ANEXO

El presente anexo contiene dos ejemplos de los programas diseñados para realizar los experimentos de Monte Carlo mediante el software econométrico Gretl.

1. GUIÓN DE INSTRUCCIONES MODELO M1

```
# Calcula el tamaño y la potencia  de BP y White modelo M1 caso A-1

#numero de datos y semilla de números aleatorios
nulldata 1000
set seed 123

# Fijar los valores del experimento
         scalar b1=1
         scalar b2=1
         scalar nsimul = 1000
         vtm=50|100|1000
         vv=0|1|2|3|4|5

         salida=zeros(1,4)


loop aa = 1..3
 loop bb = 1..6
        scalar tm=vtm[aa]
        v=vv[bb]


        nobs=tm

        print nobs

        smpl 1 nobs
        genr nbp=0
        genr nw=0
        genr x = 10 * uniform()

# Abrir un bucle "progresivo" y repetir 1000 veces
        loop nsimul --progressive
            genr s=   (10+v*x)^(1/2)
            genr uu=normal()
            genr uv=uu*s

# generar la variable dependiente
            genr y = b1+b2*x + uv

# ejecutar regression MCO
            ols y const x --quiet

# guarder el cuadrado de los residuos
            series uh2=( $uhat)^2
            genr ebp=uh2/($ess/$nobs)

# ejecutar regression Breusch-Pagan
         ols ebp const x --quiet
            scalar st=$ess/(1-$rsq)
            scalar se=st-$ess
```

```
            scalar bp= se/2
            print bp

# ejecutar regresión White
            genr x2= x^2
            ols uh2 const x x2 --quiet
            scalar w=$nobs*$rsq

# calcular p-valor
            genr pbp = pvalue(X, 1, bp)

            genr pw = pvalue(X, 2, w)

            if pbp < 0.05
                nbp = nbp +1
            endif

            if pw < 0.05
                nw = nw +1
            endif

        endloop
        genr nsbp = nbp/nsimul
        genr nsw = nw/nsimul


        sal=tm~v~nsbp~nsw
        salida= salida|sal


    endloop

endloop

matrix salidaf=salida[2:rows(salida),]
colnames(salidaf, "tm v bp w")

# guardar los resultados en archivo
outfile simulacion_1_1.xls --write
print   salidaf
outfile --close
```

2. GUIÓN DE INSTRUCCIONES MODELO M2

```
# Calcula el tamaño y la potencia  de BP y White modelo M1 caso A-1

#numero de datos y semilla de números aleatorios
nulldata 1000
set seed 123

# Fijar los valores del experimento
         scalar b1=1
         scalar b2=1
         scalar b3=1
         scalar nsimul = 1000
         vtm=50|100|1000
         vv=0|1|2|3|4|5

         salida=zeros(1,6)


loop aa = 1..3
 loop bb = 1..6
        scalar tm=vtm[aa]
        v=vv[bb]


        nobs=tm

        print nobs

        smpl 1 nobs
        genr nbp1=0
        genr nbp2=0
        genr nbp3=0
        genr nw=0
        genr x2 = 10 * uniform()
        genr x3 = 10 * uniform()

# Abrir un bucle "progresivo" y repetir 1000 veces
        loop nsimul --progressive
            genr s=   (10+v*x2)^(1/2)
            genr uu=normal()
            genr uv=uu*s

# generar la variable dependiente
            genr y = b1+b2*x2+ b3*x3 + uv

# ejecutar regression MCO
            ols y const x2 x3 --quiet

# guardar el cuadrado de los residuos
            series uh2=( $uhat)^2
            genr ebp1=uh2/($ess/$nobs)
            genr ebp2=uh2/($ess/$nobs)
            genr ebp3=uh2/($ess/$nobs)

# ejecutar regression Breusch-Pagan 1
         ols ebp1 const x2 --quiet
            scalar st1=$ess/(1-$rsq)
            scalar se1=st1-$ess
            scalar bp1= se1/2
            print bp1
```

```
 # ejecutar regression Breusch-Pagan 2
         ols ebp2 const x3 --quiet
            scalar st2=$ess/(1-$rsq)
            scalar se2=st1-$ess
            scalar bp2= se2/2
            print bp2


   # ejecutar regression Breusch-Pagan 3
         ols ebp3 const x2 x3 --quiet
            scalar st3=$ess/(1-$rsq)
            scalar se3=st3-$ess
            scalar bp3= se3/2
            print bp3


   # ejecutar regresión White
            genr x4= x2^2
            genr x5= x3^2
            genr x6= x2*x3
        ols uh2 const x2 x3 x4 x5 x6 --quiet
        scalar w=$nobs*$rsq

   # calcular p-valor
         genr pbp1 = pvalue(X, 1, bp1)
         genr pbp2 = pvalue(X, 1, bp2)
         genr pbp3 = pvalue(X, 2, bp3)
         genr pw = pvalue(X, 4, w)

         if pbp1 < 0.05
             nbp1 = nbp1 +1
             endif

         if pbp2 < 0.05
                  nbp2 = nbp2 +1
              endif

         if pbp3 < 0.05
             nbp3 = nbp3 +1
         endif

         if pw < 0.05
             nw = nw +1
         endif

      endloop
      genr nsbp1= nbp1/nsimul
      genr nsbp2= nbp2/nsimul
       genr nsbp3= nbp3/nsimul
       genr nsw = nw/nsimul


        sal=tm~v~nsbp1~nsbp2~nsbp3~nsw
       salida= salida|sal


    endloop

endloop

matrix salidaf=salida[2:rows(salida),]
```

```
colnames(salidaf, "tm v bp1 bp2 bp3 w")


   # guardar los resultados en archivo
   outfile simulacion_2_1_1.xls --write
   print   salidaf
   outfile --close
```