUS = corr(U,S)
  scalar cUD = corr(U,D)
  ols Y 0 S D
  scalar b1 = $coeff(const)
  scalar b2 = $coeff(S)
  scalar b3 = $coeff(D)
  print mU mUS mUD cUS cUD b1 b2 b3
  store "@workdir/coef.gdt" mU mUS mUD cUS cUD b1 b2 b3
endloop
open "@workdir/coef.gdt"
MediaU_b1 <- gnuplot b1 mU --output="display"
CorrUS_b2 <- gnuplot b2 cUS --output="display"
CorrUD_b3 <- gnuplot b3 cUD --output="display"
CorrUD_b2 <- gnuplot b2 cUD --output="display"
CorrUS_b3 <- gnuplot b3 cUS --output="display"
```

6. Efectos del incumplimiento de algunos supuestos (perturbaciones sin esperanza nula)

Guión: [BucleSimuladorEjPvivienda6.inp](#)

Vamos a ver cómo afecta a las estimaciones simular modelos que incumplen algunos de los supuestos. Por ejemplo, ¿qué pasa si las perturbaciones tienen esperanza no nula?

6.1. Estructura del guión

```
<<Fijamos el tamaño muestral>>
scalar mu      = 1000
scalar sigma = 100
loop 5000 --progressive --quiet
    <<Simulamos S y D>>
    <<Parte sistemática>>
    <<Generación del vector de perturbaciones U>>
    series Y = YS + U
    <<Comprobación de si U es ortogonal a los regresores S y D>>
    ols Y 0 S D
    <<Parámetros beta estimados>>
    <<Visualización de resultados y guardado en disco>>
endloop
<<Análisis gráfico de los resultados>>
```

Código completo de la práctica [BucleSimuladorEjPvivienda6.inp](#)

```
scalar TM = 500
nulldata --preserve TM
scalar mu = 1000
scalar sigma = 100
loop 5000 --progressive --quiet
    series S = randgen(U, 35, 120)
    series D = randgen(X, 5) * 3
    series YS = 100 + 3*S - 130*D
    series U = randgen(N, mu, sigma)
    series Y = YS + U
    scalar mU = mean(U)
    scalar mUS = mean(U*S)
    scalar mUD = mean(U*D)
    scalar cUS = corr(U,S)
    scalar cUD = corr(U,D)
    ols Y 0 S D
    scalar b1 = $coeff(const)
    scalar b2 = $coeff(S)
    scalar b3 = $coeff(D)
    print mU mUS mUD cUS cUD b1 b2 b3
    store "@workdir/coef.gdt" mU mUS mUD cUS cUD b1 b2 b3
endloop
open "@workdir/coef.gdt"
MediaU_b1 <- gnuplot b1 mU --output="display"
CorrUS_b2 <- gnuplot b2 cUS --output="display"
CorrUD_b3 <- gnuplot b3 cUD --output="display"
CorrUD_b2 <- gnuplot b2 cUD --output="display"
CorrUS_b3 <- gnuplot b3 cUS --output="display"
```

7. Efectos del incumplimiento de algunos supuestos (perturbaciones no ortogonales a S)

Guión: [BucleSimuladorEjPvivienda7.inp](#)

Vamos a ver cómo afecta a las estimaciones simular modelos que incumplen algunos de los supuestos. Por ejemplo, ¿qué pasa si las perturbaciones no son ortogonales a uno o más regresores no constantes?

Definición de la matriz para generar correlación entre U y S Definimos la matriz con la que generar perturbaciones correladas con los regresores (*No recuerdo de donde saqué este modo de hacerlo*).

```
matrix C = {1 , 0, 0; 0 , 1, 0; -10, 0, 1}
#matrix C = {1 , 0, 0; 0 , 1, 0; 0, 100, 1}
```

Generación del vector de perturbaciones U correladas con los regresores (*No recuerdo de donde saqué este modo de hacerlo*).

```
series U0 = randgen(N, 0, 1)
matrix Z = {S,D,U0}
Z *= C' # note: use the transpose '
series U = Z[,3] # generamos las perturbaciones
```

7.1. Estructura del guión

```
<<Fijamos el tamaño muestral>>
<<Esperanza y desviación típica de U>>
<<Definición de la matriz para generar correlación entre U y S>>
loop 5000 --progressive --quiet
  <<Simulamos S y D>>
  <<Parte sistemática>>
  <<Generación del vector de perturbaciones U correladas con los regresores>>
  series Y = YS + U
  <<Comprobación de si U es ortogonal a los regresores S y D>>
  ols Y 0 S D
  <<Parámetros beta estimados>>
  <<Visualización de resultados y guardado en disco>>
endloop
<<Análisis gráfico de los resultados>>
```

Código completo de la práctica BucleSimuladorEjPvivienda7.inp

```
scalar TM = 500
nulldata --preserve TM
scalar mu = 0
scalar sigma = 100
matrix C = {1, 0, 0; 0, 1, 0; -10, 0, 1}
#matrix C = 1, 0, 0; 0, 1, 0; 0, 100, 1
loop 5000 --progressive --quiet
  series S = randgen(U, 35, 120)
  series D = randgen(X, 5) * 3
  series YS = 100 + 3*S - 130*D
  series U0 = randgen(N, 0, 1)
  matrix Z = {S,D,U0}
  Z *= C' # note: use the transpose '
  series U = Z[,3] # generamos las perturbaciones
  series Y = YS + U
  scalar mU = mean(U)
  scalar mUS = mean(U*S)
  scalar mUD = mean(U*D)
  scalar cUS = corr(U,S)
  scalar cUD = corr(U,D)
  ols Y 0 S D
  scalar b1 = $coeff(const)
  scalar b2 = $coeff(S)
  scalar b3 = $coeff(D)
  print mU mUS mUD cUS cUD b1 b2 b3
  store "@workdir/coef.gdt" mU mUS mUD cUS cUD b1 b2 b3
endloop
open "@workdir/coef.gdt"
MediaU_b1 <- gnuplot b1 mU --output="display"
CorrUS_b2 <- gnuplot b2 cUS --output="display"
CorrUD_b3 <- gnuplot b3 cUD --output="display"
CorrUD_b2 <- gnuplot b2 cUD --output="display"
CorrUS_b3 <- gnuplot b3 cUS --output="display"
```