

# Lección 5.D - Simulación de procesos AR(2) en la región de estacionariedad

Marcos Bujosa

## 1. Objetivo de la práctica

Guión: [P-L05-D-simulacion-procesos-AR.inp](#)

### Objetivo

1. Observar la ACF y PACF de distintos modelos AR(2) en las distintas regiones del triángulo de estacionariedad.

### Requerimientos previos

Programe o recupere de una práctica anterior una función que simule procesos AR( $q$ )

```
function series SimuladorAR(matrix phi)
    # SimuladorAR(phi) simula un proceso AR(p),
    # donde phi es el polinomio AR y p es su grado.
    scalar p = cols(phi)
    series U = normal(0,1)
    series Y = 0
    setinfo Y --description="Serie simulada"
    loop i = (p+1)..$nobs
        scalar comb_pasado_Yt = 0
        scalar perturbacion = U[i]
        loop j = 2..p
            comb_pasado_Yt += -phi[1,j] * Y[i-j+1]  # expresión abreviada
        endloop
        Y[i] = comb_pasado_Yt + perturbacion
    endloop
    return Y
end function
```

Para que se observe bien la estructura de las ACF y PACF estimadas, establezca un tamaño de muestra suficientemente grande.

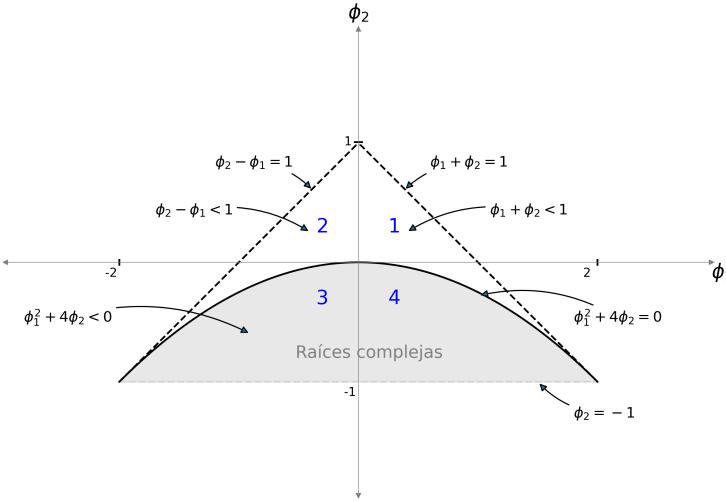
```
# establecemos la muestra
nulldata 3500
setobs 12 1900:01 --time-series
```

Recuerde cómo usar la función

```
scalar phi1 = 0
scalar phi2 = 0.8
series X = SimuladorAR( {1, -phi1, -phi2} )
figura <- corrgm X 12
```

## 2. Actividad 1 - Probando pares de valores en distintas regiones de invertibilidad

Asigne varios pares de valores  $\phi_1$  y  $\phi_2$  que pertenezcan a cada una de las regiones indicadas en la figura y explore cómo se comportan la AFC y PACF en cada caso.

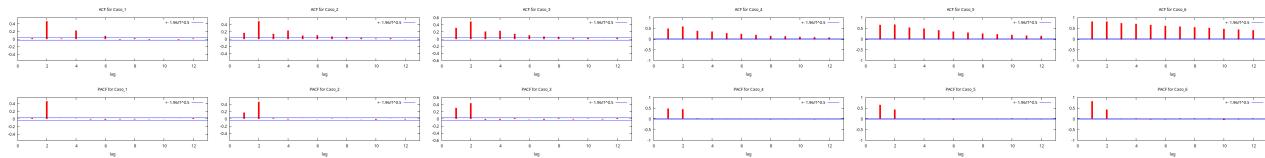


$$\rho_1 = \frac{-\phi_1}{1 - \phi_2}; \quad \rho_2 = \frac{\phi_1^2}{1 - \phi_2} + \phi_2; \quad \rho_k = \phi_1 \rho_{k-1} + \phi_2 \rho_{k-2}; \quad \pi_1 = \rho_1; \quad \pi_2 = \phi_2; \quad \pi_k = 0, \quad k \geq 3$$

### 2.1. Zona 1

Correlogramas en zona 3

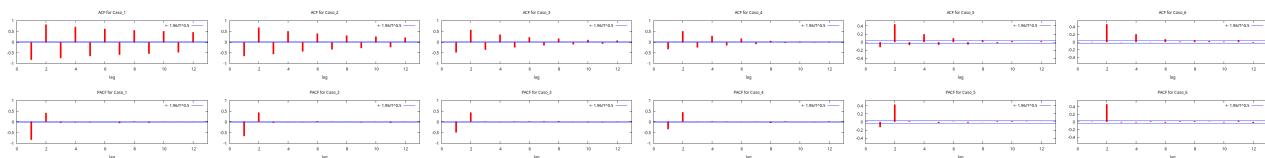
- $\phi_2 = 0,45$
- $\phi_1 \approx 0, \quad 0,1, \quad 0,2, \quad 0,3, \quad 0,4, \quad 0,45.$



### 2.2. Zona 2

Correlogramas en zona 3

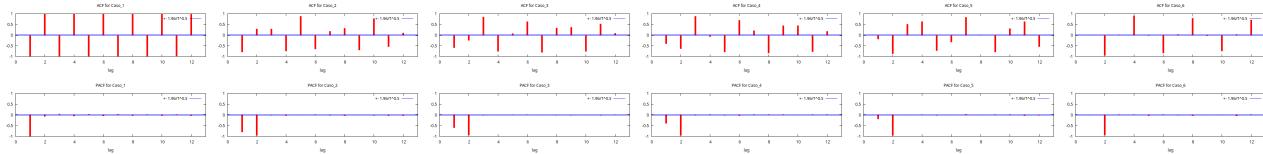
- $\phi_2 = -0,45$
- $\phi_1 \approx -0,45, \quad -0,4, \quad -0,3, \quad -0,2, \quad -0,1, \quad 0.$



## 2.3. Zona 3

Correlogramas en zona 3

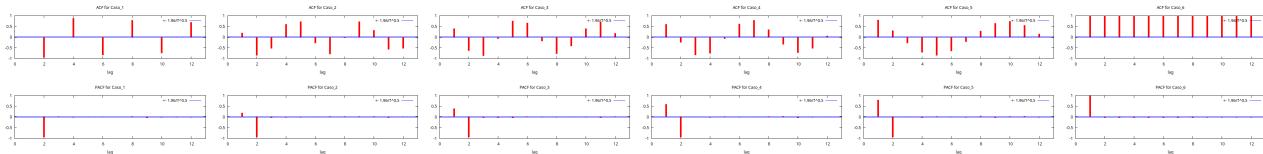
- $\phi_2 = -0,95$
- $\phi_1 \approx -1,95, -1,6, -1,2, -0,8, -0,4, 0.$



## 2.4. Zona 4

Correlogramas en zona 4

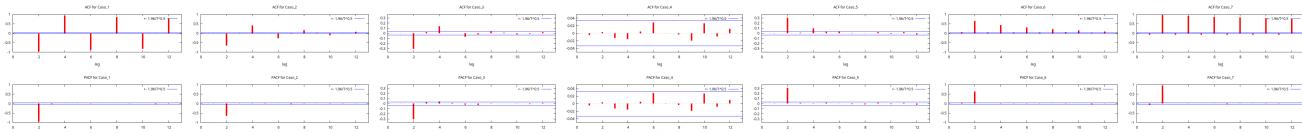
- $\phi_2 = -0,95$
- $\phi_1 \approx 0, 0,4, 0,8, 1,2, 1,6, 1,95.$



## 2.5. Eje vertical

Correlogramas correspondientes a puntos sobre el eje vertical

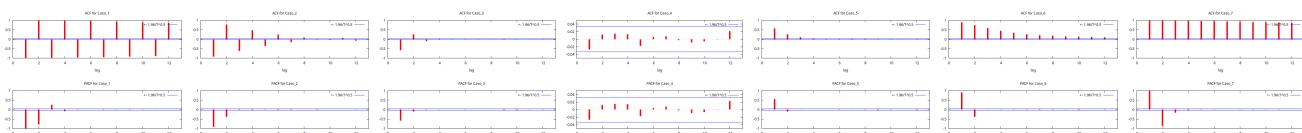
- $\phi_1 = 0$
- $\phi_2 \approx -0,9, -0,6, -0,3, 0, 0,3, 0,6, 0,9.$



## 2.6. Parábola

Correlogramas correspondientes a puntos sobre la parábola

- $\phi_1 \approx -1,9, -1,3, -0,6, 0, 0,6, 1,3, 1,9.$
- $\phi_2 = \frac{-\phi_1^2}{4}$



## Código completo de la práctica

```
# Los dos primeros comandos son necesarios para que Gretl guarde los resultados de la práctica en el directorio de trabajo
# al ejecutar lo siguiente desde un terminal (use los nombres y ruta que correspondan)
#
# DIRECTORIO="Nombre_Directorio_trabajo" gretlcli -b ruta/nombre_fichero_de_la_practica.inp
#
# Si esto no le funciona en su sistema, comente las siguientes dos líneas y sitúese en el directorio de trabajo de gretl
# que corresponda (configure dicho directorio de trabajo desde la ventana principal de Gretl).

string directory = getenv("DIRECTORIO")
set workdir "@directory"

function series SimuladorAR(matrix phi)
    # SimuladorAR(phi) simula un proceso AR(p),
    # donde phi es el polinomio AR y p es su grado.
    scalar p = cols(phi)
    series U = normal(0,1)
    series Y = 0
    setinfo Y --description="Serie simulada"
    loop i = (p+1)..$nobs
        scalar comb_pasado_Yt = 0
        scalar perturbacion = U[i]
        loop j = 2..p
            comb_pasado_Yt += -phi[1,j] * Y[i-j+1]  # expresión abreviada
        endloop
        Y[i] = comb_pasado_Yt + perturbacion
    endloop
    return Y
end function

# establecemos la muestra
nulldata 3500
setobs 12 1900:01 --time-series

scalar phi1 = 0
scalar phi2 = 0.8
series X = SimuladorAR( {1, -phi1, -phi2} )
figura <- corrgm X 12

scalar phi2 = 0.45
n = 0
loop for (r=0; r<=1-phi2-0.04; r+=.09)
    n += 1
    scalar phi1 = r
    nombre = sprintf("Zona1-%d(%1.2f-%1.2f)", n, phi1, phi2)
    sname = sprintf("Caso_%d", n)
    series @sname = SimuladorAR( {1, -phi1, -phi2} )
    corrgm @sname 12 --plot=@nombre.png
endloop

scalar phi2 = 0.45
scalar n = 0
loop for (r=-phi2; r<=0.03; r+=.09)
    n += 1
    scalar phi1 = r
    nombre = sprintf("Zona2-%d(%1.2f-%1.2f)", n, phi1, phi2)
    sname = sprintf("Caso_%d", n)
    series @sname = SimuladorAR( {1, -phi1, -phi2} )
    corrgm @sname 12 --plot=@nombre.png
endloop

scalar phi2 = -0.95
scalar n = 0
loop for (r=phi2-1; r<=0.01; r+=.39)
    n += 1
    scalar phi1 = r
    nombre = sprintf("Zona3-%d(%1.2f-%1.2f)", n, phi1, phi2)
```

```

    sname = sprintf("Caso_%d", n)
    series @sname = SimuladorAR( {1, -phi1, -phi2} )
    corrgm @sname 12 --plot=@nombre.png
endloop

scalar phi2 = -0.95
scalar n = 0
loop for (r=0; r<=1-phi2+0.01; r+=.39)
    n += 1
    scalar phi1 = r
    nombre = sprintf("Zona4-%d(%1.2f-%1.2f)", n, phi1, phi2)
    sname = sprintf("Caso_%d", n)
    series @sname = SimuladorAR( {1, -phi1, -phi2} )
    corrgm @sname 12 --plot=@nombre.png
endloop

scalar phi1 = 0
scalar n = 0
loop for (r=-.96; r<=1; r+=0.32)
    n += 1
    scalar phi2 = r
    nombre = sprintf("EjeVertical-%d(%1.2f-%1.2f)", n, phi1, phi2)
    sname = sprintf("Caso_%d", n)
    series @sname = SimuladorAR( {1, -phi1, -phi2} )
    corrgm @sname 12 --plot=@nombre.png
endloop

scalar n = 0
loop for (r=-1.91; r<=2; r+=0.636)
    n += 1
    scalar phi1 = r
    scalar phi2 = -phi1*phi1/4
    nombre = sprintf("Parabola-%d(%1.2f-%1.2f)", n, phi1, phi2)
    sname = sprintf("Caso_%d", n)
    series @sname = SimuladorAR( {1, -phi1, -phi2} )
    corrgm @sname 12 --plot=@nombre.png
endloop

```