



Curso : Teoría Macroeconómica I-EAE320B-1

Profesor : Alexandre Janiak

Ayudantes: Bianca Hincapié y Pablo Vega

Fecha : 17 de abril de 2022

Estudiante : Valentina Andrade

Mail : vandrade@uc.cl

---

## Tarea N°1

# Problema 1 - Simulación de Montecarlo

```
clear;
close all;
% Global settings
warning('off'); % warnings about rank
figure('Renderer', 'painters', 'Position', [10 10 900 600]); % figures size
```

El propósito de este ejercicio es utilizar de manera eficiente ciertos comandos de Matlab tales como matrices, loops, while, plot y subplot. Se aplican diversos conocimientos econométricos y se han desarrollado algoritmos de la forma más eficientes posible. Las funciones trabajadas se pueden acceder aquí:

- xx.m
- plotxx.m
- random.m
- algoritmo.m
- algoritmo2.m

El modelo de estudio está dado por:

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \beta_3 \cdot x_3 + \beta_4 \cdot x_4 + \beta_5 \cdot x_5 + \beta_6 \cdot x_6 + \epsilon \quad (1)$$

## (a) Distribución de variables

### 0. Parámetros

```
r = 1000; % Muestra de tamaño 1000
c = 1;
x = rand(r,c);
x_i = randn(r,c);
```

1.  $x_1 \sim U(1, 8)$

```
x1 = x*(8-1)+1;
```

2.  $x_2 \sim N(3, 4)$

Se debe construir con una matriz de números aleatorios con distribución normal, con media 3 y desvío 4 (con randn)

```
x2 = x_i*4 + 3;
```

3.  $x_3 \sim X_{100}^2$

Chi cuadrado de 100 gl usando randn que genere una normal de media 0 y varianza 1

```
x3 = 100*(x_i.^2);
```

4.  $x_4 \sim (t \text{ student})_2$

t student 2 gl

```
x4a = 0;
for i = 1:2
    x4a = x4a + randn(r,c).^2;
    i = i+1;
end

x4 = x_i./sqrt(x4a/2);
```

5.  $x_5 \sim$  una mixtura tal que el 50% de las veces se comporta como  $x_2$  y el 50% de las veces como  $x_3$

Para generar esta distribución utilizaré mi variable aleatoria adicional (x) y a partir de ella seleccionaré una muestra donde con un 50% de probabilidad se recupere el valor de x2 o en caso contrario x3

```
x5 = zeros(r,c);

for i = 1:r
    if x (i,1) < 0.5
        x5(i,1) = x2(i, 1);
    else
        x5(i) = x3(i,1);
    end
    i = i+1;
end
```

6.  $x_6$  es un promedio entre  $x_4$  y  $x_5$  más un error normal con media 0 y desvío 0.1

X6: promedio de X4 y X5 mas un error normal con media cero y desvio 0.0001.

```
error = x_i* 0.1;
x6 = (x4+x5)/2 + error;
```

7.  $y \sim \exp(0.6)$

```
% --> 1-rand = exp(-lamda*x) --> log(1-rand) = (-lamda*x) -->
% log(1-rand)/-lamda = x
y = log(1-x)/(-0.6);
```

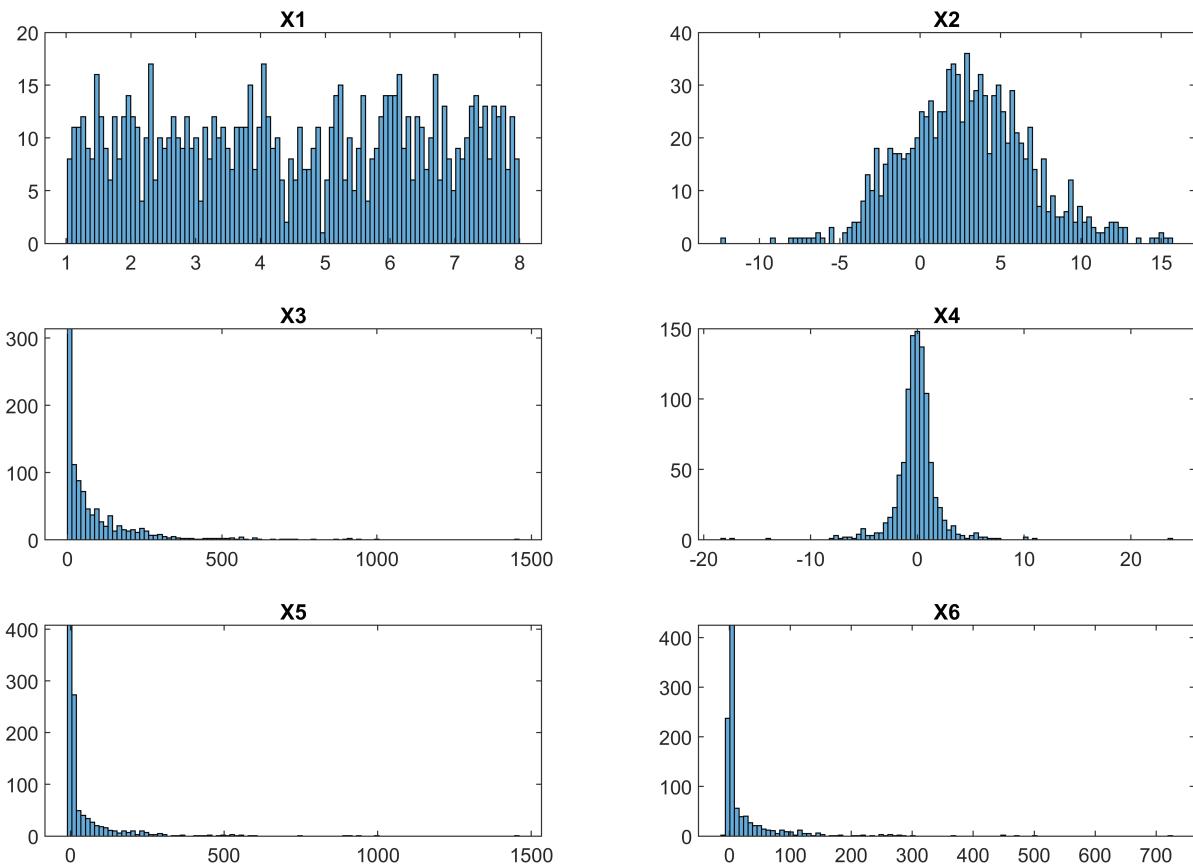
8.  $\epsilon \sim N(0, 1)$

```
epsilon = x_i;
```

Se ha creado una función xx.m que construye las distribuciones de las variables indicadas, y que recibe como input el número de observaciones. Con la función plotxx.m graficaremos estas distribuciones en un histograma

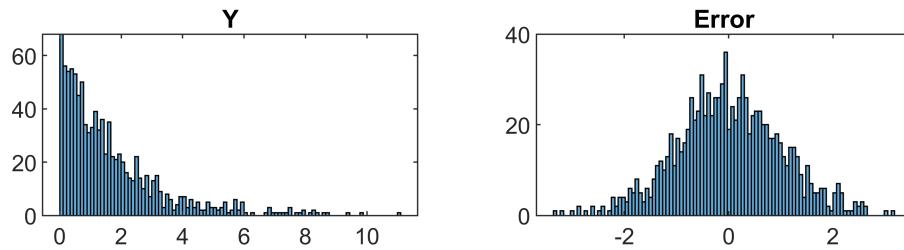
```
mm = xx(1000,1);
```

```
plotxx(mm, 1000,1)
```



Además también se muestra la distribución de **y** y el **error**

```
figure(2)
subplot(3,2,1)
histogram(y,100)
title('Y')
subplot(3,2,2)
histogram(epsilon,100)
title('Error')
hold off
```



(b) Genere 1000 muestras de tamaño n = 100 para todas la variables x

```
x_i = randn(100,1000)
```

```
x_i = 100×1000
 -0.2720    0.1553   -0.3770    1.0696   -0.5956   -0.1079    0.3474   -0.6334 ...
 -0.9596    0.5499   0.3887   -0.5196   -0.2508    0.4376    0.4582   -0.6959
 -1.5208    0.6736   -1.0119   -0.1829   -2.0685   -2.0307   -0.5910   -0.5256
  0.1852   -1.0349   -0.9492   -0.4233    0.7865   -0.4649    0.7339   1.5388
  1.1726   -0.3060    0.0071    0.1174    0.6798    0.3386   -1.0877   0.0605
 -0.2241   -0.8126   -0.3462   -0.3816    0.3767   -1.1032    1.1843   0.1701
 -0.0860    1.2470    0.3322   -0.5007    1.3763   -0.6115    0.6706   0.2726
  1.2747    0.6493    0.1550    0.0374   -0.4622   -0.7172   -0.4497   0.7486
  0.7079   -0.1880   -0.6242   -0.7194   -0.2868    3.5675   -0.0819   -1.3496
  0.3679    1.0939   -0.2602   -0.8267   -1.3888   -0.3930    1.2699    0.9138
 .
 :
```

```
x = rand(100,1000)
```

```
x = 100×1000
  0.2078    0.9048    0.3286    0.6572    0.8834    0.9197    0.9779    0.9924 ...
  0.9898    0.1602    0.8020    0.0652    0.8384    0.1964    0.2477    0.1391
  0.4238    0.5884    0.6714    0.6296    0.4409    0.9569    0.6822    0.5366
  0.4454    0.3916    0.3526    0.3638    0.0378    0.8326    0.9518    0.4757
  0.2470    0.5432    0.8325    0.3899    0.7834    0.3060    0.1718    0.6726
  0.1920    0.0787    0.6638    0.2432    0.9349    0.9174    0.4710    0.5426
  0.7278    0.8289    0.3817    0.6866    0.5631    0.5331    0.0569    0.3914
  0.6388    0.2568    0.2956    0.2729    0.6392    0.9221    0.2973    0.8296
  0.1987    0.5360    0.0640    0.2369    0.3808    0.2521    0.2520    0.6907
```

```

0.5544    0.3000    0.7020    0.8916    0.4196    0.9728    0.0785    0.4538
⋮

```

### (c ) Crear función

Cree la función random que recibe como inputs el número de muestras y el tamaño de cada muestra. Además, debe entregar como output una matriz que compile todas las variables  $x(1-6)$  junto con entregar un subplot que indique como distribuye cada una de las variables (histograma).

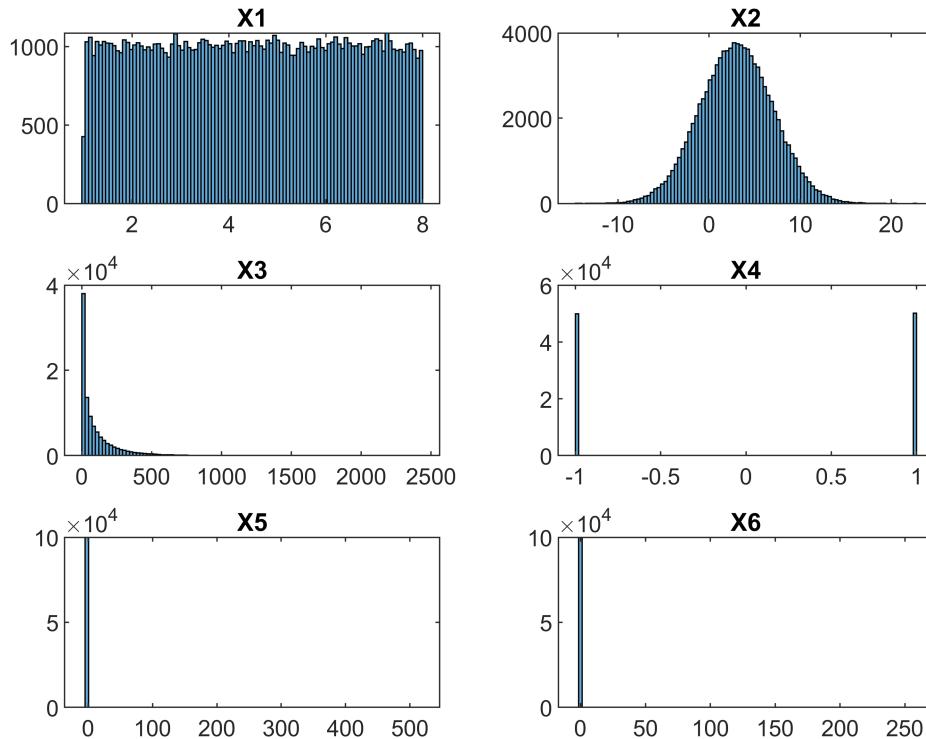
```
random(100,1000)
```

```

ans = 100×6000
103 ×
0.0013    0.0013    0.0045    0.0047    0.0055    0.0047    0.0021    0.0072 ...
0.0026    0.0069    0.0031    0.0060    0.0033    0.0018    0.0022    0.0015
0.0057    0.0044    0.0042    0.0046    0.0046    0.0064    0.0046    0.0058
0.0048    0.0028    0.0077    0.0059    0.0041    0.0043    0.0068    0.0027
0.0057    0.0066    0.0066    0.0059    0.0039    0.0033    0.0033    0.0051
0.0077    0.0018    0.0080    0.0061    0.0075    0.0047    0.0051    0.0029
0.0022    0.0048    0.0055    0.0019    0.0037    0.0048    0.0042    0.0076
0.0050    0.0041    0.0057    0.0059    0.0063    0.0039    0.0035    0.0065
0.0048    0.0057    0.0010    0.0016    0.0048    0.0069    0.0072    0.0025
0.0048    0.0020    0.0047    0.0044    0.0030    0.0077    0.0041    0.0016
⋮

```

```
X = random(100, 1000);
```



(d) Compute la media, mediana, mínimo, máximo, varianza y percentiles 25 y 75 de las muestras obtenidas.

La tabla se lee como que cada fila corresponde a los estadísticos descriptivos de cada  $x_n$

```
statistics = [ mean(X(:,1:1000:6000))' std(X(:,1:1000:6000))' min(X(:,1:1000:6000))' max(X(:,1:1000:6000))'
```

```
statistics = 6x7
4.7001    2.0812    1.1620    7.9631    4.8678    2.9692    6.5360
2.7870    3.9903   -6.1136   11.1098    2.7745    0.0640    5.6551
98.8051   119.6668    0.0002  519.1137   49.1089    7.0975  159.9057
-0.0600    1.0032   -1.0000    1.0000   -1.0000   -1.0000    1.0000
54.8542   100.3070   -5.3336   519.1137    6.3363    2.3476   51.2967
27.3918   50.1399   -3.3751  258.8290    3.7516    0.6585   26.2200
```

## OLS

Ahora, dado que se conoce la distribución de cada una de las variables  $x(1-6)$  y el vector  $y$ , aplicando la metodología OLS, estimar a los parámetros  $\beta_1-6$ .

Sabemos que para poder calcular los coeficientes de regresión debemos utilizar la matrix de los  $X_n$  para luego calcular  $\hat{\beta} = (X'X)^{-1}X'y$ .

Ahora bien, en **Matlab** es más rápido resolver como  $\hat{\beta} = (Xy)'$

(e) Compute el vector3 de estimadores  $\beta_{7 \times 1}$  para una muestra arbitraria.

```
mm = [ones(1000,1) mm]
```

```
mm = 1000x7
10^3 x
0.0010    0.0011   -0.0021    0.1631   -0.0036   -0.0021   -0.0030
0.0010    0.0023    0.0058    0.0497    0.0010    0.0058    0.0035
0.0010    0.0053    0.0111    0.4086    0.0019    0.4086    0.2054
0.0010    0.0067   -0.0007    0.0841   -0.0013    0.0841    0.0413
0.0010    0.0015    0.0022    0.0043   -0.0004    0.0022    0.0009
0.0010    0.0069   -0.0054    0.4362   -0.0024    0.4362    0.2167
0.0010    0.0038    0.0023    0.0029   -0.0010    0.0023    0.0006
0.0010    0.0020    0.0006    0.0369   -0.0007    0.0006   -0.0001
0.0010    0.0048    0.0033    0.0005    0.0001    0.0005    0.0003
0.0010    0.0048    0.0051    0.0265    0.0005    0.0265    0.0136
:
```

```
%beta_1 = inv(mm'*mm)*mm'*y
beta_1 = (mm\y)'
```

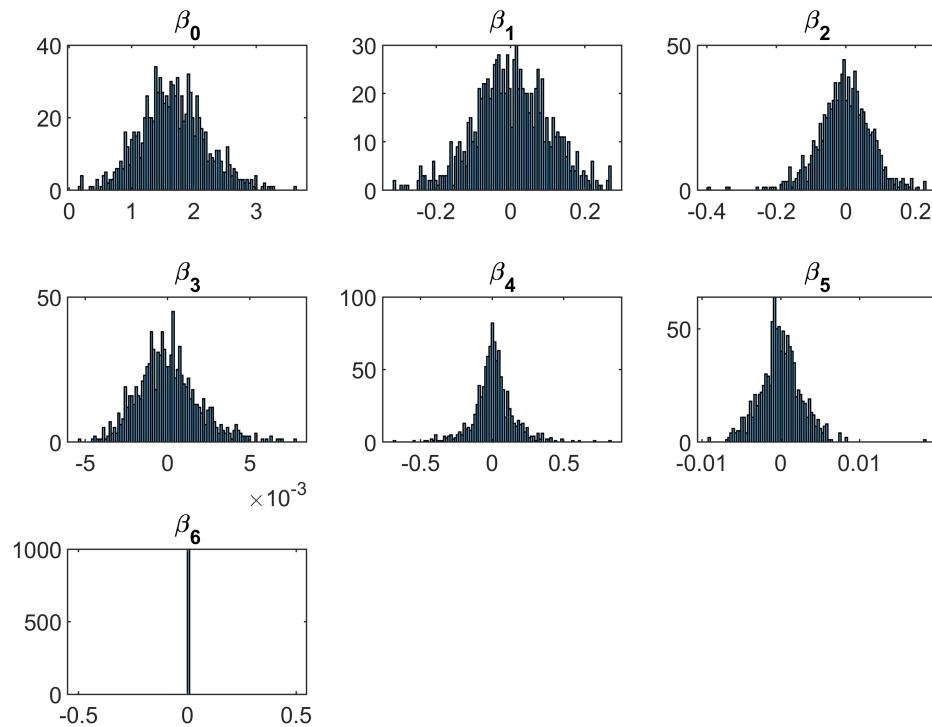
```
beta_1 = 1x7
1.6549   -0.0124    0.0081    0.0003    0.0182    0.0000      0
```

(f) Compute una matriz que contenga los vectores de estimadores  $\beta_{ols}$  para 1000 muestras

```
for i=1:1000
    X = [(ones(100,1)) xx(100,1)];
    y = log(1-rand(100,1))/(-0.6);
    beta_2(i,:) = (X\y)';
end
```

Podemos representar los coeficientes de regresión de la siguiente manera

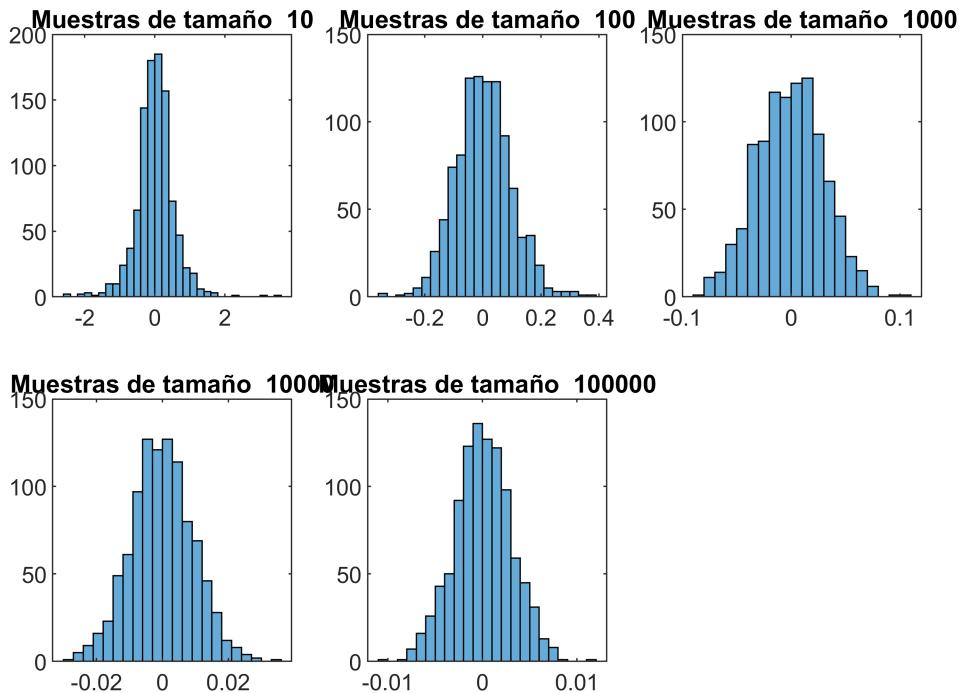
```
figure(5)
for j=1:7
    subplot(3,3,j);
    histogram(beta_2(:,j),100);
    title(['\beta_',num2str(j-1)])
end
```



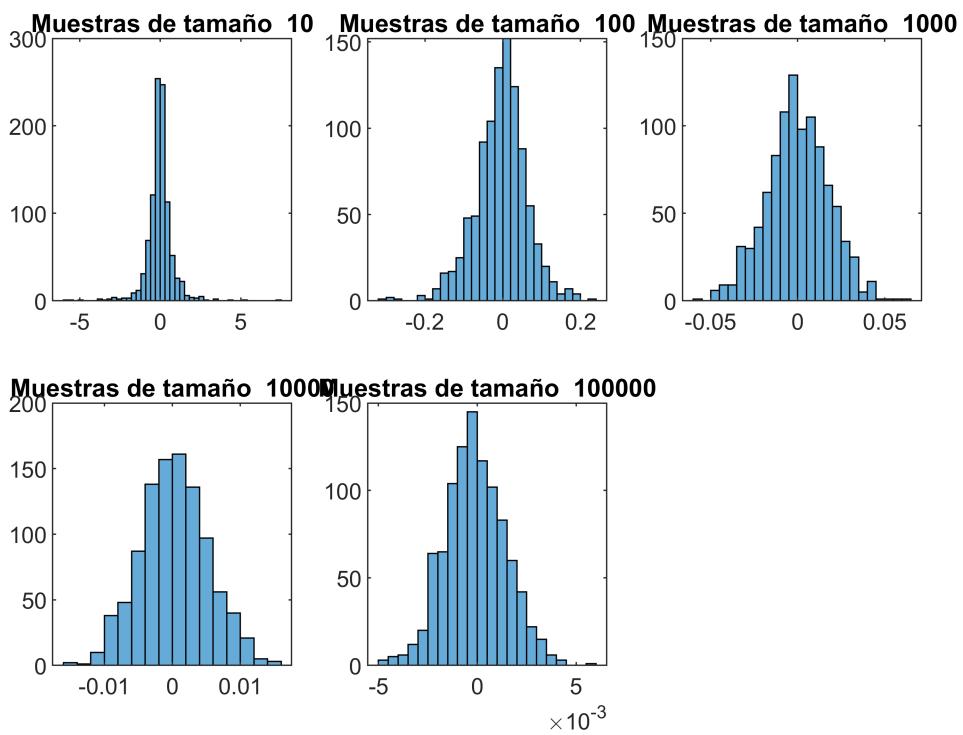
g) Desarrolle un algoritmo que reciba como *input* el **tamaño de cada muestra (m)** y el **número de muestras (i)** y que entregue como *output* un **subplot** donde se observe la distribución de cada estimador  $\beta_{ols}$  para cada figura asociada a cada tamaño de muestra. Utilice el vector  $t[10^2 \ 10^3 \ 10^4 \ 10^5]$  para los distintos tamaños de muestras.

```
m = 1000;
t = [10 10^2 10^3 10^4 10^5];
algoritmo(t,m)
```

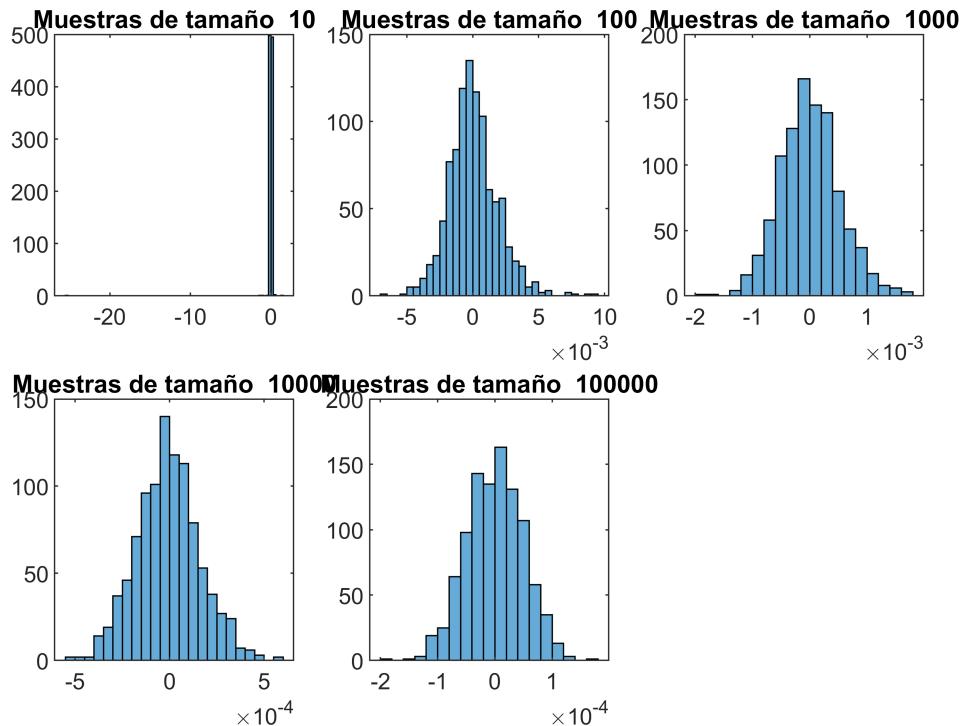
## Distribución $\beta_1$



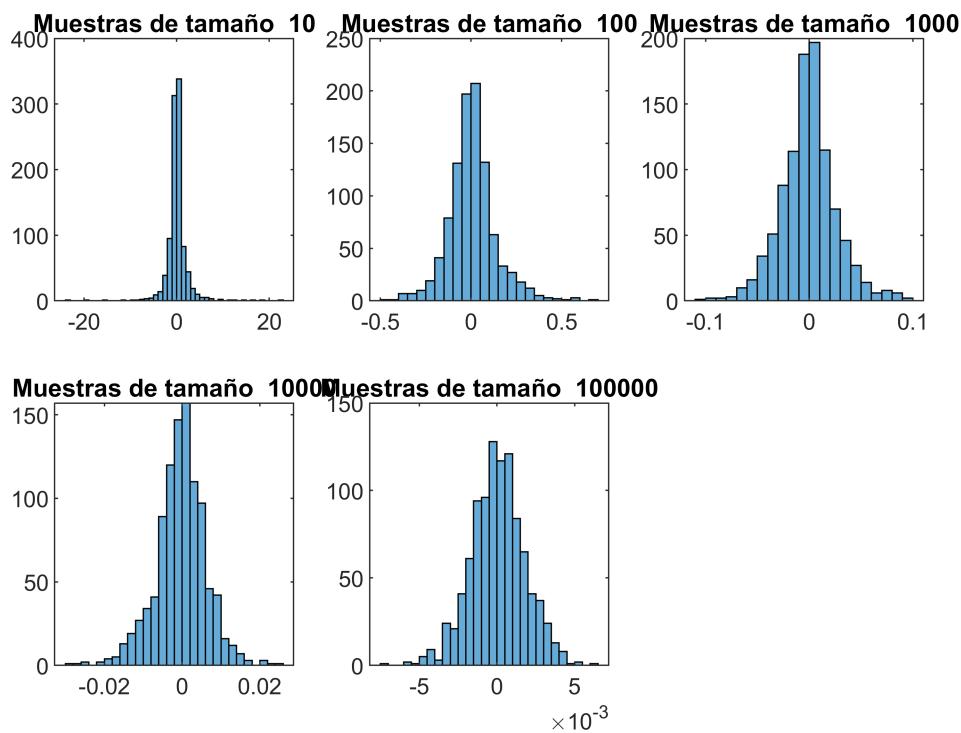
## Distribución $\beta_2$



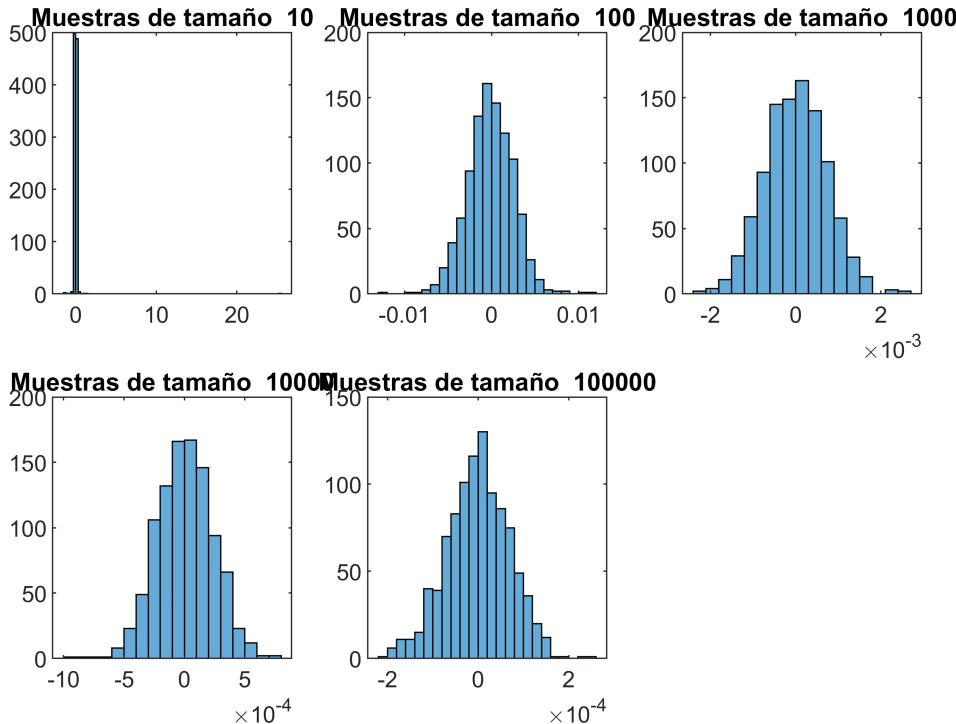
### Distribución $\beta_3$



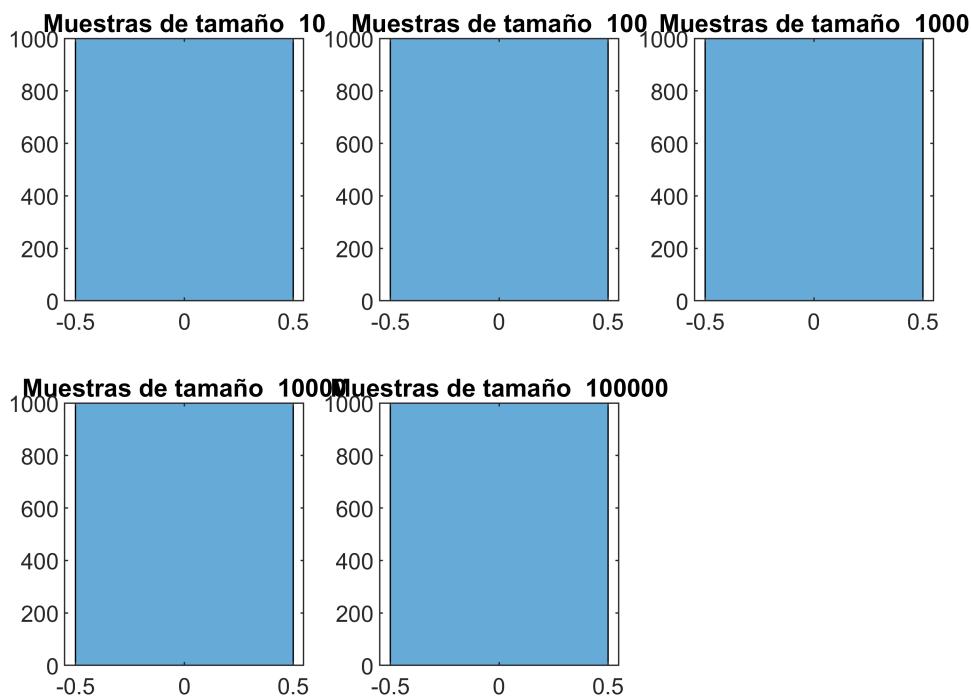
### Distribución $\beta_4$



### Distribución $\beta_5$



### Distribución $\beta_6$



```
ans = struct with fields:  
  ols_10: [1000x7 double]  
  ols_100: [1000x7 double]  
  ols_1000: [1000x7 double]
```

```
ols_10000: [1000x7 double]
ols_100000: [1000x7 double]
```

Responda: ¿Cómo distribuyen los estimadores? ¿En torno a que valor están centradas las distribuciones? ¿Por qué? Interprete y explique brevemente

En el caso de los  $\beta_0$ ,  $\beta_2$  y  $\beta_4$  podemos notar que las distribuciones están centradas en el cero, y a medida que aumenta el tamaño de la muestra su distribución es cada vez más simétrica y con menor curtosis. En estos dos casos es notable como contrasta esto con muestras de tamaño 10, donde son mucho más asimétricas y hay presencia de outliers en valores negativos en caso del intercepto, y en valores positivos en caso de el coeficiente 2.

En el caso de  $\beta_1$  esta distribución no cambia mucho con el aumento del tamaño de la muestra. Mientras que en  $\beta_3$  y  $\beta_5$  las muestras de tamaño más pequeño tienen una curtosis notable que se va suavizando a medida que aumenta el tamaño de la muestra.

El caso de  $\beta_6$  tenemos un valor también centrado en el cero, pero casi único. Recordemos que esta variable se calculada con el promedio entre  $x_4$  y  $x_5$  y estas dos tienen una distribución "espejo" donde su único valor que intersecta es el 0 (ver figuras iniciales).

En términos generales, gracias a estas gráficas podemos notificar el poder que tienen las simulaciones de Montecarlo. Montecarlo es una metodología de estudio en la cual, con la generación de una gran cantidad de números aleatorios, podemos determinar el comportamiento de un sistema y/o el valor de distintas variables.

La idea básica detrás de usar Montecarlo es ejecutar simulaciones una y otra vez para obtener la distribución probabilística de una entidad probabilística desconocida. Los métodos numéricos, como Montecarlo, muchas veces son útiles cuando los métodos analíticos son muy difíciles de resolver o la solución no existe. La razón: nos permite rescatar las **propiedades asintóticas** de la estimación de las regresiones. Y eso mismo estamos haciendo en estos ejercicios.

## Estimación de coeficientes de regresión

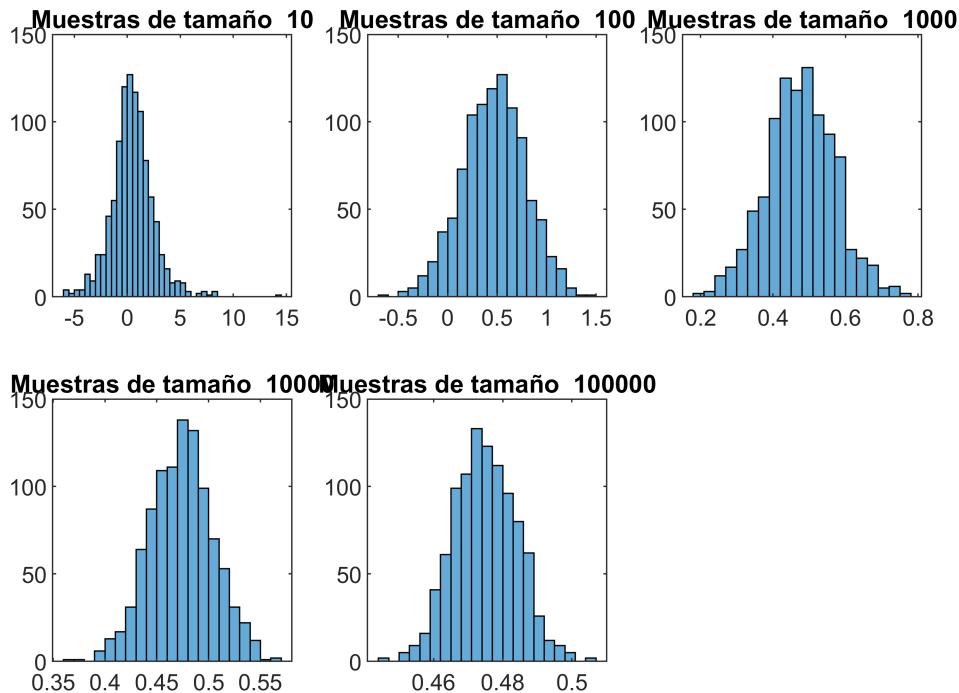
A partir de la última pregunta estimaremos el vector  $y$  y haciendo uso del vector  $\hat{\beta}_{ols} [1 2 3 4 5 6 7]'$  y del vector de residuos  $\epsilon \sim N(0, 1)$

$$y = X\beta + \epsilon$$

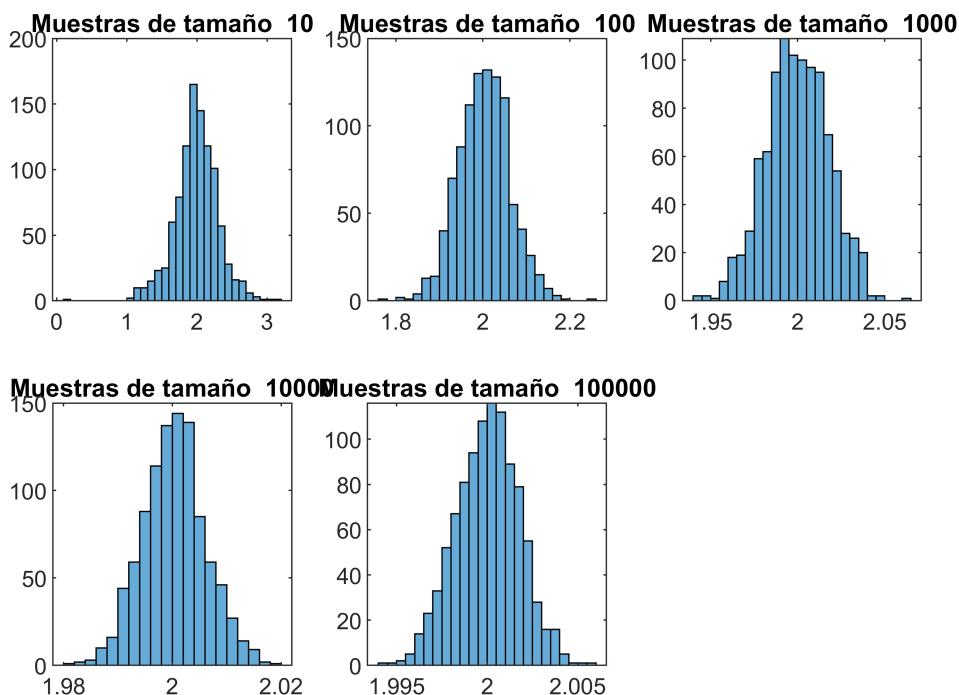
(h) De forma similar a lo desarrollado en el ítem anterior, cree un algoritmo que entregue como output un subplot donde se observe la distribución de cada  $\hat{\beta}_{ols}$  para cada tamaño de muestra distinta

```
algoritmo2(t,m)
```

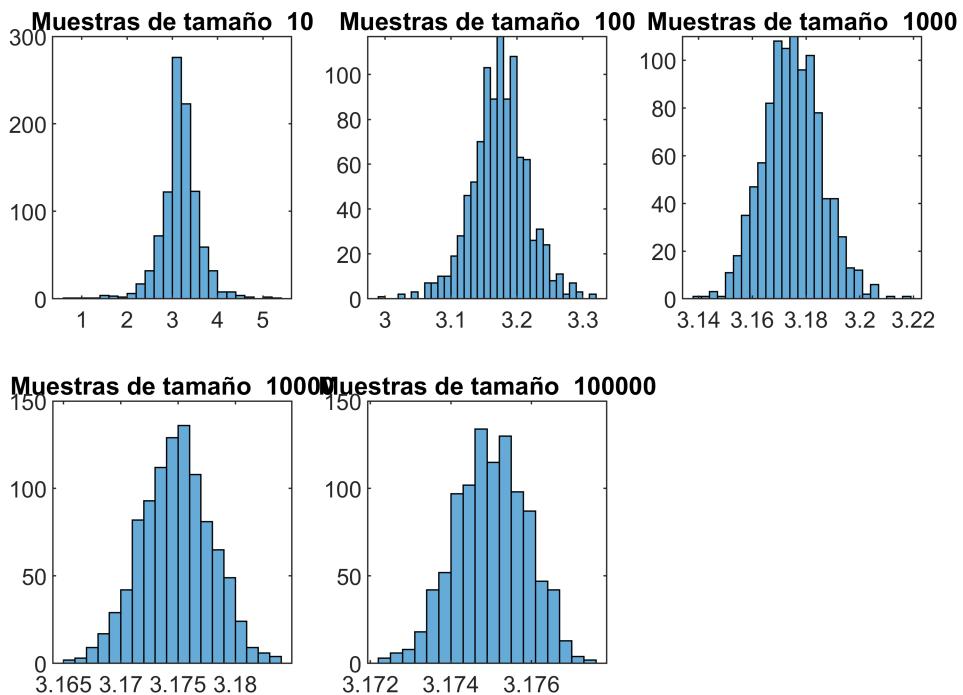
## Distribución $\beta_0$



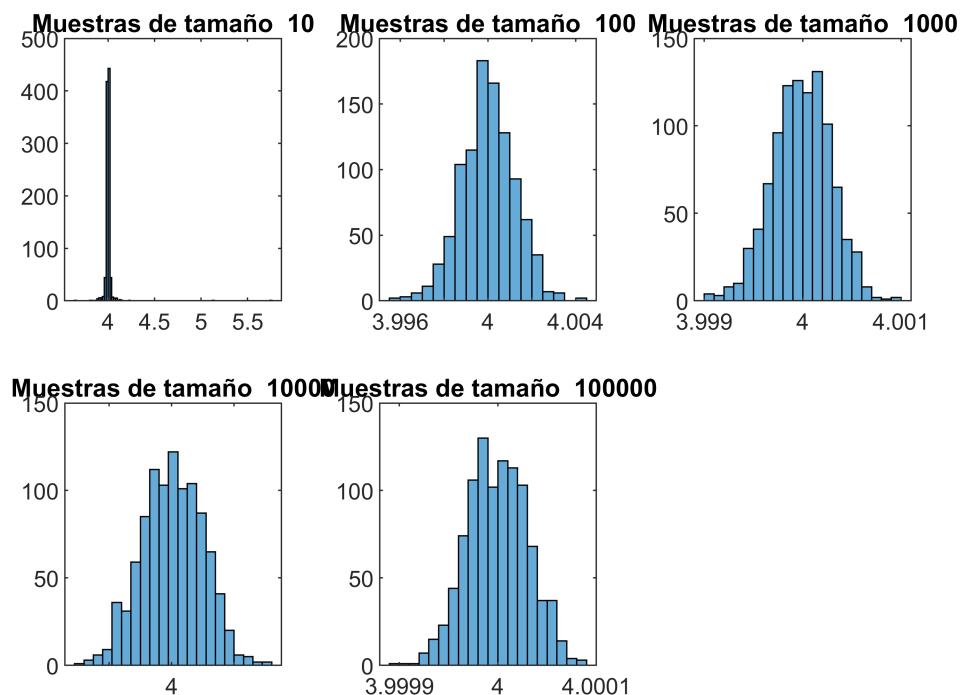
## Distribución $\beta_1$



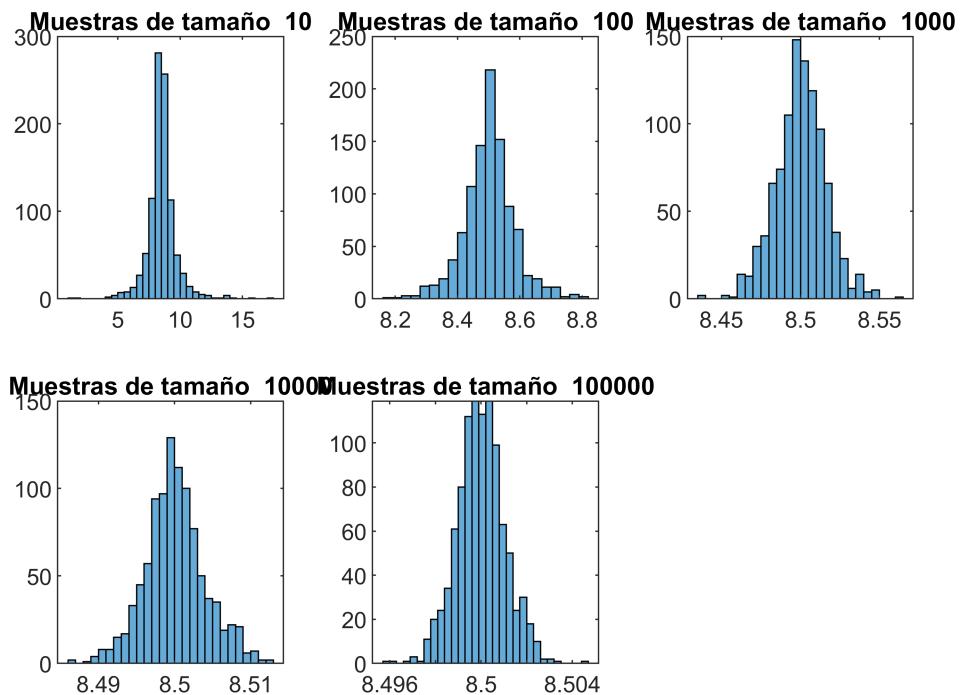
## Distribución $\beta_2$



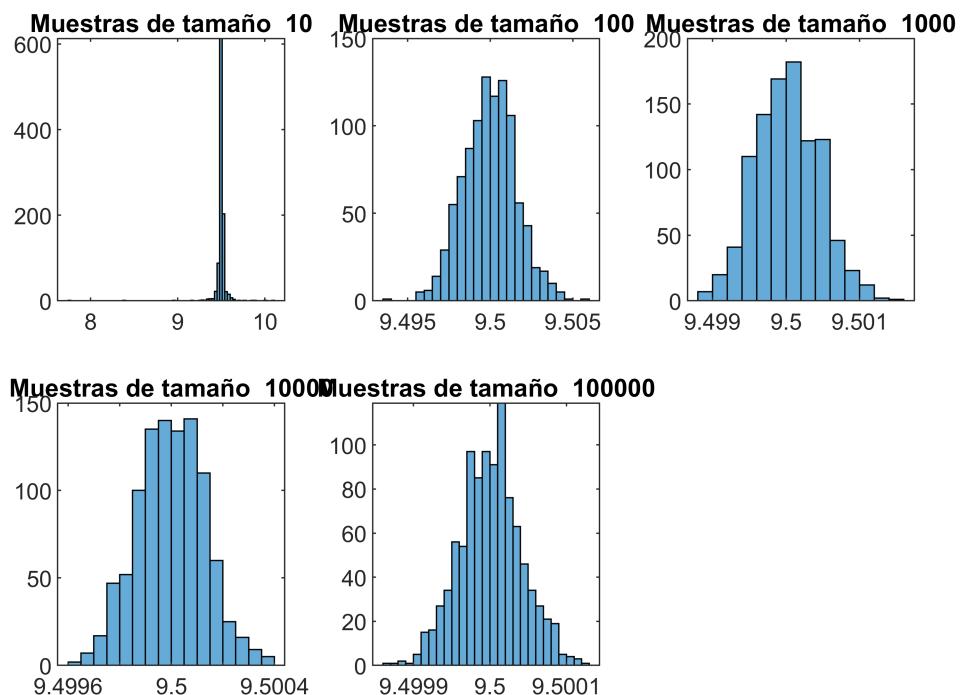
## Distribución $\beta_3$



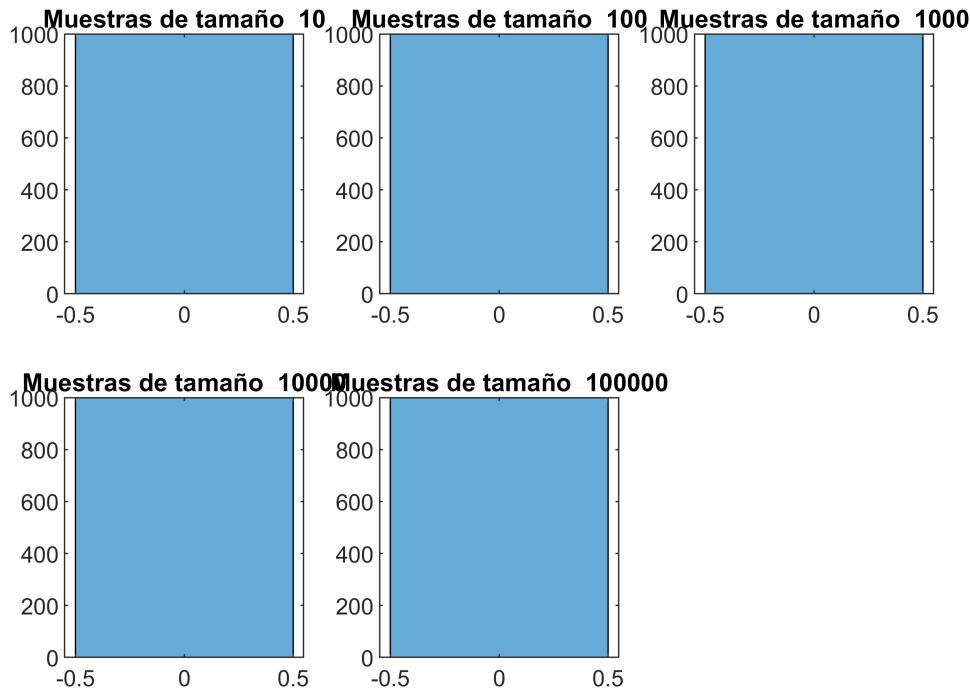
### Distribución $\beta_4$



### Distribución $\beta_5$



## Distribución $\beta_6$



```
ans = struct with fields:
    ols_10: [1000x7 double]
    ols_100: [1000x7 double]
    ols_1000: [1000x7 double]
    ols_10000: [1000x7 double]
    ols_100000: [1000x7 double]
```

¿En torno a que valores están centradas las distribuciones? ¿Porqué? Explique las diferencias respecto al inciso anterior

A diferencia del inciso anterior, las distribuciones no están centradas en el cero, sino que sus estimaciones están desviadas positivamente. La razón es que ahora, para calcular los coeficientes de regresión no utilizamos directamente los valores de la matriz de vectores de variables aleatorias, y su relación con el **y**.

Si no que partimos prediciendo el **y** con una perturbación normal ( $\epsilon \sim N(0, 1)$ ) y en donde, entonces tendremos que nuestra variable real estará estimada con un error. Formalmente tenemos que :

$$\epsilon = y - y^* \text{ donde } y^* \text{ es la variable real}$$

Luego tenemos que si bien el modelo original es

$$y^* = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \beta_3 \cdot x_3 + \beta_4 \cdot x_4 + \beta_5 \cdot x_5 + \beta_6 \cdot x_6 + u \quad (1)$$

Nosotros estimamos un

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \beta_3 \cdot x_3 + \beta_4 \cdot x_4 + \beta_5 \cdot x_5 + \beta_5 \cdot x_5 + \beta_6 \cdot x_6 + u + \epsilon \quad (1)$$

Esto implica que tendremos ahora un error compuesto: uno conformado por el "error" que teóricamente tiene el modelo estimado y otro que le agregamos a la estimación.

Notemos un punto importante y es que los estimadores no dejan de ser consistentes dado que el valor esperado del error sigue siendo cero. El problema más bien estará en la varianza condicional del error en relación al X, y que esto se podrá notar claramente en las gráficas. Matemáticamente

Sea  $v = \epsilon + u$

$$\text{Var}(v|X) = \text{Var}(u + \epsilon|X) = \text{Var}(u|X) + \text{Var}(\epsilon|X) + 2\text{Cov}(u, \epsilon|X)$$

Si decimos que existe una relación entre los errores y el error de medición tendremos que  $\text{Cov}(u, \epsilon|X) \neq 0$

Si es así, los  $\hat{\beta}$  serán sesgados. Si  $\text{Cov}(u, \epsilon|X) > 0$  el sesgo de  $\hat{\beta}$  será positivo. Eso implica que los estimadores están desviados respecto del valor real. Además de ello, los estimadores serán más ineficientes. Este resultado es esperable pues construimos sumamos una perturbación que sigue la misma distribución que el error "teórico" del modelo.

Ahora bien notemos dos cosas relevantes:

1. La primera es que la forma de y no va a cambiar. Sigue teniendo una distribución logarítmica, pero desviada en su valor real.
2. Al igual que en el caso del predictor, los estimadores **no cambiarán** su forma de distribución que se asimila a una normal. Las razones son dos. La primera tiene que ver con una propiedad lineal de la normal. Sumar normales da como resultado una normal. La segunda tiene un fuerte respaldo en el ejercicio que hemos hecho en este apartado y es que, por los procesos iterativos con grandes tamaños muestrales y de muestras repetidas nos permiten **asintóticas** decir que pese a las perturbaciones, en los límites se seguirían cumpliendo las propiedades de los estimadores (propiedades asintóticas). No así las de insesgamiento.

### Adicional: Comprobando resultados

Adicionalmente, hemos reafirmado este punto mostrando el resultado de la estimación de los coeficientes de regresión. Notemos que su valor esperado está cercano a cero en la mayoría de estos, a excepción del intercepto pero que no tiene una gran confiabilidad si miramos su error estándar. Es decir, con un 95% de confianza podemos decir que los coeficientes no tienen un efecto sustantivo sobre y dado que su valor es cero. Y como pudimos ver en el punto h, si agregamos una perturbación, podríamos llegar a resultados equivocados y sesgados respecto de la estimación.

```
regresion = fitlm(X,y)
```

```
regresion =
Linear regression model:
y ~ 1 + x1 + x2 + x3 + x4 + x5 + x6 + x7

Estimated Coefficients:
              Estimate        SE      tStat     pValue
(Intercept)  2.1158    0.6147   3.4421  0.00087036
x1            0         0       NaN      NaN
x2          -0.029526  0.11459  -0.25768  0.79723
```

x3	-0.043848	0.086832	-0.50497	0.61478
x4	-0.0029352	0.002554	-1.1492	0.25343
x5	0.23899	0.20601	1.1601	0.24901
x6	0.0044493	0.0035467	1.2545	0.21284
x7	0	0	NaN	NaN

Number of observations: 100, Error degrees of freedom: 94

Root Mean Squared Error: 1.8

R-squared: 0.0362, Adjusted R-Squared: -0.0151

F-statistic vs. constant model: 0.706, p-value = 0.62

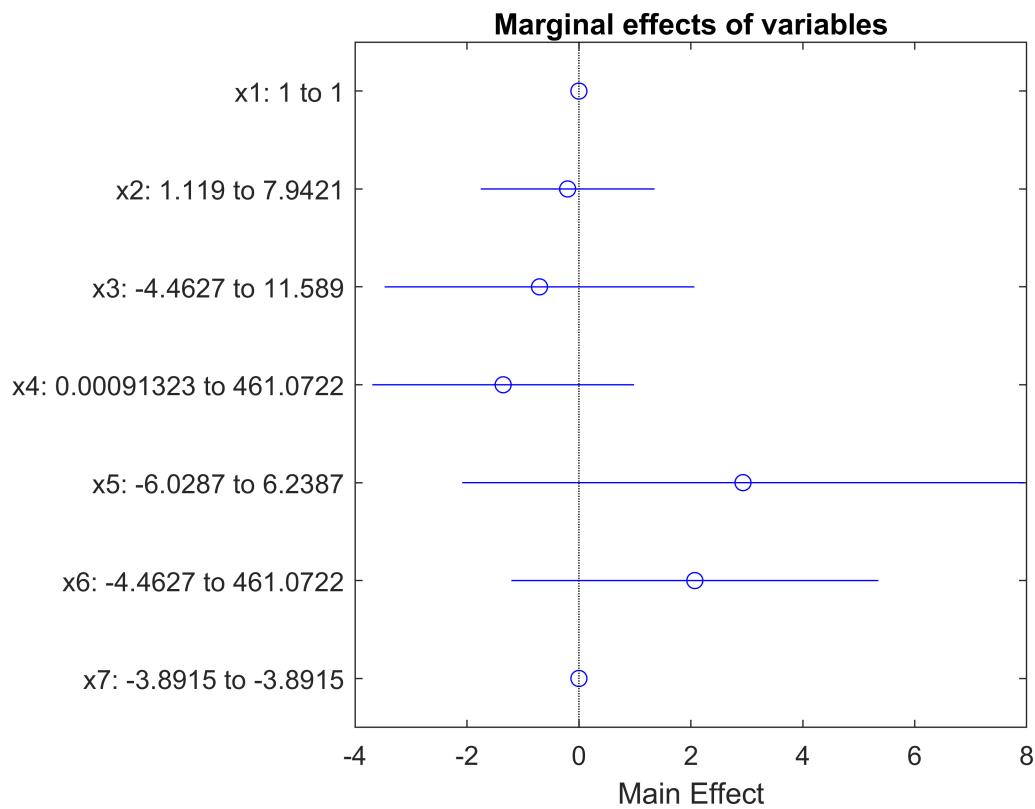
### regresion.Coefficients

ans = 8x4 table

	Estimate	SE	tStat	pValue
1 (Intercept)	2.1158	0.6147	3.4421	0.0009
2 x1	0	0	NaN	NaN
3 x2	-0.0295	0.1146	-0.2577	0.7972
4 x3	-0.0438	0.0868	-0.5050	0.6148
5 x4	-0.0029	0.0026	-1.1492	0.2534
6 x5	0.2390	0.2060	1.1601	0.2490
7 x6	0.0044	0.0035	1.2545	0.2128
8 x7	0	0	NaN	NaN

Presentamos una figura final para mostrar esta ausencia de efecto.

```
figure(30)
plotEffects(regresion)
title("Marginal effects of variables")
hold off
```



## Referencias

Greene (2016) Econometric Analysis. Pearson

Russell, K. G. (febrero de 1991). «[Estimating the Value of e by Simulation](#)». *The American Statistician* **45** (1): 66-68.

[Learning Matlab using OLS](#)

# Detrending: Hodrick–Prescott filtering [HP]

Nombre: Valentina Andrade

Profesor: Alexandre Janiak

Ayudantes: Pablo Vega y Bianca Hincapié

```
clear;
close all;
% Global settings
warning('off'); % warnings about rank
figure('Renderer', 'painters', 'Position', [10 10 900 600]); % figures size
```

En series de tiempo, para que el proceso sea covariance-stationary se debe cumplir que:

$$\mu_t = \mu; \quad \forall t \quad (2)$$

$$E[(x_t - \mu_t)(x_{t-\tau} - \mu_{t-\tau})] = E[(x_{t+s} - \mu_{t+s})(x_{t-\tau+s} - \mu_{t-\tau+s})]; \quad \forall t, s, \tau \quad (3)$$

En términos simples, las ecuaciones indican que una serie de tiempo sea estacionaria, la media (2) y la covarianza (3) no deben cambiar con el tiempo. Típicamente, este no es el caso, siendo común trabajar con series que tienen tendencias y ciclos, dando cuenta de procesos no estacionarios.

Para dar solución a esta problemática es necesario capturar la tendencia de la serie y separarla de ciclo. La idea general del proceso está dada por

$$x_t = \tau_t + c_t + s_t \quad (4)$$

Donde cada una corresponde a

$x_t$  = serie de tiempo

$\tau_t$  = tendencia

$c_t$  = ciclo

$s_t$  = estacionalidad

Para efectos de estos ejercicios  $s_t = 0$

El objetivo de este ejercicio es utilizar el filtro Hodrick y Prescott para separar la tendencia y el ciclo de la serie del precio del cobre. Para ello utilice las siguientes ecuaciones

$$y^c = (I - A^{-1})y$$

$$y^s = A^{-1}y$$

$$A = I + \lambda K'K$$

Donde  $I$  es la matriz identidad,  $\lambda$  el parámetro de sensibilidad de la tendencia a las fluctuaciones a corto plazo y  $K$  corresponde a la matriz tridiagonal

Responda las siguientes preguntas

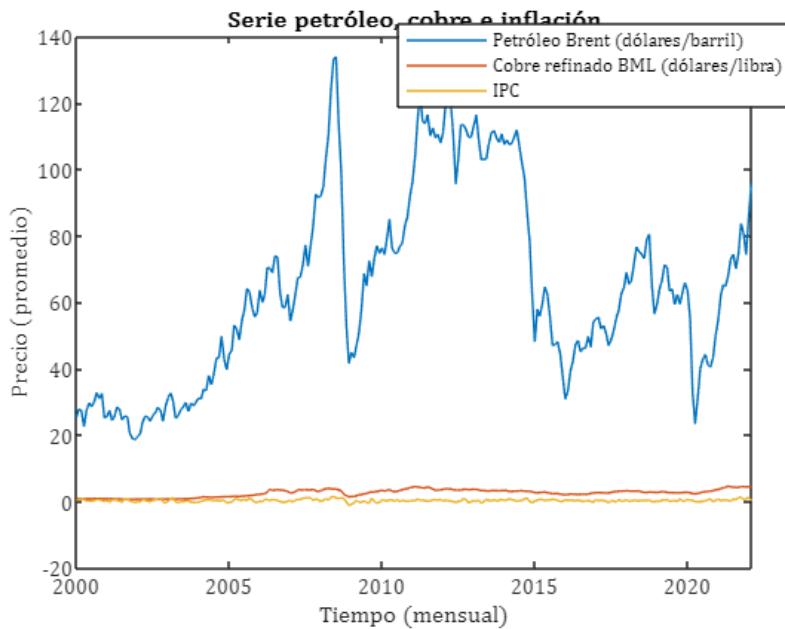
(a) Importa las series de precio del cobre ( $cu_t$ ), inflación  $\pi_t$  y el precio del barril de petróleo Brent ( $p_t$ ) haciendo uso de comandos de Matlab

```
importdata; % Funcion para importar datos
```

```
y = data{:,vartype("numeric")}; % Extraer de tabla elementos numerics
```

Primero, iniciaré mostrando la serie de datos importados de manera descriptiva

```
figuredescriptive(data.(1),y) % Grafico descriptivo
```



(b) Cree una que cumpla con las siguientes características:

- Input: (1) serie que desea descomponer y (2) parámetro de sensibilidad  $\lambda$
- Output: (1) tendencia y (2) ciclo de la serie

La función que cree se llama **hp\_filter**. Dentro de ella aparecen los comentarios del paso a paso de como la construí. Lo interesante es que esta función descompone no solo para 1 serie sino que para varias a la vez (una matriz de series) y da como resultado una matriz de series de tendencias y ciclos.

```
lambda = 1600; % Para probar la función  
[ytrend, ycycle] = hp_filter(y,lambda) % Funcion HP filter
```

```
ytrend = 266x3  
28.1146 0.8397 0.4197  
28.0566 0.8350 0.4078  
27.9969 0.8303 0.3958  
27.9337 0.8255 0.3836  
27.8646 0.8207 0.3715  
27.7840 0.8158 0.3596  
27.6860 0.8106 0.3481  
27.5658 0.8051 0.3371  
27.4191 0.7992 0.3264  
27.2434 0.7928 0.3159  
:  
:
```

```
ycycle = 266x3  
-2.7146 -0.0033 -0.2197  
-0.2866 -0.0182 0.1922  
-0.6369 -0.0413 0.3042
```

```

-5.3937 -0.0640 0.1164
-0.4646 -0.0108 -0.1715
1.8960 -0.0205 -0.1596
0.8240 0.0056 -0.2481
2.4742 0.0367 -0.0371
5.3609 0.0900 0.2736
3.6866 0.0683 0.2841
:

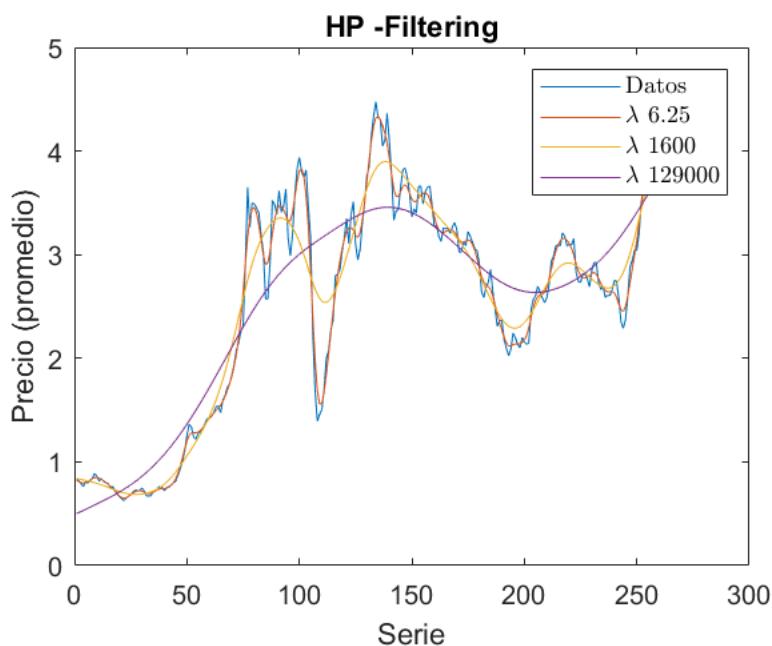
```

(c) Grafique la serie original del precio del cobre junto a las tendencias obtenidas para un vector de sensibilidad  $\lambda$  [6.25 1600 129.000]. Interprete brevemente.

```

lambda = [6.25 1600 129000];
% Dado que no se recomienda renombrar dinámicamente en matlab se crea cada
% objeto a mano
[ytrend625, ycycle625] = hp_filter(y,lambda(1));
[ytrend16, ycycle16] = hp_filter(y,lambda(2));
[ytrend129, ycycle129] = hp_filter(y,lambda(3));
graph = [y(:,2) ytrend625(:,2) ytrend16(:,2) ytrend129(:,2)]; % Matriz con variables de cobre
hpfigure(graph) % Grafico

```



### Interpretación econométrica

Cuando Hodrick, y Prescott (1997) propusieron un procedimiento para representar las series de tiempo, pensaron en el problema del componente cíclico y tendencial de los fenómenos temporales, sobre todo por las diferencias en análisis que producían no reconocer estos componentes.

En ese sentido, este método para filtrar estos componentes es bastante útil para análisis macroeconómicos de series de tiempo, como el precio del cobre y la inflación. Como muestra Uhlig (1995, p. 46) el método remueve la tendencia de un set de datos cuando

$$\min \sum_{t \in T} ((s_t - \tau_t)^2 + \lambda((\tau_{t+1} - \tau_t) - (\tau_t - \tau_{t-1}))^2)$$

Así, a medida que crece  $\lambda$  las diferencia entre la tendencia y el ciclo se van a ir homogenizando respecto al valor tendencial actual y pasado.

Justamente esto es lo que se puede evidenciar en la figura construida con **figurehp.m**. Entre menor es el parámetro de sensibilidad, más similar es a la serie original. Mientras que en el caso de la línea púrpura se puede evidenciar un suavizamiento del ciclo mucho más pronunciado.

### Interpretación económica

En general, existe una relación estrecha entre el precio del cobre y el ciclo económico en Chile (De Gregorio y Labbe, 2011). Por ejemplo, investigaciones han mostrado que si la política fiscal se conduce bajo la regla de superávit estructural, entonces un aumento del 10% en el precio del cobre incrementaría el PIB en solo 0,05% y generaría una pequeña caída en la inflación (Medina y Soto, 2007). Principalmente se ha argumentado que esto se debe a las apreciaciones cambiares que acompañan el shock y el ahorro transitorio de los ingresos desde los hogares. Ahora bien, podemos notar que esta sensibilidad ha ido disminuyendo en el tiempo.

Otros modelos modelo dinámica de equilibrio general estocástico (DSGE) han mostrado que la volatilidad del producto respecto a la contribución del precio del cobre la disminuido a corto plazo. Tal como pronosticaban Medina y Soto (2007) existe un efecto de amortiguación de los shocks por parte del tipo de cambio real: si bien la volatilidad ha aumentado a corto plazo, a largo plazo se ha mantenido estable.

Para cerrar, este último punto es importante para entender la metodología y la gráfica propuesta. A corto plazo, el precio del cobre está fuertemente relacionado al ciclo macroeconómico, por lo que su componente cíclico puede ser influyente de su distribución no desestacionalizada. Mientras que a largo plazo el efecto del precio del cobre tendería a ser menor, sobre todo considerando en que en los últimos años la regla fiscal en Chile ha estado fija (sobre todo después de la crisis subprime) y el régimen de metas de inflación han jugado un rol clave en la estabilización.

### (d) Estime una regresión lineal mediante OLS de los siguientes modelos

$$M1 : \pi_t = \beta_0 + \beta_1 \pi_{t-1} + \beta_2 cu_t + \beta_3 cu_{t-1} + \epsilon$$

$$M2 : \pi_t = \beta_0 + \beta_1 \pi_{t-1} + \beta_2 p_t + \beta_3 p_{t-1} + \epsilon$$

Para ello he creado una función que tiene de input los datos y de output los dos modelos de regresión solicitados. Esto se ha hecho de manera algebraica.

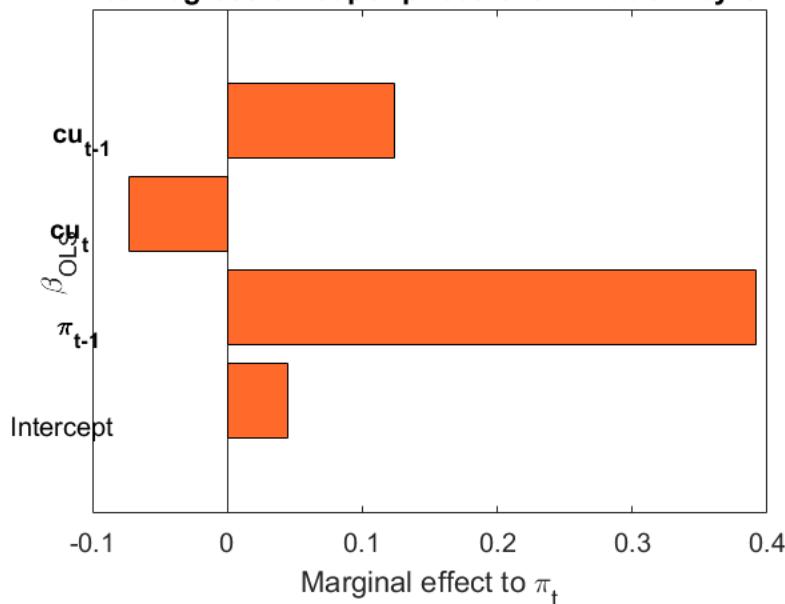
```
[m1,m2] =regresions(data) % Funcion para hacer regresiones
```

```
m1 = 1x4
0.0443    0.3917   -0.0730    0.1234
m2 = 1x4
0.0484    0.4159   0.0109   -0.0091
```

Para hacer más fácil la interpretación se han realizado dos gráficos

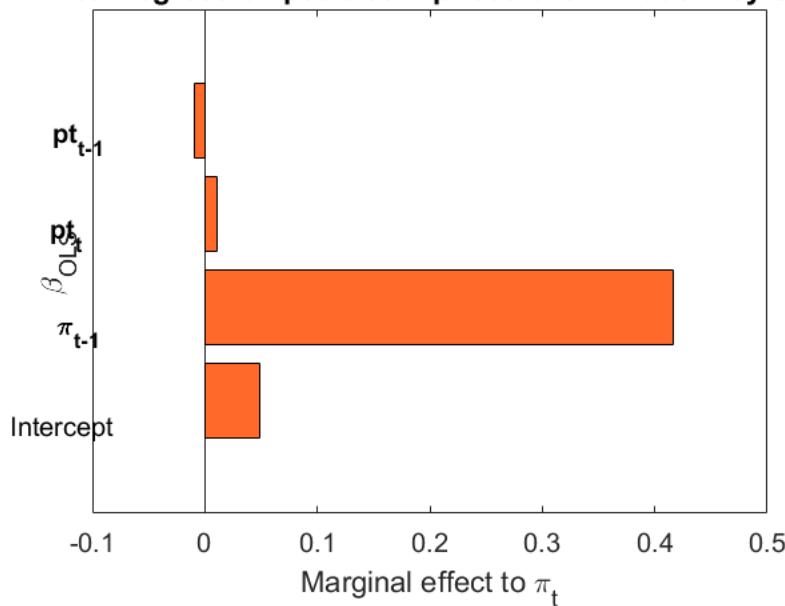
```
m1regression(m1)
```

### Linear regression copper prices over inflation by OLS



m2regression(m2)

### Linear regression petroleum prices over inflation by OLS



Como se había descrito más arriba, si bien se ha relevado la importancia del precio del cobre para la performance macroeconómica, recientes investigaciones han mostrado que su importancia sobre la contribución al PIB y la inflación ha ido disminuyendo. Según De Gregorio y Labbe (2011) las razones son dos:

1. La regla fiscal ha sido muy importante en la gestión macroeconómica. Esta ha otorgado una capacidad predictiva sobre la política fiscal, y con ello, la política monetaria ha podido ajustarse mucho mejor ante contextos inflacionarios.

2. La política monetaria basada en metas ha sido clave para entender las tendencias de inflación en Chile. Es decir, en la medida en que el Banco Central ha fijado las expectativas respecto a la inflación esperada, y su alta capacidad de acción ante las reglas fiscales claras, la mejor forma de predecir la inflación futura se ha transformado en ver el comportamiento del Banco Central y sus anuncios de tasas de política monetaria.

De este modo, a diferencia de investigaciones anteriores como las de Calvo y Mendoza (1999), el *path de la inflación* está fuertemente influenciado por la inflación misma más que en el precio del cobre o el precio del cobre dividido en el petróleo. De hecho, tal como argumenta De Gregorio y Labbé (2011) las ideas de Calvo (1999) y Caballero (2002) podrían haber sido relevantes incluso cuando Chile estaba limitado en el acceso de financiamiento internacional. A su vez, estos estudios tampoco consideran los datos más recientes de los 2000s donde los resultados de las fluctuaciones incluso han sido más moderadas.

En síntesis, tanto el cobre como el petróleo han perdido en la última década su poder predictivo sobre la inflación, y más bien la inflación pasada es la que por ahora nos otorga sus mejores predicciones. No obstante esta estabilidad hoy podría ser puesta en duda: la crisis económica y financiera producida por la pandemia, la presión por stock de petróleo ante el bloqueo a Rusia por la guerra en Ucrania podrían ser más que un *shock* transitorio y cambiar el rumbo de la economía.

## Referencias

Banco Central (2022) Series de precios de cobre y petróleo.

Uribe, M and Schmitt, S (2017) *Open Economy Macroeconomics*. Princeton University Press

Hodrick, R.J. and E.C.Prescott (1997), "Postwar U.S. Business Cycles: An Empirical Investigation." Jounal of Money, Credit and Banking. 29(1), Feb. pp. 1–16.

Medina, J. P., & Soto, C. (2007). *Copper price, fiscal policy and business cycle in Chile* (p. 1). Santiago: Banco Central de Chile. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.506.2292&rep=rep1&type=pdf>

De Gregorio, J., & Labbé, F. (2011). *Copper, the real exchange rate and macroeconomic fluctuations in Chile*. Santiago: Central Bank of Chile

Uhlig, H (1995) A toolkit for Analizing Nonlinear Dynamic Stochastic Models. CentER, University of Tilburg

Mathworks (2022) Tutorial why variables shold not be named dynamically.

# Raíces de una función

```
clear;
close all;
% Global settings
warning('off'); % warnings about rank
```

El siguiente ejercicio contiene los siguientes códigos para su desarrollo

- [bisection.m](#)
- [newtonraphson.m](#)

## 3.1 Entendiendo las distintas metodologías

### 3.1.1 Newton-Raphson

El algoritmo de Newton-Raphson permite encontrar la raíz de una función partiendo desde un punto del dominio cercano a dicha raíz. Es un método **iterativo** y preciso que es útil sobre polinomios de grado *impar*. Desde una aproximación de **Taylor** de primer orden es posible obtener la siguiente sucesión que define el algoritmo. Es importante notar que dos valores consecutivos de la sucesión serán cada vez más similares en la medida en que se aproximen a la raíz de la función :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Con  $x_0$  conocido. La figura adjunta en el enunciado explica de forma gráfica el método de estudio

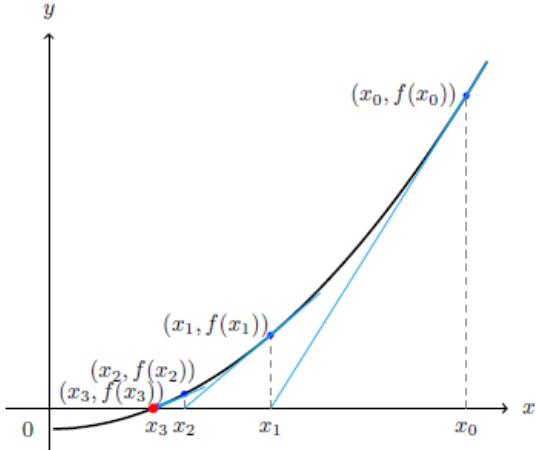


Figure 3: Descripción gráfica del método.

Formalmente sea  $f(x)$  una función  $C^1$  y continua, y  $x_r$  una raíz desconocida sobre  $f(x)$ . Asumamos que  $x_0$  es un *guess para*  $x_r$ . A menos que hayamos tenido mucha suerte,  $f(x_0)$  no es una raíz. Dado ese escenario queremos encontrar un  $x_1$  tal que se mejore la performance de  $x_0$  (esto es, estar más cerca de  $x_r$  que de  $x_0$ ). Ahora bien, si suponemos que de todas maneras  $x_0$  está lo suficientemente cerca de  $x_r$ , entonces podremos hacer una **aproximación lineal de**  $f(x)$  alrededor de  $x_0$  tal que buscaremos encontrar la intersección de esa

línea con el eje x. En síntesis, estaremos haciendo una aproximación lineal de  $f(x)$  alrededor de  $x_o$  que se escribe de la siguiente manera:

$$f(x) \approx f(x_o) + f'(x_o)(x - x_o)$$

Luego, como indicamos, encontraremos  $x_1$  tal que  $f(x_1) = 0$ . Tomando esos valores y llevándolos a la aproximación lineal resulta la siguiente ecuación:

$$0 = f(x_o) + f'(x_o)(x - x_o)$$

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_o)}$$

En términos generales, el método **de Newton** computa la aplicación del guess,  $x_n$ , usando un guess anterior  $x_{n-1}$  y que está dado por la siguiente ecuación

$$x_n = x_{n-1} = \frac{g(x_{n-1})}{g'(x_{n-1})}$$

Ahora bien el método **Newton-Raphson** aplica encontrar esa raíz pero iterando el método de Newton desde  $x_0$  hasta que el error sea menor a una tolerancia esperada.

### 3.1.2 Algoritmo de Bisección

Basado en el teorema de los Valores Intermedios, el algoritmo de Bisección establece:

Sea

$f : [a, b] \rightarrow R$  una función continua diferenciable  $C^1$  donde todo valor entre  $f(a)$  y  $f(b)$  es imagen de al menos un valor del intervalo  $[a, b]$ . Luego, si  $f(a)$  y  $f(b)$  tienen signos opuestos ( $f(a) \cdot f(b) < 0$ ), el teorema asegura la existencia de al menos un valor  $x$  del dominio de  $f$  para el cual  $f(x) = 0$ .

Para programar el algoritmo de Bisección siga los siguientes pasos

1. Compute una tolerancia  $\tau = 10^{-4}$  y defina un intervalo  $[a, b]$  que contenga la raíz de la función.
2. Dada una función que cumpla con las características, compute  $f(a)$  y  $f(b)$
3. Compute  $\bar{r} = \frac{a+b}{2}$  y luego obtenga  $f(\bar{r})$ . Si el signo de  $f(\bar{r})$  es igual al signo  $f(a)$ , entonces el nuevo intervalo definido en el dominio de la función seá  $[\bar{r}, b]$  de otra forma el nuevo intervalo será  $[a, \bar{r}]$ .
4. Compute el error de aproximación  $\epsilon = f(\bar{r})$
5. Mientras  $\epsilon > \tau$  repita el proceso hasta que el algoritmo converja a la tolerancia deseada

Generalmente el algoritmo de bisección no produce una solución exacta a la ecuación  $f(x)=0$ . Ahora bien, se puede proveer de una estimación sobre el error absoluto. El teorema es el siguiente.

Teorema. Sea  $f(x)$  una función continua entre  $[a, b]$  tal que  $f(a)f(b) < 0$ . Luego de  $N$  iteraciones del método de bisección, sea  $[X_n]$  el punto intermedio en el subintervalo  $n$ -ésimo  $[a_n, b_n]$

$$X_n = \frac{a_n + b_n}{2}$$

Entonces, existe una solución exacta y verdadera de la ecuación  $f(x) = 0$  en el subintervalo  $[a_n, b_n]$  y el error absoluto es

$$\epsilon > [x * -x_n] \leq \frac{b - a}{2^{n+1}}$$

Notemos que podemos reorganizar el límite del error de modo de poder notificar cuál es el número mínimo de iteraciones requeridas para garantizar que el error absoluto sea menor al descrito

$$\begin{aligned} \frac{b - a}{2^{n+1}} &< \epsilon \\ \frac{b - a}{\epsilon} &< 2^{n+1} \\ \ln \frac{b - a}{\epsilon} &< n + 1 \cdot \ln(2) \\ \frac{\ln \frac{b - a}{\epsilon}}{\ln(2)} - 1 &< n \end{aligned}$$

(a) Usando el método Newton-Raphson, encuentra la raíz de las siguientes funciones

$$\begin{aligned} f(x) &= x^3 + x + 2 \\ f(x) &= x^5 + x^4 + x^3 + x^2 + 1 \\ f(x) &= \log(x) + \log(3x^3) \end{aligned}$$

```
clear;
f = @(x) x^3 - x + 2;
x0 = 0
```

$x0 = 0$

```
newtonraphson(f,x0)
```

$ans = -1.5214$

```
x0 = 1;
F = {@(x) x.^3-x+2; @(x) x.^5+x.^4+x.^3+x.^2+1; @(x) log(x)+log(3*x.^3)} ;
newtonraphson(F{1},x0)
```

$ans = -1.5214$

```
newtonraphson(F{2},x0)
```

$ans = -1.2499$

```
newtonraphson(F{3},x0)
```

```
ans = 0.7598
```

(b) Obtenga la raíz de las funciones usando el algoritmo de Bisección. ¿Obtiene los mismos resultados?

Como podemos ver se obtienen los mismos resultados para ambos métodos numéricos de cálculo de raíces

```
% La funcion bisection calcula para un f e intervalo a,b, la raiz y error
F = {@(x) x.^3-x+2; @(x) x.^5+x.^4+x.^3+x.^2+1; @(x) log(x)+log(3*x.^3)} ;
a = -6;
b = 6;
bisection(F{1},a,b)
```

```
Solucion: -1.5214 Error: 2.6274e-46
ans = -1.5214
```

```
bisection(F{2},a,b)
```

```
Solucion: -1.2499 Error: 8.7581e-47
ans = -1.2499
```

```
% Para log definimos el intervalo entre 0 y +infinito pues en negativos se
% indefinie y ya no se cumpliría el teorema
bisection(F{3},0,12)
```

```
Solucion: 0.75984 Error: 2.1895e-47
ans = 0.7598
```

(c) ¿Cuál algoritmo es más eficiente? Justifique brevemente

```
f = @( ) bisection(F{1},a,b); % handle to function
timeit(f)
```

```
Solucion: -1.5214 Error: 2.6274e-46
ans = 0.0016
```

```
f = @( ) newtonraphson(F{1},x0); % handle to function
timeit(f)
```

```
ans = 0.0257
```

Si medimos la eficiencia en términos de tiempo en que se demora el método en calcular la raíz, el algoritmo de bisección se demora 0.0048 segundos, mientras que el algoritmo de Newton Raphson se demora 0.0270 segundos. Ahora bien, en programación la eficiencia no solo tiene que ver con el tiempo si no que de los parámetros con los cuales se construyen las funciones, y su capacidad de replicabilidad. Por ello, presentaré una breve discusión de las ventajas y desventajas de ambos métodos, haciendo alusión a sus potenciales respecto a eficiencia y eficacia (precisión).

1. La ventaja principal del método de Newton es que cuando se converge, esto ocurre rápidamente. Ahora bien, no asegura la existencia de esa convergencia, lo que evidentemente es una desventaja comparada con el método de bisección que garantizan la convergencia hacia una solución.
2. Además, el método de Newton requiere computar las derivadas de la función objetivo, algo que potencialmente puede ser una desventaja si la derivada es difícil de calcular.
3. Penúltimo el criterio de convergencia del método de Bisección y Newton es muy distinto: en el primero nosotros sabemos qué tan cerca estamos de la solución pues estamos computando la estimación dentro de intervalos que si o si contienen la solución; en el segundo, no sabemos realmente qué tan cerca estamos de la solución.
4. Por último, computacionalmente es mucho menos preciso Newton que Bisección. Esto pues en el primero debemos indicar hasta qué número de iteraciones se quiere seguir probando la convergencia, mientras que en método de bisección el criterio de *stopping* es qué tan pequeño es el error de estimación. Las iteraciones quedan sujetas a ello.

(d) Cree las funciones NR y BS que permite encontrar las raíces de una función con características indicadas mediante el método Newton-Raphson y Bisección respectivamente. Utilice las funciones para computar raíces de los incisos anteriores nuevamente.

Las funciones se crearon enseguida, dado que me es más fácil pensar en las funciones en términos generales y luego ver si su aplicación funciona para los casos particulares propuestos.

Solo agregar que en ambos métodos de igual manera debe existir una vigilancia del usuario sobre los **dominios de la función**. No es trivial dónde viven las funciones a evaluar, por lo que son métodos dependientes de las cotas de las funciones. Por lo mismo se crearon *warnings* ante ciertas propiedades que el usuario no podría prever. Por ejemplo,

1. El número de iteraciones excede las que el programa se propuso. Para lograr converger se propusieron al menos 100 para Newton-Raphson, mientras que para Bisección eso no fue necesario.
2. En el caso de bisección la direccionalidad de las funciones era relevante, por lo que dado el teorema, era necesario asegurarse que las soluciones de la función vivieran dentro del intervalo. Lo positivo es que da igual si se elige un intervalo mayor al encontrado (por ejemplo que la solución viva entre  $[a,b]$  pero se elija  $[a+30,b+30]$ ) pues la estimación es tan consistente que eso no trae problemas.

### 3.2 Aplicación de tiro vertical

En física la notación de Tiro Vertical describe un movimiento rectilíneo uniformemente acelerado, es decir, un movimiento que se caracteriza por tener aceleración constante. Este tipo de movimiento intenta estudiar la trayectoria de objetos que son lanzados verticalmente desde una cota inferior hacia una cota superior, es decir, hacia arriba. Una buena forma de comprender este movimiento es imaginar a una maquina “lanzapelotas”

posicionada en  $90^\circ$ . La trayectoria del movimiento de la pelota puede ser explicada por las ecuaciones de un Tiro Vertical.

El propósito de este ejercicio es utilizar el algoritmo de Bisección y Newton-Raphson para hallar la solución al problema.

## Contexto

Imagine que durante un día de partido de tenis, su rival no se presenta y usted decide practicar su saque con la ayuda de una máquina “lanzapelotas”. Para optimizar sus movimientos y energía, **desea saber el momento exacto en que la pelota alcanzar a la altura máxima en su recorrido**, de tal forma de poder aplicar un golpe limpio. Usted sabe que las ecuaciones de Tiro Vertical son las que mejor describen el movimiento de la pelota y están dadas por

$$h(t) = h_0 + v_0 \cdot t - \frac{1}{2} g \cdot t^2$$

$h(t)$  : altura de la pelota al instante  $t$

$h_0$  : altura inicial  $\rightarrow 50$  (cm)

$v_0$  : velocidad incial  $\rightarrow 4.5 \left( \frac{m}{s} \right)$

$g$  : aceleración de gravedad  $\rightarrow 9.81 \left( \frac{m}{s^2} \right)$

(a) Mediante el algoritmo de Bisección y Newton-Raphson obtenga el tiempo  $t^*$  en el cual la pelota alcanza la altura máxima.

El lanzamiento de la pelota es una parábola cuya trayectoria se describe por  $h(t)$ . En ese sentido nosotros queremos saber cuál es la cúspide de esa trayectoria, o matemáticamente, queremos saber el valor de

$t$  cuando  $\frac{\partial h(t)}{\partial t} = 0$ . Los pasos son

1. Encontrar y clasificar los extremos globales de la función (que realizaremos con método Newton-Raphson y Bisección)
2. Encontrar los puntos críticos computando la primera derivada y luego despejando  $t$

```
f = @(t)50+4.5*t -0.5*9.81*t^2; % Expresamos el problema numericamente y con x=t
```

Si calculamos con el método de Newton Raphson obtendremos la menor raíz de la solución ( $t = -2.7668$ )

```
t0= 0 % guess
```

```
t0 = 0
```

```
newtonraphson(f,t0)
```

```
ans = -2.7668
```

Cuando calculamos con el método de bisección obtenemos la mayor raíz de la solución ( $t=3.6843$ )

```
bisection(f,0,8)
```

```
Solucion: 3.6843 Error: 8.7581e-47
ans = 3.6843
```

Hasta aquí uno se podría asustar y pensar que hay un error. Pero, recordemos lo que indicamos al inicio de este ejercicio: ambos métodos son eficientes pero sensibles a los intervalos por el cuál nosotros los estudiamos, por lo que podremos muchas veces obtener raíces para máximo o mínimos, y no necesariamente serán los globales. Veamos si tomamos el método de bisección por el otro lado obtenemos la raíz -2.7668

```
bisection(f,-8,0)
```

```
Solucion: -2.7668 Error: 1.7516e-46
ans = -2.7668
```

Lo mismo con Newton Raphson

```
newtonraphson(f,1)
```

```
ans = 3.6843
```

Lo que evidentemente evidencia es que **entre estas raíces existe el máximo de la función, y con ello podemos clasificar los extremos globales de la función**

El segundo paso que viene consiste en derivar la función

```
syms t
f1 = str2sym(char(f));
df = diff(f1);
df = matlabFunction(df);
% Obtener punto critico. Sabemos que newton raphson funcional igual, asi
% que para simplificar ocupamos biseccion
[t1] = bisection(df,0,8)
```

```
Solucion: 0.45872 Error: 2.1895e-47
t1 = 0.4587
```

Entonces con la primera bisección encontramos los intervalos donde vive el máximo o mínimo global, y ahora con la segunda de estas obtuvimos el máximo.

(b) Obtenga la altura máxima que alcanza la pelota.

Tenemos que evaluar la función original y obtentemos el maximo

```
maxf = 0.5+4.5*t1 -0.5*9.81*t1^2
```

Nuevamente, como nuestro punto crítico vive en el intervalo de los zeros que calculamos inicialmente podemos decir que en 1.5321 es un punto de inflexión entre mínimos globales y máximos globales.

(c) ¿Cuál es la velocidad de la pelota en el instante  $t$ ? ¿Por qué son válidos estos algoritmos para encontrar la solución al problema? Explique brevemente

La velocidad se obtiene así

$$v(t) = \frac{\partial h(t)}{\partial t} = -9.8t + 4.5$$

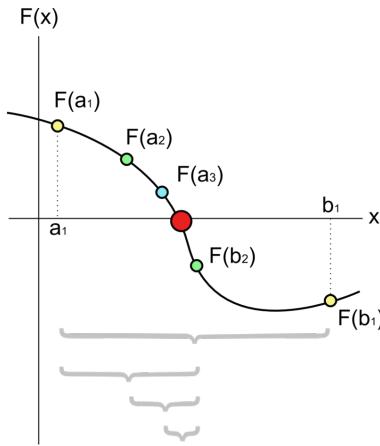
-9.8\*t1 + 4.5

ans = 0.0046

Notemos que la velocidad en ese punto es aproximadamente cero, lo que rectifica que nuestros cálculos son correctos dado que en un movimiento parabólico regido por MRU y MRUA, cuando se alcanza el máximo de distancia, el vector velocidad es igual a cero debido a que el cuerpo pierde toda su energía cinética que lo hizo desplazarse hacia la posición máxima. Luego, el cuerpo va a seguir su trayectoria pero a una velocidad dependiente de la aceleración. Formalmente cuando la velocidad en el eje y vale 0, obtenemos el tiempo t que tarda el cuerpo en llegar a dicha altura.

Ahora respecto a la validez de estos métodos para estos problemas, se pueden hacer dos acotaciones, una formal y otra sustantiva.

1. El movimiento descrito consiste en una función  $f: R^2 \rightarrow R$ , acotada, continua y diferenciable en todo su intervalo. En ese sentido tanto el método de Newton como de Bisección que se inspira en el **teorema del valor intermedio**, pueden ser matemáticamente aplicados.
2. En términos sustantivos, nosotros tenemos una trayectoria que se describe por una función de la cual desconocemos sus puntos o raíces exactamente. Una forma de aproximarnos es partir conociendo sus valores extremos (Teorema de Weierstrass). Una vez conocido ese máximo y ese mínimo, por el teorema de la acotación (Bolzano) podemos establecer que en toda función continua cerrada, esta es acotada. El ejercicio será ir construyendo cada vez menores intervalos que nos lleven a un único valor c que está contenido en la cota. Gráficamente



En ese sentido, en esta búsqueda por ese  $f(c) = 0$ , estaremos **mapeando** todo el recorrido de la función. Con esa aproximación, luego, podemos conocer cualquier momento de nuestra distribución, tal como los puntos máximos, la velocidad (varianza) u otros.

(d) Imagine que ahora desea estudiar el movimiento denominado caída libre, ¿Podría utilizar las funciones NR y BS para estudiar la trayectoria del objeto de estudio? No es necesario que haga cálculos. Explique brevemente la lógica que hay detrás de los movimientos estudiados.

Anteriormente se entregó una intuición sobre el estudio de funciones de las cuales tenemos información general, pero no sabemos cómo se comporta en cada uno de los puntos. Un punto de partida es el teorema de los valores extremos absolutos, para luego pasar al teorema de la acotación, ambos componentes esenciales del teorema del valor intermedio (de hecho, este es un resultado de estos). Una vez reconocido eso entendremos que **si se puede tomar otro movimiento que se comporta como una subsucesión de cada punto de la sucesión general (la función del movimiento parabólico), entonces podremos también aplicar esta metodología.**

En términos simples, la caída libre que va en la misma dirección que el proyectil en la parábola se comporta de igual manera que un movimiento parabólico, solo que comprende la trayectoria desde que la altura es máxima (y velocidad cero), hasta cuando el proyectil toca el piso. En condiciones ideales donde estos proyectiles son lanzados al aire (y no hay resistencia) podemos notar que

1. La caída libre y el lanzamiento del proyectil desde una misma altura tardan lo mismo en llegar al suelo
2. Tanto en esta parábola como semiparábola, la masa es independiente de la trayectoria.

Como resultante la función que describe el tiempo que tarda en llegar a la altura máxima el cuerpo, es el mismo que tarda en llegar desde el punto máximo hacia el suelo, y por consiguiente, incluso basta con calcular una semiparábola de caída libre y obtendremos el mismo resultado respecto a los momentos de la función. La única diferencia es **el intervalo de análisis, algo que recalcamos bastante al inicio.** Así, en el caso de la parábola tomamos todo el intervalo comprendido entre sus raíces, mientras que en la semiparábola, la cota inferior corresponderá a la altura dada del problema (que debe coincidir con la máxima de la parábola completa).

En síntesis, si la caída libre sigue la misma trayectoria de lanzamiento que el movimiento parabólico, podemos llegar a los mismo resultados. Cosa distinta es si la caída libre es totalmente independiente, pues en ese caso, tendremos que evaluar la función objetivo nuevamente con la misma rigurosidad con que lo hicimos para el caso del movimiento parabólico.

## Apéndice

1. Resolvamos x en los números reales

$$-4.9x^2 + 4.5x + 50 = 0$$

$$x^2 - \frac{45}{49}x - \frac{500}{49} = 0$$

$$\left(x - \frac{45}{98}\right)^2 = \frac{100025}{9604}$$

2. Tomemos el cuadrado por ambos lados

$$x_1^* = \frac{45}{98} + \frac{5\sqrt{4001}}{98} \approx 3.6843 \quad x_2^* = \frac{45}{98} - \frac{5\sqrt{4001}}{98} \approx -2.7668$$

# Funciones de Impulso Respuesta

Nombre: Valentina Andrade

Profesor: Alexandre Janiak

Ayudantes: Pablo Vega y Bianca Hincapié

```
clear;
close all;
% Global settings
warning('off'); % warnings about rank
figure('Renderer', 'painters', 'Position', [10 10 900 600]); % figures size
```

En esta pregunta se estudiarpá superficialmente las Funciones de Impulso Respuesta, aplicación utilizada en series de tiempo que permiten comprender cuanto afecta un shock puramente transitorio al estado estacionario de una serie

Una forma de entender en qué consiste esta aplicación implicarecordar un proceso autoregresivo de orden 1 AR(1) del tipo

$$x_t = \phi x_{t-1} + \epsilon_t \quad (1)$$

Donde  $\epsilon_t \sim N(0, \sigma^2_\epsilon)$  es un ruido blanco. Se dice que la serie es estacionaria cuando  $|\phi| < 1$ , es decir la serie converge a un valor de estado estacionario. La velocidad de convergencia de la serie depende del valor de  $\phi$ , por lo cual mientras este valor sea más cercano a 1, más lenti será su retorno al estado estacionario. Por el contrario, cuando  $|\phi|$  es un valor cercano a cero, el proceso converge rápidamente al estado estacionario. Además se dice que hay una raíz unitaria cuando  $\phi = 1$ .

## Contexto

Imagine dos series idénticas que describen un proceso AR(1) descrito por la ecuación (1). Vamos a perturbar una de las dos series en un momento  $t$ , sumando un shock de magnitud  $\alpha$  al residuo  $\epsilon^t$  de la serie. El resto de los periodos permanece igual, es decir, los residuos de la serie serán iguales excepto en el periodo perturbado. Dado que las series eran idénticas antes de la perturbación, tenemos la serie y su **contrafactual**, por lo cual podemos **conocer la magnitud del efecto del shock transitorio restando ambas series**.

(a) Simule una serie AR(1) idéntica a (1) de 500 periodos con  $\phi = 0.9$ . De la misma forma simule la misma serie, pero ahora pertúrbela con un shock transitorio  $\epsilon = 1$  en el periodo  $t = 0$ . Grafique la IRF.

```
time = 500; %500 periodos
phi = 0.9; % persistencia
error = randn(time,1); % mean = 0, sd = 1
% Primera serie
x1 = NaN*ones(time,1); % Serie 1
x1(1) = 0 + error(1);
for t = 2:time
    x1(t)= phi*x1(t-1) + error(t);
end
```

```

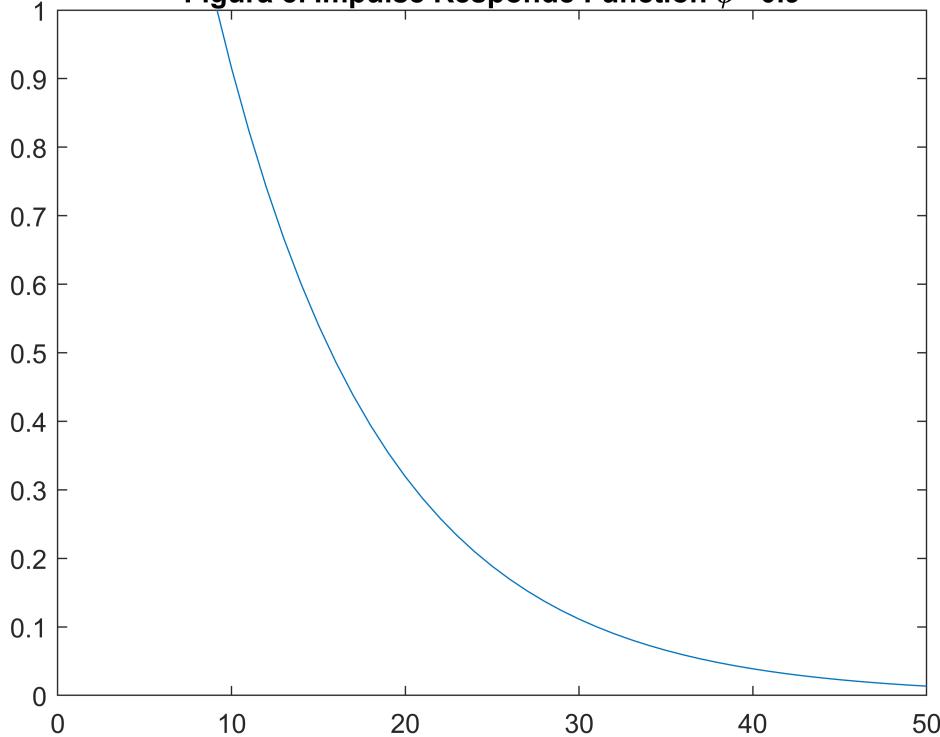
figure(1)
plot(x1)
title({'Figura 1. AR(1) sin shock'})

%Segunda serie
x2 = NaN*ones(time,1); % Serie 2
x2(1) = 0 + 1; % error
for t = 2:time
x2(t)= phi*x2(t-1) + error(t);
end
figure(2)
plot(x2)
title({'Figura 2. AR(1) con shock'})

%%Graficar IRF
figure(3)
plot(x2-x1)
xlim([0 50])
ylim([0 1])
title({'Figura 3. Impulse Responde Function \phi =0.9'})
hold off

```

**Figura 3. Impulse Responde Function  $\phi =0.9$**



(b) Construya un subplot para distintos coeficientes autoregresivos  $\phi \in (0, 1)$  dados por el vector [6x1]

$$\phi = [0.1 \ 0.2 \ 0.4 \ 0.6 \ 0.8 \ 0.95]'$$

```

phi = [0.1 0.2 0.4 0.6 0.8 0.95]'; % persistencia
time = 500; %500 periodos
error = randn(time,1); % mean = 0, sd = 1
for i = 1:6
    x1 = NaN*ones(time,1); % Serie 1
    x1(1) = 0 + error(1);
    for t = 2:time
        x1(t)= phi(i)*x1(t-1) + error(t);
    end
    x2 = NaN*ones(time,1); % Serie 2
    x2(1) = 0 + 1; % error
    for t = 2:time
        x2(t)= phi(i)*x2(t-1) + error(t);
    end
    subplot(3,2,i)
    plot(x2-x1)
    xlim([0 50])
    ylim([0 1])
    title(['Perturbacion \phi =', num2str(phi(i))])
end

```

Antes de la interpretación es necesario precisar algunos conceptos y nociones útiles para el análisis. En los modelos VAR (vector autoregressive) existe un caso especial de sistemas dinámicos que son las funciones de **impulso respuesta (IRF)**. Su particularidad en el estudio de series de tiempo está en que sirven para estudiar el comportamiento de variables endógenas una vez que se ha aplicado un shock exógeno ya sea permanente o transitorio (Sargent y Ljungqvist, 2000, p.48) Así, IRF permite describir la evolución del modelo VAR a partir de la reacción que este representa a partir del shock en una o varias de sus variables.

En ese sentido, es evidente que uno de sus momentos estadísticos a analizar, es la transmisión de un shock hacia todo el sistema de ecuaciones que define el problema macroeconómico. Así el modelo se define en términos generales de la siguiente manera:

$$X_t = c + \sum_{i=0}^{\infty} D_i Y_{t-i} + \sum_{i=0}^{\infty} \phi x_{t-i}$$

Donde  $Y$  representa un set de variables exógenas y  $D_i$  una matriz un multiplicador dinámico de las funciones que describen los shocks, que son conocidos como funciones de transferencia (Stockey and Lucas, 1989, p. 283). Además,  $\phi$  representa la función de impulso respuesta en el horizonte  $i$  (Koop et al, 1996) . En el contexto tradicional esta función puede ser representada de la siguiente forma

$$I_x(n, \delta, \phi) = \delta \frac{1 - \phi^{n+1}}{1 - \phi}$$

Precisamente lo que muestran los subplots son distintos valores de  $\delta$ . Según un texto clásico de Koop (1996) esta medida no cambia la forma de la función sino que escala la medida, lo que es una medida restrictiva de modelos lineales en general. Luego, tal como se muestra en nuestros gráficos, los modelos AR(1) y ARIMA tienen funciones de impulso respuesta tienen la propiedad de ser **siméticas** (esto es, un shock en 1 tiene un efecto exactamente opuesto a un shock -1) y de que sus **shock son lineales** (i.e, un shock de tamaño 2 tiene un efecto exactamente del doble que un shock de 1). Además, se tiene la propiedad de que el pasado no afecta el efecto de la respuesta (*history independence property*). Las últimas dos propiedades se ven de manera clara en la figura, dado que la forma funcional no cambia debido a los shocks (history independence)

y los shocks afectan a todas las funciones de forma igual (mismo  $\delta$ ) y lo único que varía es  $\frac{1 - \phi^{n+1}}{1 - \phi}$  donde

los  $\phi$  mayores presentan una mayor persistencia o resistencia al shock. En el último apartado haremos una discusión analítica de su importancia en el análisis del PIB.

(c) Descargue la serie del PIB per cápita de Chile desde los datos del Banco Mundial (NY.GDP.PCAP.KN).

Los siguientes datos han sido descargados directamente desde la API del Banco Mundial, a partir del acceso a sus bases de datos. Para establecer la conexión web se debe descargar la [siguiente herramienta](#). Esto permite acceder a datos desde 1960 a 2020, a diferencia de los datos públicos que están desde 2000.

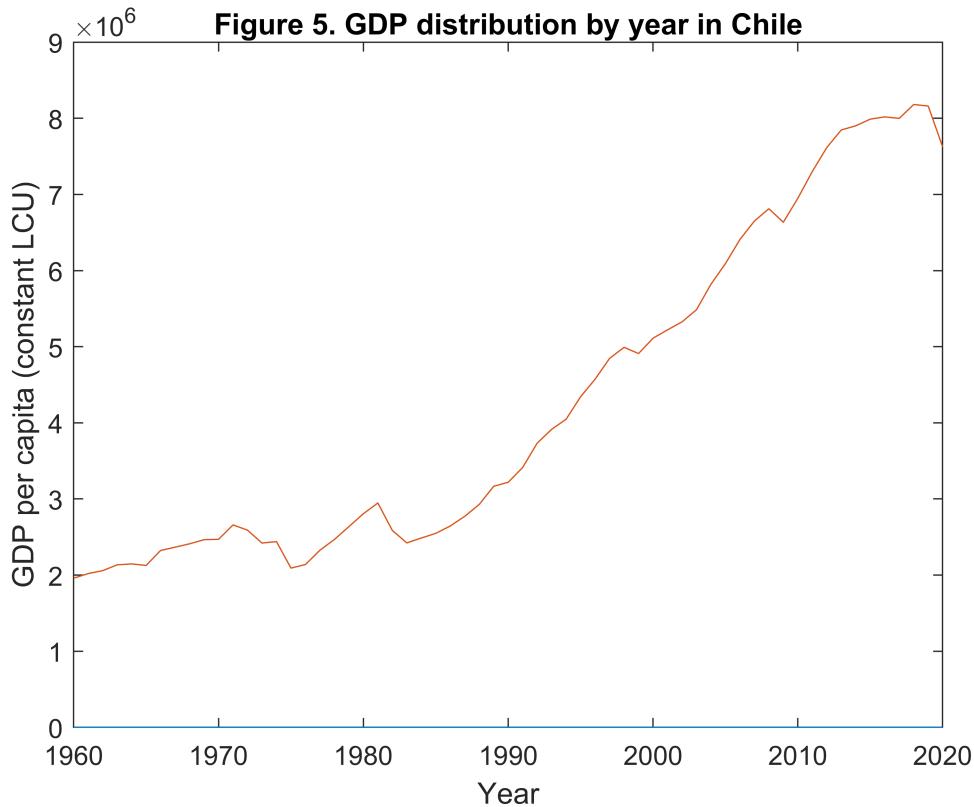
```
clear; close all;
% Conectar con web api.
conn = wb();

% Importar data del pib per capital con los filtros sql necesarios
gdp_data = query(conn,'source','2',...
    'series','NY.GDP.PCAP.KN',...
    'country','CHL',...
    'time','all');

% Traer tabulado
raw_data = {gdp_data.source.data.value};

% Manipular datos
PIB = flip(transpose(raw_data));
year = [1960:2020]';
PIB = [ year cell2mat(PIB)];

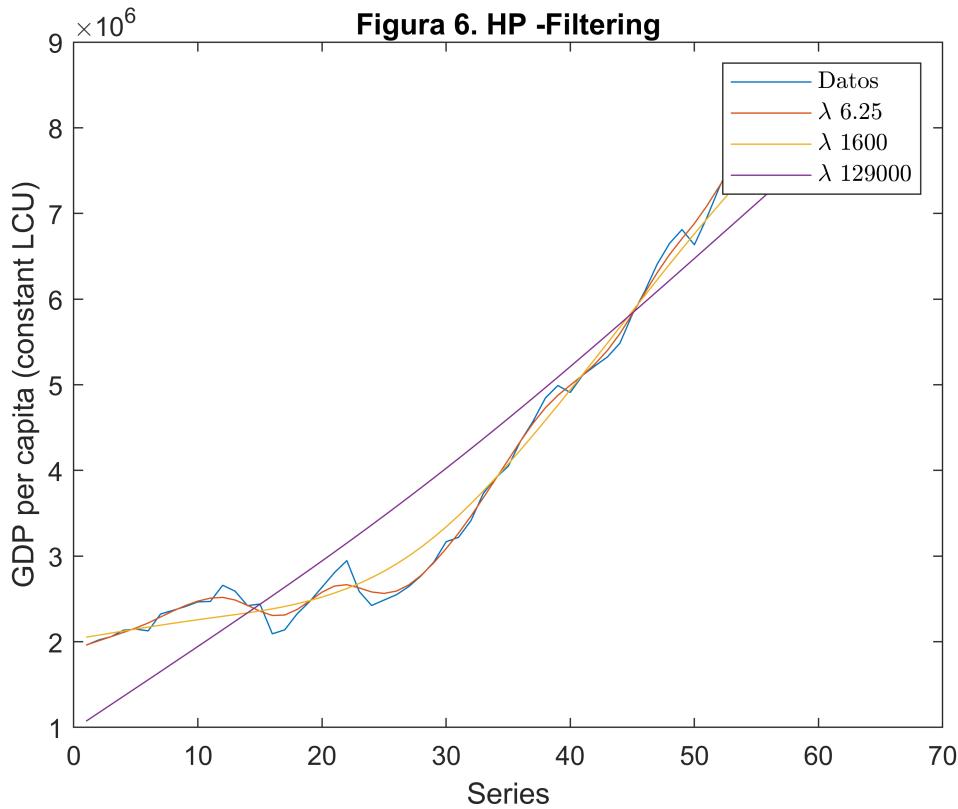
% Grafico para explorar datos
figure(12)
plot(year,PIB)
% Create ylabel
ylabel({'GDP per capita (constant LCU)'})
% Create xlabel
xlabel({'Year'})
% Create title
title({'Figure 5. GDP distribution by year in Chile'})
hold off
```



(d) Obtenga a tendencia y el ciclo de la serie utilizando el filtro Hodrick-Prescott estudiado en preguntas anteriores. Puede reutilizar las funciones creadas previamente si lo desea.

Se utilizará el filtro construido en la pregunta anterior, junto con graficar las tendencias según distintos niveles de sensibilidad.

```
lambda = [6.25 1600 129000];
% Dado que no se recomienda renombrar dinámicamente en matlab se crea cada
% objeto a mano
[ytrend625, ycycle625] = hp_filter(PIB,lambda(1));
[ytrend16, ycycle16] = hp_filter(PIB,lambda(2));
[ytrend129, ycycle129] = hp_filter(PIB,lambda(3));
graph = [PIB(:,2) ytrend625(:,2) ytrend16(:,2) ytrend129(:,2)]; % Matriz con variables de cobre
hpfigure(graph) % Grafico
```



(e) Mediante OLS, compute el coeficiente autoregresivo ausmiendo que la serie sigue un proceso AR(1).

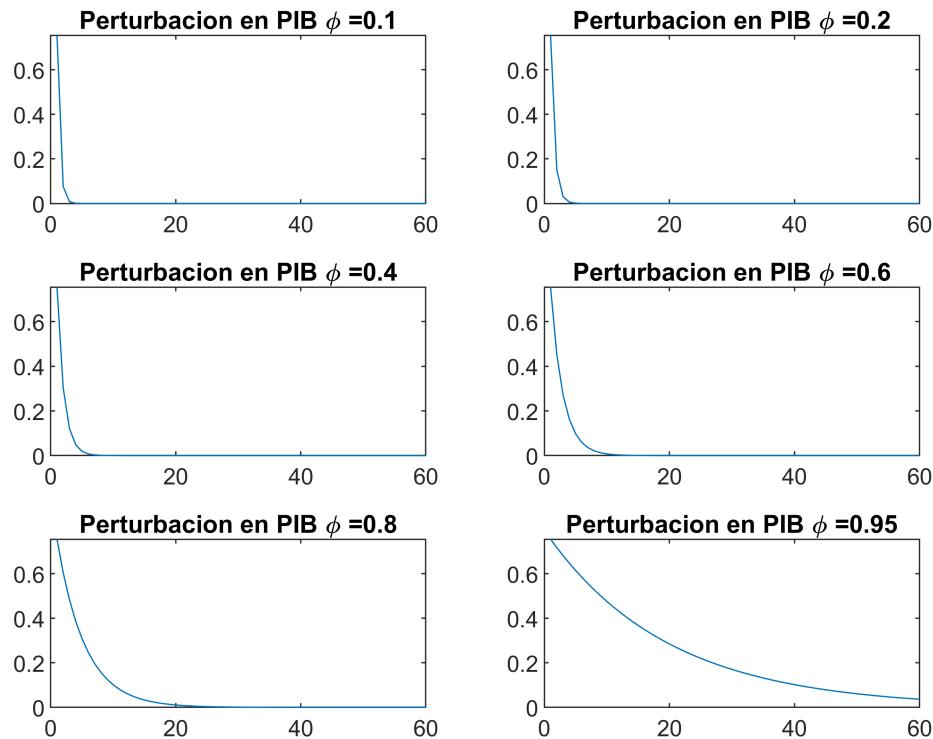
```
% Dimensiones
y = ycycle625(1:end,2);
X = y(1:end-1);
y = y(2:end);

% OLS and statistics
n = length(y);
k = 2;
X = [ones(length(X), 1) X];
beta = (X\y)';
e = y - X.*beta;
s2 = e'*e/ (n-k);
se = sqrt(s2*diag(inv(X'*X)));
t = beta./se;
rmse = sqrt(s2);
p = 2*(1-tcdf(abs(t), length(y) - k));
```

Tal como aparecen en los outputs el  $\hat{\phi} \approx 0.2459$ , valor que fue rectificado con la función fitlm. En base a la discusión que presentamos anteriormente, podremos notar que la función de impulso respuesta en este caso depende principalmente de  $\delta$ . Ahora profundizaremos para esta tendencia que incidencia tiene distintos niveles de persistencia sobre la función de impulso respuesta.

(f) Compute una IRF asumiendo un shock transitorio  $\epsilon = 1$  en t=0 sobre el residuo de la serie

```
phi = [0.1 0.2 0.4 0.6 0.8 0.95]'; % persistencia
time = length(y); %60 periodos
error = randn(time,1); % mean = 0, sd = 1
y2 = y ;
for i = 1:6
    y(1) = 0 + error(1);
    for t = 2:time
        y(t)= phi(i)*y(t-1) + error(t);
    end
    y2(1) = 0 + 1; % shock 1
    for t = 2:time
        y2(t)= phi(i)*y2(t-1) + error(t);
    end
end
figure(15)
subplot(3,2,i)
plot(y2-y)
title(['Perturbacion en PIB \phi =', num2str(phi(i))])
hold off
end
```



(g) ¿Cómo cambian los resultados al variar la persistencia del proceso? Grafique y explique

En el punto anterior generamos un algoritmo que permitió obtener la función de impulso respuesta, esto es,  $\Delta y = y_{\epsilon=1} - y_{\epsilon \sim N(0, \sigma^2)}$ , es decir, la comparación entre la tendencia del PIB con un shock transitorio y el PIB con su serie filtrada con HP filter. Como discutimos anteriormente, si el proceso puede ser modelado a partir de un ARIMA se cumplirán tres propiedades

1. History independence property
2. Linearity
3. Symmetry

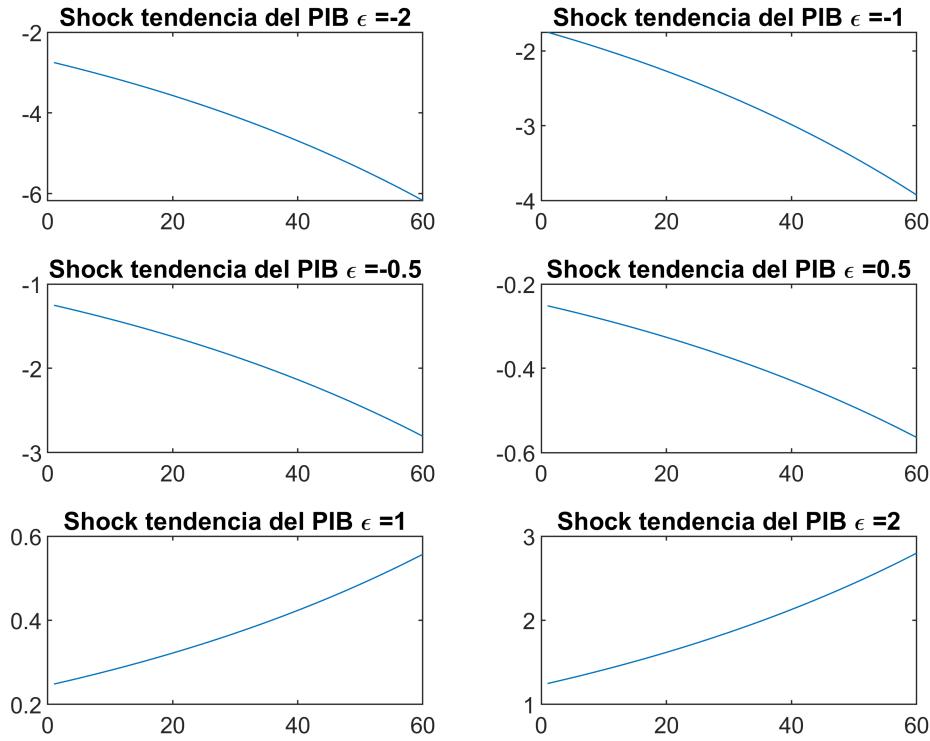
Hasta ahora hemos podido abordar las primeras dos, toda vez que refieren a la relación que tienen las variables endógenas con su pasado y la diferencia que producen diferentes niveles de persistencia. Ahora bien, en este apartado avanzaremos hacia una discusión más analítica. Lo interesante de este ejercicio es notar que una vez que se desestacionaliza la serie de PIB, si comparamos el PIB sin shock transitorio y con shock transitorio (IRF) obtendremos que su capacidad de respuesta dependerá del  $\phi$ . Cuando la serie es débilmente persistente, la recuperación respecto al shock es bastante rápida, mientras que cuando la perturbación es más grande ( $\phi \approx 0.95$ ) la recuperación es más lenta. Como podemos ver en las figuras, incluso en 60 años de serie, sigue existiendo un ruido de este shock. El resultado no es trivial: en el apartado anterior llegamos a la conclusión de que  $\hat{\phi} \approx 0.2459$  por lo que la IRF dado el shock transitorio se recupera rápidamente en el tiempo (converge). Ahora bien, es probable que estos resultados cambien si en esta estimación de igual manera incluimos el efecto del ciclo. Mejoras en el análisis podrían incluir, en este caso, la inclusión de medias móviles y análisis de cointegración dada la tendencia de la serie.

(h) ¿Cómo cambian los resultados anteriores al variar la magnitud del shock transitorio? Grafique y explique.

```

shock = [-2 -1 -0.5 0.5 1 2]'; % persistencia
time = length(y); %60 periodos
phi = 1.01378;
error = randn(time,1); % mean = 0, sd = 1
y2 = y ;
for i = 1:6
    y(1) = 0 + error(1);
    for t = 2:time
        y(t)= phi*y(t-1) + error(t);
    end
    y2(1) = 0 + shock(i); % shock 1
    for t = 2:time
        y2(t)= phi*y2(t-1) + error(t);
    end
end
figure(16)
subplot(3,2,i)
plot(y2-y)
title(['Shock tendencia del PIB \epsilon = ', num2str(shock(i))])
hold off
end

```



Con este último ejercicio demostramos una de las propiedades que habíamos destacado: los modelos AR(1) y ARIMA tienen funciones de impulso respuesta tienen la propiedad de ser **siméticas** (esto es, un shock en 1 tiene un efecto exactamente opuesto a un shock -1). Notemos que el subplot del shock de -2 y 2 son simétricos, así cada uno correspondientemente. Entonces, si bien antes discutimos sobre la **magnitud de la persistencia y su relación con el proceso autoregresivo** (esto es, la propiedad lineal y de independencia, respectivamente), ahora podemos mostrar que también se cumple la propiedad de simetría. Como señalaron Koop (1996, p. 123) en "*Impulse response analysis in nonlinear multivariate models*" este tipo de propiedades son tradicionales a las funciones de impulso respuesta, y que incluso han sido probadas empíricamente en investigaciones clásicas como las de Campbell y Mankiw (1987) cuando normalizaron el  $\delta$  (el que nos da el nivel o escala del shock).

En relación al resultado obtenido respecto del PIB, evidentemente existe un sesgo en el resultado dado la desestacionalización de la serie, y de que asumimos un modelamiento lineal. De hecho, en nuestros resultado ya podemos ver algunos problemas, toda vez que la persistencia no resulta significativa en toda la serie, por lo que probablemente se deban explorar otros modelos probabilísticos que se adequen mejor la la forma de la distribución. Entonces, si bien la simetría es evidente y cumple con la versión tradicional de IRF, este resultado se da **dado que elegimos un modelamiento lineal**. Como mencionan investigaciones posteriores, el modelamiento del PIB puede tener como resultado asimetrías en la medida en que se recosnidera el modelamiento hacia uno no lineal y que considera el ciclo de la serie (véase en Cancelo y Mourelle, 2005)

Ahora bien, no siempre estas propiedades se mantienen en modelos no lineales, dado que esto también dependerá del signo de la IRF, pues en algunos casos shocks positivos no producirán cambios positivos, y con

ello la propiedad de simetría se perderá y el modelamiento sobre el futuro se hará más difícil (encontrar un contrafactual perfecto), y la dependencia del shock y pasado serán problemas.

## Referencias

- Cancelo, J. R., & Mourelle, E. (2005). Modeling cyclical asymmetries in GDP: international evidence. *Atlantic Economic Journal*, 33(3), 297-309.
- Koop, G., Pesaran, M. H., & Potter, S. M. (1996). Impulse response analysis in nonlinear multivariate models. *Journal of econometrics*, 74(1), 119-147.
- Sargent, T. J., & Ljungqvist, L. (2000). Recursive macroeconomic theory. *Massachusetts Institute of Technology*.
- Stokey, N. L. (1989). *Recursive methods in economic dynamics*. Harvard University Press.