1. Considere el proceso $AR(2)$:
$$X_t - 0.4X_{t-1} - 0.45X_{t-2} = Z_t$$
donde $\{Z_t\}$ es un ruido blanco.

   a) Encuentre las raíces del polinomio autoregresivo.
   Solución: Consideremos que
$$Z_t = X_t - 0.4X_{t-1} - 0.45X_{t-2} = (1 - 0.4\mathbf{B} - 0.45\mathbf{B}^2)X_t$$
   por lo que el polinomio autoregresivo está dado por $\phi(\mathbf{B}) = 1 - 0.4\mathbf{B} - 0.45\mathbf{B}^2$, cuyas raíces son $\mathbf{B}_1 = \frac{10}{9}$ y $\mathbf{B}_2 = -2$. De tal, manera podemos considerar la factorización
$$\phi(\mathbf{B}) = \left(1 - \frac{9}{10}\mathbf{B}\right)\left(1 + \frac{1}{2}\mathbf{B}\right)$$
   Luego, por la Proposición 2 de *Causalidad*, podemos concluir que el proceso $\phi(\mathbf{B})X_t = Z_t$ es causal dado que $|\mathbf{B_1}|, |\mathbf{B_2}| > 1$.

   b) Calcule σ_X^2 suponiendo que $\sigma_Z^2 = 2$.
   Solución: Considerando la factorización dada en el inciso anterior del polinomio autoregresivo, tenemos que
$$X_t = \frac{Z_t}{\left(1 - \frac{9}{10}\mathbf{B}\right)\left(1 + \frac{1}{2}\mathbf{B}\right)}$$
   Por ende, procederemos a emplear fracciones parciales para descomponer la fracción $1/\left(1 - \frac{9}{10}\mathbf{B}\right)\left(1 + \frac{1}{2}\mathbf{B}\right)$, con lo cual se obtiene que
$$\frac{1}{\left(1 - \frac{9}{10}\mathbf{B}\right)\left(1 + \frac{1}{2}\mathbf{B}\right)} = \frac{\frac{9}{14}}{\left(1 - \frac{9}{10}\mathbf{B}\right)} + \frac{\frac{5}{14}}{\left(1 + \frac{1}{2}\mathbf{B}\right)}$$
   Por consiguiente deducimos la siguiente expresión para X_t:
$$X_t = \sum_{i=0}^{\infty}\left(\frac{9}{14}(0.9)^i + \frac{5}{14}(-0.5)^i\right)Z_{t-i}$$
   la cual nos permite dar una expresión para calcular σ_X^2:
$$\sigma_X^2 = \sigma_Z^2 \cdot \sum_{i=0}^{\infty}\left(\frac{9}{14}(0.9)^i + \frac{5}{14}(-0.5)^i\right)^2 = 2 \cdot \sum_{i=0}^{\infty}\left(\frac{9}{14}(0.9)^i + \frac{5}{14}(-0.5)^i\right)^2$$
   Podemos obtener un valor explícito para dicha varianza, lo que haremos a continuación utilizando Python.
   Lo que haremos será crear una función que reciba como parámetro un valor n para calcular la suma $2 \cdot \sum_{i=0}^{n}\left(\frac{9}{14}(0.9)^i + \frac{5}{14}(-0.5)^i\right)^2$. Para ello

```python
def suma_parcial(n):
  list_suma = []
  for i in range(0,n):
    # Calculamos un valor de la suma para un i
    valor = 0.643 * (0.9) ** i + 0.3571 * (-0.5) ** i
    # Elevamos el valor anterior al cuadrado
    valor2 = valor ** 2
    list_suma.append(valor2)
  # Calculamos la suma hasta el termino n
  suma = sum(list_suma)
  # Mostramos la aproximacion del valor de la varianza de x
  print(2 * suma)
```

Ahora, podemos ver los siguientes valores de aproximación para σ_X^2, donde, evidentemente, mientras mayor sea n mejor será la aproximación

```
suma_parcial(10)
suma_parcial(100)
suma_parcial(1000)
suma_parcial(10000)

## Output:
## 4.796241553872182
## 5.325570677679712
## 5.325570680750147
## 5.325570680750147
```

de tal manera, podemos decir que $\sigma_X^2 \approx 5.3255$

c) Encuentre la expresión general para la función de autocorrelación ρ_k.
Solución: Considerando $X_t = 0.4X_{t-1} + 0.45X_{t-2} + Z_t$, multiplicamos por X_{t-k} en ambos lados de dicha ecuación, donde $k \in \mathbb{N}$ es arbitrario. Por ende

$$X_t X_{t-k} = 0.4X_{t-1}X_{t-k} + 0.45X_{t-2}X_{t-k} + Z_t X_{t-k}$$

y entonces:

$$\begin{aligned} &(1) \quad \mathbb{E}\left[X_t X_{t-k}\right] = 0.4\mathbb{E}\left[X_{t-1}X_{t-k}\right] + 0.45\mathbb{E}\left[X_{t-2}X_{t-k}\right] + \mathbb{E}\left[Z_t X_{t-k}\right] \\ &(2) \quad \gamma(k) = 0.4\gamma(k-1) + 0.45\gamma(k-2) \\ &(3) \quad \rho(k) = 0.4\rho(k-1)k + 0.45\rho(k-2) \end{aligned}$$

donde:

(1) Aplicamos esperanza de ambos lados de la ecuación.
(2) Usamos que $\gamma(k) = \mathbb{E}[X_t X_{t-k}]$. Además, ocupamos que $\mathbb{E}[Z_t X_{t-k}] = 0$
(3) Usamos que $\frac{\gamma(k)}{\gamma(0)} = \rho(k)$

Siendo la expresión en la ecuación (2) la forma general de la función de autocorrelación ρ_k, $k \in \mathbb{N}$.

d) Grafique ρ_k para $k = 0, 1, \ldots, 10$.
Solución: Para resolver este ejercicio utilizaremos Python. Sabemos que $\rho_0 = 1$ y podemos calcular ρ_1 fácilmente como sigue:

$$\begin{aligned} \rho(1) &= 0.4\rho(0) + 0.45\rho(-1) = 0.4 \cdot 1 + 0.45\rho(1) \\ \Rightarrow \quad \rho(1) &= \frac{0.4}{1.45} = \frac{8}{11} \approx 0.7272 \end{aligned}$$

Ahora bien, construiremos una función que obtenga los valores $\rho_2, \ldots, \rho_{10}$, para ello

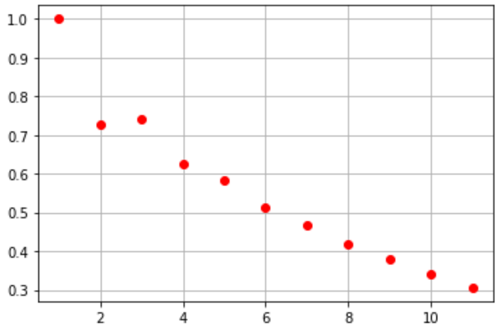
```
# Creamos una lista con los valores rho_0 y rho_1:
rho = [0, 1, 0.7272]

# Creamos una funcion para calcular rho_k, con k>1 arbitrario:
def cal_rho(k):
  n = k
  while n >= k:
    resultado = 0.4 * rho[n] + 0.45 * rho[n - 1]
    rho.append(resultado)
    n -= 1
```

Luego, obtenemos el resto de valores de *rho* utilizando la función que creamos:

```
for i in range(2,11):
  cal_rho(i)
print(rho[2:])

## Output:
## [0.7272, 0.74088, 0.6235919999999999, 0.5828328, 0.51374952,
↪  0.46777456800000006, 0.4182971112, 0.37781740008000003, 0.33936066007200005,
↪  0.30576209406480004] 0.0185815322032]
```

Posteriormente, realizaremos la gráfica de ρ_k para $k = 0, \ldots, 10$, para lo cual también crearemos una función:

```
# Importamos la libreria necesaria
import matplotlib.pyplot as mplot

# Graficamos cada punto de la lista phi
for i in range(0,12):
  mplot.plot(i, rho[i], marker="o", color="r")

mplot.grid()
mplot.show()
```

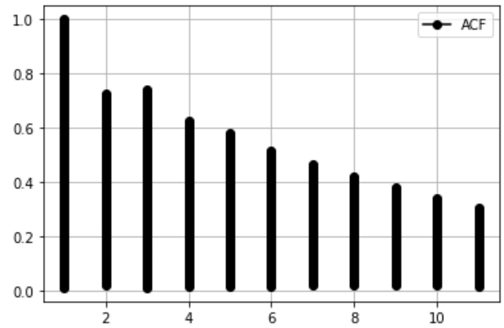


Lo que haremos a continuación será dibujar las rectas verticales para cada punto:

```
# Primero crearemos la siguiente funcion, la cual dibujara dichas rectas
    verticales:

def graf(x,p):
    if rho[p] > 0:
        rango = [ rho[p] - 0.01 * i for i in range(int(rho[p] * 100))]
        for y in rango:
            mplot.plot(x, y, marker="o", color="black")
    elif rho[p] < 0:
        rank = abs(rho[p]) * 100
        rango2 = [ rho[p] + 0.01 * i for i in range(int(rank))]
        for y in rango2:
            mplot.plot(x, y, marker="o", color="black")
    else:
        mplot.plot(0, 0, marker="o", color="black")
```

Posteriormente, utilizamos la función anterior para obtener el resultado esperado:

```
for i in range(1,12):
  graf(i,i)
mplot.plot(10, 0, marker="o", color="black")
mplot.grid()
mplot.show()
```



Por otra parte, podemos hallar el valor de σ_X^2 de otra forma a como lo hicimos en el inciso *b*), lo cual lograremos empleando lo obtenido en el inciso *c*) y el valor de $\rho(2)$ obtenido en los cálculos anteriores. Procedemos considerando que

$$\gamma(0) = 0.4\rho(1)\gamma(0) + 0.45\rho(2)\gamma(0) + \sigma_Z^2 \qquad (*)$$

dado que sustituimos el valor $\gamma(k) = \rho(k)\gamma(0)$ en la ecuación $\gamma(0) = 0.4\gamma(1) + 0.45\gamma(2) + \sigma_Z^2$. Luego, despejamos $\gamma(0)$ de la ecuación $(*)$ para conseguir que

$$\sigma_X^2 = \gamma(0) = \frac{\sigma_Z^2}{1 - 0.4\rho(1) - 0.45\rho(2)}$$

y dado que $\rho_1 = 0.7272$, $\rho_2 = 0.7409$ y $\sigma_Z^2 = 2$, entonces se concluye que

$$\sigma_X^2 = \frac{2}{1 - 0.4(0.7272) - 0.45(0.7409)} = 5.3231$$

e) Encuentre la expresión general para la función de autocorrelación parcial ϕ_{kk}.
Solución: Dicha expresión está dada por

$$\phi_{kk} = \frac{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-2} & \rho_1 \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_1 & \rho_k \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-2} & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_{k-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_1 & 1 \end{vmatrix}}$$

extraído de las notas de clase de la profesora.

Ahora, dado que estamos trabajando con un proceso $AR(2)$ tendremos que $\phi_{kk} = 0$ para $k > 2$, lo cual corroboraremos en el siguiente inciso al realizar las cuentas.

f) Grafique ϕ_{kk} para $k = 0, \ldots, 10$.

Solución: Sabemos que $\rho_1 = \phi_{11}$ por lo que $\phi_{11} \approx 0.7272$. Ahora, para hallar el resto de los valores de ϕ_{kk} utilizaremos la expresión dada en el inciso anterior y realizaremos los siguientes cálculos en Python, donde nos valdremos en este caso de la librería *numpy*:

```python
# Importamos la libreria que ocuparemos
import numpy as np

# Para phi_22
vphi2_num = np.array([ [1, phi[2]],
                       [phi[2], phi[3] ])
vphi2_den = np.array([ [1, phi[2]],
                       [phi[2], 1] ])
vphi2 = np.linalg.det(vphi2_num) /  np.linalg.det(vphi2_den)
```

Repetiremos el proceso anterior hasta hallar el valor de ϕ_{kk} para $k = 10$.

```python
# Para phi_33
vphi3_num = np.array([ [1, phi[2], phi[2]],
                       [phi[2], 1, phi[3]],
                       [phi[3], phi[2], phi[4]] ])
vphi3_den = np.array([ [1, phi[2], phi[3]],
                       [phi[2], 1, phi[2]],
                       [phi[3], phi[2], 1] ])
vphi3 = np.linalg.det(vphi3_num) /  np.linalg.det(vphi3_den)
```

Para ϕ_{44}:

```python
vphi4_num = np.array([ [1, phi[2], phi[3], phi[2]],
                       [phi[2], 1, phi[2], phi[3]],
                       [phi[3], phi[2], 1, phi[4]],
                       [phi[4], phi[3], phi[2], phi[5]] ])
vphi4_den = np.array([ [1, phi[2], phi[3], phi[4]],
                       [phi[2], 1, phi[2], phi[3]],
                       [phi[3], phi[2], 1, phi[2]],
                       [phi[4], phi[3], phi[2], 1] ])
vphi4 = np.linalg.det(vphi4_num) /  np.linalg.det(vphi4_den)
```

Para ϕ_{55}:

```
vphi5_num = np.array([ [1, phi[2], phi[3], phi[4], phi[2]],
                       [phi[2], 1, phi[2], phi[3], phi[3]],
                       [phi[3], phi[2], 1, phi[2], phi[4]],
                       [phi[4], phi[3], phi[2], 1, phi[5]],
                       [phi[5], phi[4], phi[3], phi[2], phi[6]]])
vphi5_den = np.array([ [1, phi[2], phi[3], phi[4], phi[5]],
                       [phi[2], 1, phi[2], phi[3], phi[4]],
                       [phi[3], phi[2], 1, phi[2], phi[3]],
                       [phi[4], phi[3], phi[2], 1, phi[2]],
                       [phi[5], phi[4], phi[3], phi[2], 1] ])
vphi5 = np.linalg.det(vphi5_num) /  np.linalg.det(vphi5_den)
```

Para ϕ_{66}:

```
vphi6_num = np.array([ [1, phi[2], phi[3], phi[4], phi[5], phi[2]],
                       [phi[2], 1, phi[2], phi[3], phi[4], phi[3]],
                       [phi[3], phi[2], 1, phi[2], phi[3], phi[4]],
                       [phi[4], phi[3], phi[2], 1, phi[2], phi[5]],
                       [phi[5], phi[4], phi[3], phi[2], 1, phi[6]],
                       [phi[6], phi[5], phi[4], phi[3], phi[2], phi[7]] ])
vphi6_den = np.array([ [1, phi[2], phi[3], phi[4], phi[5], phi[6]],
                       [phi[2], 1, phi[2], phi[3], phi[4], phi[5]],
                       [phi[3], phi[2], 1, phi[2], phi[3], phi[4]],
                       [phi[4], phi[3], phi[2], 1, phi[2], phi[3]],
                       [phi[5], phi[4], phi[3], phi[2], 1, phi[2]],
                       [phi[6], phi[5], phi[4], phi[3], phi[2], 1] ])
vphi6 = np.linalg.det(vphi6_num) /  np.linalg.det(vphi6_den)
```

Para ϕ_{77}:

```
vphi7_num = np.array([ [1,      phi[2], phi[3], phi[4], phi[5], phi[6], phi[2]],
                       [phi[2], 1,      phi[2], phi[3], phi[4], phi[5], phi[3]],
                       [phi[3], phi[2], 1,      phi[2], phi[3], phi[4], phi[4]],
                       [phi[4], phi[3], phi[2], 1,      phi[2], phi[3], phi[5]],
                       [phi[5], phi[4], phi[3], phi[2], 1,      phi[2], phi[6]],
                       [phi[6], phi[5], phi[4], phi[3], phi[2], 1,      phi[7]],
                       [phi[7], phi[6], phi[5], phi[4], phi[3], phi[2], phi[8]]
                       ↪  ])

vphi7_den = np.array([ [1,      phi[2], phi[3], phi[4], phi[5], phi[6], phi[7]],
                       [phi[2], 1,      phi[2], phi[3], phi[4], phi[5], phi[6]],
                       [phi[3], phi[2], 1,      phi[2], phi[3], phi[4], phi[5]],
                       [phi[4], phi[3], phi[2], 1,      phi[2], phi[3], phi[4]],
                       [phi[5], phi[4], phi[3], phi[2], 1,      phi[2], phi[3]],
                       [phi[6], phi[5], phi[4], phi[3], phi[2], 1,      phi[2]],
                       [phi[7], phi[6], phi[5], phi[4], phi[3], phi[2], 1] ])
vphi7 = np.linalg.det(vphi7_num) /  np.linalg.det(vphi7_den)
```

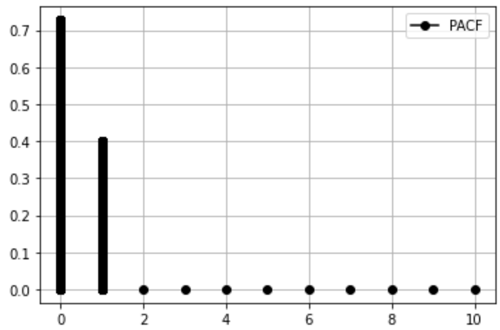
Para ϕ_{88}, ϕ_{99} y $\phi_{10\ \ 10}$ se ha omitido el código que genera dichos valores, pues son muchas más líneas que hacen que se pierda la legibilidad a la hora de la lectura de este documento, además, dichos valores serán todos cero como veremos a continuación.

Ahora, veamos todos los valores obtenidos para ϕ_{kk}:

```python
# Creamos una lista con estos valores
vphis = [vphi1,vphi2,vphi3,vphi4,vphi5,vphi6,vphi7,vphi8,vphi9,vphi10]

# Veamos
for i in vphis:
  print(i)

## Output:
## 0.7272
## 0.3999151067820851
## 4.791203900725592e-05
## 5.100552674113396e-09
## 5.4279015259217778e-13
## 0.0
## 0.0
## 0.0
## 0.0
## 0.0
```

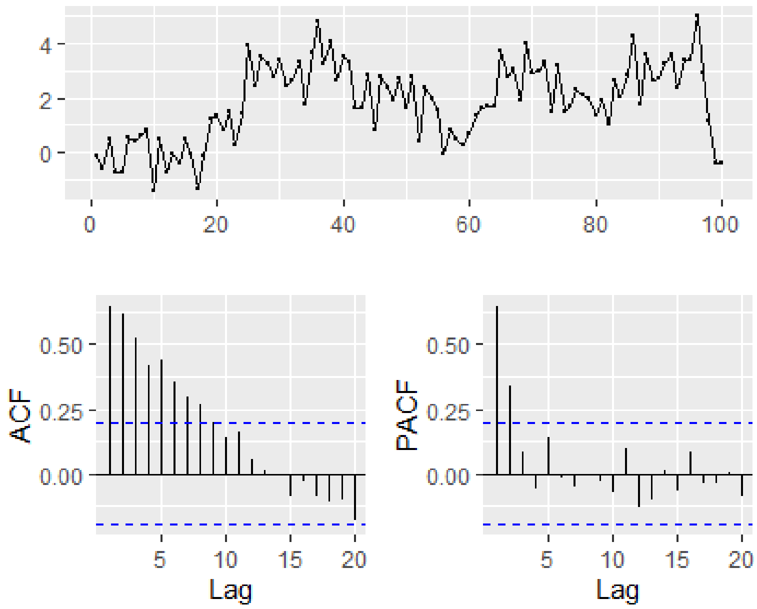
notamos que, para $k > 2$, ϕ_{kk} es muy cercano a cero (de acuerdo al cálculo empleado por Python) o igual a cero, lo cual es claro puesto que, en el modelo AR(2), los valores de las autocorrelaciones parciales se "cortan" (valen cero) después del lag 2. Finalmente, procedemos a realizar la gráfica **PACF**:

```python
# Creamos una funcion para graficar
def graf2(x,p):
        rank = abs(vphis[p]) * 10000
        rango2 = [ vphis[p] + 0.0001 * i for i in range(int(rank))]
        for y in rango2:
            mplot.plot(x, y, marker="o", color="black")
# Graficamos
for i in range(0,2):
  graf2(i,i)
for i in range(2,11):
  mplot.plot(i, 0, marker="o", color="black")

mplot.grid()
mplot.show()
```

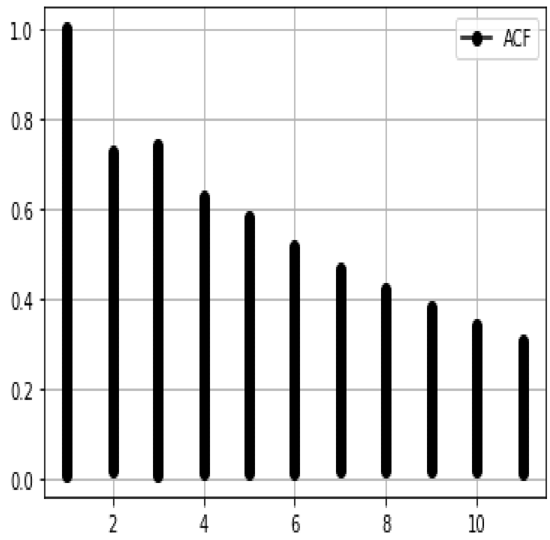


g) En R simule el proceso $\{X_t\}$ para un tamaño de muestra n, grafique la serie de tiempo y los correlogramas. Compare los correlogramas simulados con los del proceso original.
Solución: Para la simulación utilizaremos la función *arima.sim()* como sigue

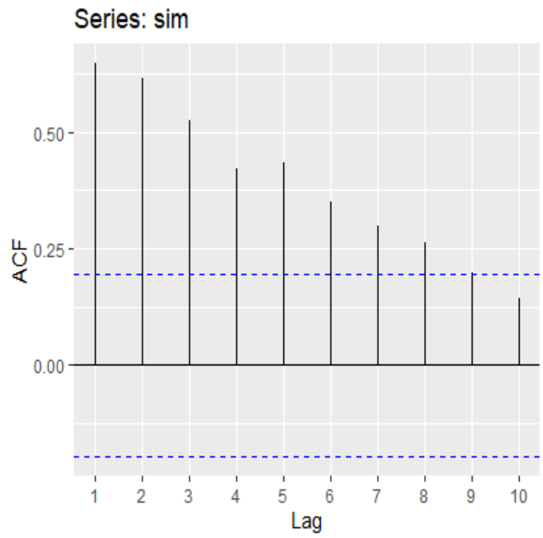
```r
# Cargamos las librerias que ocuparemos
library(forecast)
library(GGally)

# Realizamos la simulacion
sim <- arima.sim(list(order=c(2,0,0), ar=c(0.4,0.45)), n=100)

# Graficamos la serie de tiempo con los correlogramas
ggtsdisplay(sim)
```



Observemos las gráficas **ACF** del proceso original versus el proceso simulado:



(a) **ACF** del proceso original



(b) **ACF** del proceso simulado

de donde

(1) La gráfica en la figura (b) no contempla como primer valor el correspondiente a $\rho_0 = 1$; valor que si es incluido en la gráfica en la figura (a).

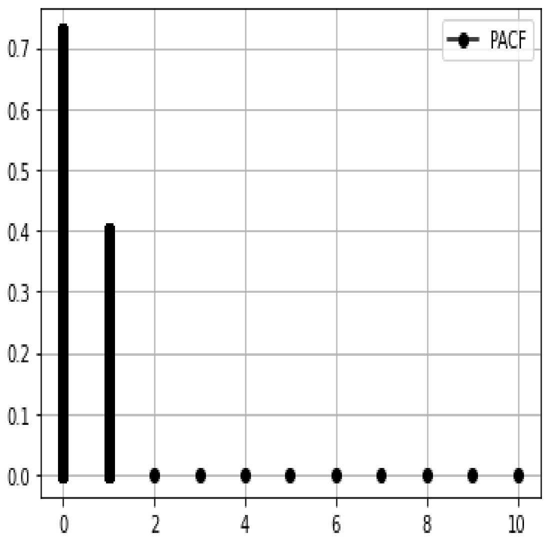
(2) Notamos un patrón muy similar en cuanto al comportamiento de los valores: los primeros dos valores (sin considerar el $lag = 1$ de la figura (a)) son casi idénticos y el resto de los valores tiene un comportamiento decreciente.

(3) Los primeros dos valores para la gráfica en (a) (sin contar el valor 1 para $lag = 1$) son 0.72 y 0.74 respectivamente; para el caso de la gráfica en (b), dichos valores son bastante próximos a los de la gráfica en (a).

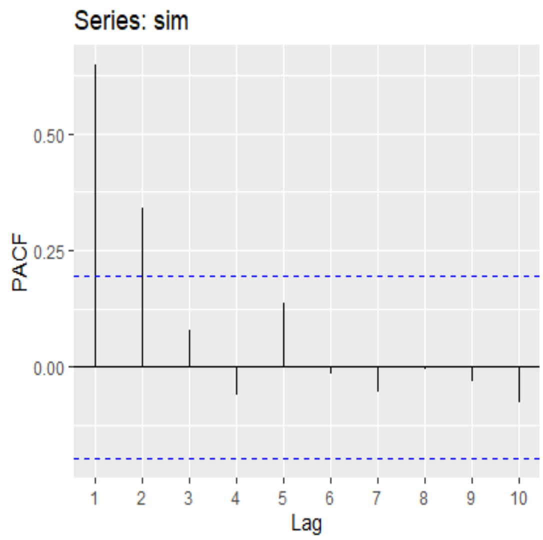
El gráfico **ACF** de la figura (b) fue generado utilizando el siguiente código:

```
ggAcf(sim, lag=10)
```

Continuamos, ahora contrastando el gráfico **PACF** del proceso original versus el simulado



(a) **PACF** del proceso original



(b) **PACF** del proceso simulado

de donde

(1) Para este caso vemos nuevamente que el patrón es muy similar, donde los primeros dos valores tienen un comportamiento idéntico, teniendo en ambos gráficos valores muy próximos para los dos primeros lag's

(2) Después del $lag = 2$, el resto de los valores cae dentro del intervalo de confianza, en la figura (b), lo que quiere decir que en general dichos valores son cercanos a cero. En contraste, en la figura (a), en realidad los valores para $lag > 2$ son prácticamente cero.

El gráfico **PACF** de la figura (b) fue generado utilizando el siguiente código:

```
ggPAcf(sim, lag=10)
```

2. Considere el proceso $ARMA(1,1)$:

$$X_t = \phi X_{t-1} + \theta Z_{t-1} + Z_t$$

donde $\mid \phi \mid < 1$, y $\{Z_t\}$ es un ruido blanco Gaussiano

a) Encuentre la expresión general para la función de autocovarianza γ_k
Solución: Recordemos nuestro proceso original

$$X_t = \phi X_{t-1} + \theta Z_{t-1} + Z_t$$

Buscamos γ_k que por definición tenemos:

$$\gamma_k = Cov(X_{t+k}, X_t) = \mathbb{E}[X_{t+k}X_t]$$

Que con la definición de nuestro proceso podemos verlo como:

$$\mathbb{E}[(\phi X_{t+k-1} + \theta Z_{t+k-1} + Z_{t+k})X_t] = \mathbb{E}[\phi X_{t+k-1}X_t + \theta Z_{t+k-1}X_t + Z_{t+k}X_t]$$

Con lo que finalmente tendríamos:

$$\phi\mathbb{E}[X_{t+k-1}X_t] + \theta\mathbb{E}[Z_{t+k-1}X_t] + \mathbb{E}[Z_{t+k}X_t]$$

Comencemos por $\mathbb{E}[Z_{t+k}X_t]$ por definicion un proceso ARMA(1,1) lo podemos ver como $X_t = \sum_{j=0}^{\infty} \psi_j Z_{t-j}$, por lo que tendríamos:

$$\mathbb{E}[Z_{t+k}X_t] = \mathbb{E}[Z_{t+k}\sum_{j=0}^{\infty}\psi_j Z_{t-j}] = \sum_{j=0}^{\infty}\psi_j\mathbb{E}[Z_{t+k}Z_{t-j}]$$

Notemos que si $k = 0$ entonces será $\psi_0\sigma_Z^2$ y para cualquier otro caso será cero.

$$\mathbb{E}[Z_{t+k}X_t] = \begin{cases} \psi_0\sigma_Z^2 & si \quad k = 0 \\ \\ 0 & si \quad k \geq 1 \end{cases}$$

Ahora para $\mathbb{E}[Z_{t+k-1}X_t]$ realizamos un proceso parecido

$$\mathbb{E}[Z_{t+k-1}X_t] = \mathbb{E}[Z_{t+k-1}\sum_{j=0}^{\infty}\psi_j Z_{t-j}] = \sum_{j=0}^{\infty}\psi_j\mathbb{E}[Z_{t+k-1}Z_{t-j}]$$

Notemos que si $k = 0$ el valor sería $\psi_1\sigma_Z^2$, si $k = 1$ tenemos $\psi_0\sigma_Z^2$ y para cualquier k mayor a 1 tomaría un valor de 0, por lo tanto:

$$\mathbb{E}[Z_{t+k+1}X_t] = \begin{cases} \psi_1\sigma_Z^2 & si \quad k = 0 \\ \\ \psi_0\sigma_Z^2 & si \quad k = 1 \\ \\ 0 & si \quad k \geq 2 \end{cases}$$

Además tenemos que $\psi_0 = 1$ y $\psi_1 = \theta + \phi$, al juntar estos resultados obtenemos que:

$$\gamma_k = \begin{cases} \phi\gamma_1 + \sigma_Z^2(1 + \theta\phi + \theta^2 9 & si \quad k = 0 \\ \\ \phi\gamma_0 + \sigma_Z^2\theta & si \quad k = 1 \\ \\ \phi\gamma_{k-1} & si \quad k \geq 2 \end{cases}$$

b) Encuentre la expresión general para la función de autocorrelación ρ_k **Solución:** Notemos que para γ_k lo podemos ver como una función iterativa de la cual obtendríamos que:

$$\gamma_k = \phi^{k-1}\gamma_1$$

Para proceder es necario resolver el sistema de ecuaciones dado por:

$$\gamma_0 = \phi\gamma_1 + \sigma_Z^2(1 + \theta\phi + \theta^2) \tag{1}$$
$$\gamma_1 = \phi\gamma_0 + \sigma_Z^2 \tag{2}$$

Resolvendo el sistema obtenemos las siguientes soluciones:

$$\gamma_0 = \frac{(1 + 2\phi\theta + \theta^2)\sigma_Z^2}{1 - \phi^2} \tag{3}$$
$$\gamma_1 = \frac{(1 + \phi\theta)(\phi + \theta)\sigma_Z^2}{1 - \phi^2} \tag{4}$$

Por lo tanto la expresion para γ_k sería:

$$\gamma_k = \phi^{k-1}\frac{(1 + \phi\theta)(\phi + \theta)\sigma_Z^2}{1 - \phi^2}$$

Resolviendo $\frac{\gamma_k}{\gamma_0}$ llegamos a que:

$$\rho_k = \frac{\phi^{k-1}(1 + \phi\theta)(\phi + \theta)}{1 + 2\phi\theta + \theta^2}$$

c) Encuentre la expresión general para la funcion de autocorrelación parcial ϕ_{kk}
Solución: Comenzaremos por resolver ϕ_{11}

$$\phi_{11} = \rho_1 = \frac{(1 + \phi\theta)(\phi + \theta)}{1 - \phi^2}$$

Ahora resolvamos para ϕ_{22}

$$\phi_{22} = \frac{\rho_2 - \rho_1^2}{1 - \rho_1^2}$$

Que sustituyendo con los valores de ρ que tenemos llegamos a:

$$\phi_{22} = \frac{\phi(1 + \phi\theta)(\phi\theta)(1 + 2\phi\theta + \theta^2) - (1 + \phi\theta)^2(\phi\theta)^2}{(1 + 2\phi\theta + \theta^2)^2 - (1 + \phi\theta)^2(\phi\theta)^2}$$

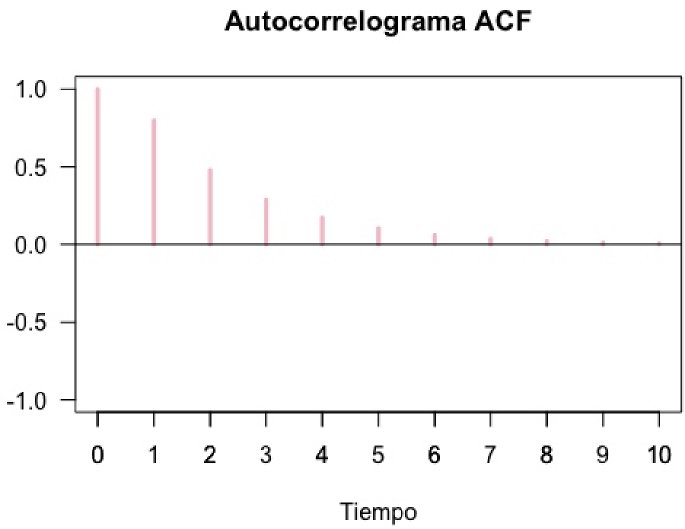
Así para encontrar ϕ_{33} habría que resolver:

$$\phi_{33} = \frac{(\rho_3 - \rho_1\rho_2) - \rho_1(\rho_1\rho_3 - \rho_2^2) + \rho_1(\rho_1^2 - \rho_2)}{(1 - \rho_1^2) - \rho_1(\rho_1 - \rho_1\rho_2) + \rho_2(\rho_1^2 - \rho_2)}$$

De esta manera deberíamos usar Crammer para encontrar la expresión en términos ρ de ϕ_{kk}

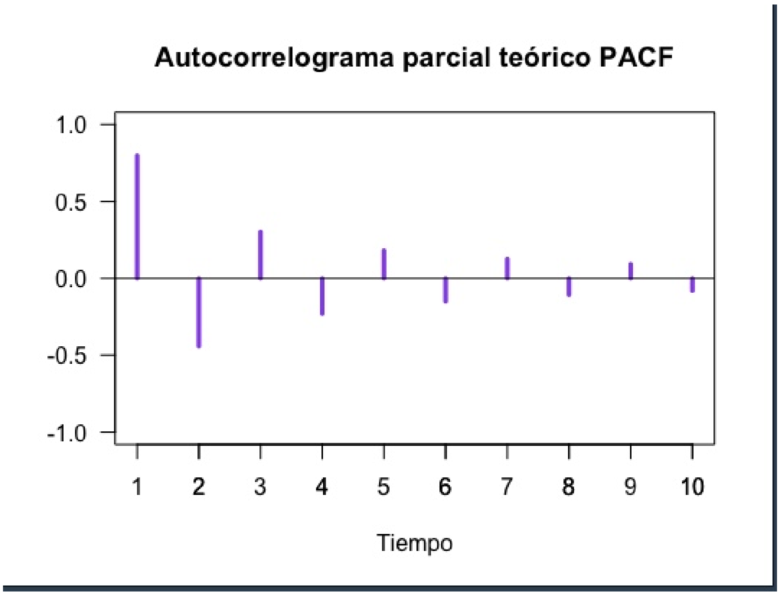
d) Usando $\phi = 0.6$ y $\theta = 0.9$, grafique ρ_k (correlograma ACF), para k = 0, 1, 2,..., 10.

```
        ACF_AR <- ARMAacf(ar = 0.6, ma = 0.9, lag.max = 10)
        plot(0:10, ACF_AR, type = "h", main = "Autocorrelograma ACF",
        ↪  ylim=c(-1,1), col="pink",las=1, ylab="",xlab="Tiempo", lwd=3)
        abline(h=0,col="black",lwd=1)
        axis(1, at = 0:10)
```



e) Usando $\phi = 0.6$ y $\theta = 0.9$, grafique ϕ_{kk} (correlograma PACF), para k = 0, 1, 2,..., 10.

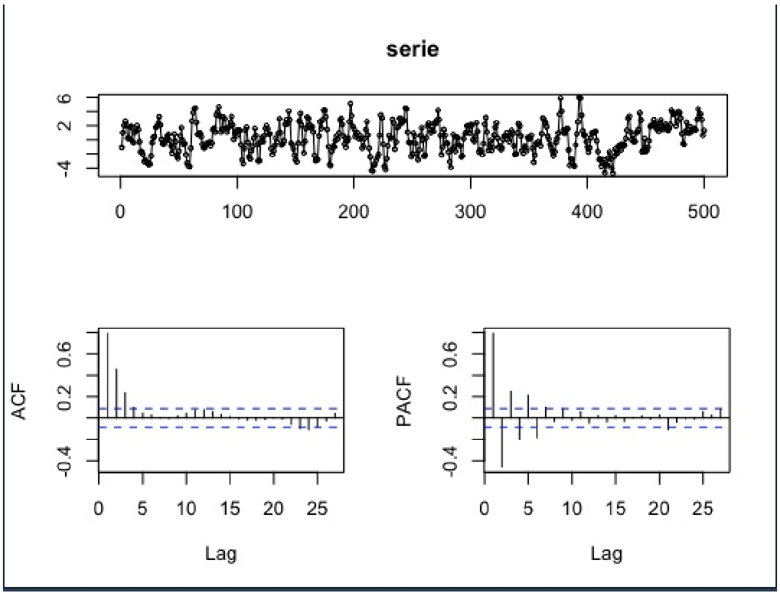
```
PACF_AR <- ARMAacf(ar=0.6, ma=0.9, lag.max = 10, pacf = T)
plot(PACF_AR, type="h",main="Autocorrelograma parcial teórico PACF", col="purple", las=1, ylab="",xlab="Tiempo", lwd=3,ylim=c(-1,1))
abline(h=0,col="black",lwd=1)
axis(1, at = 1:10)
```



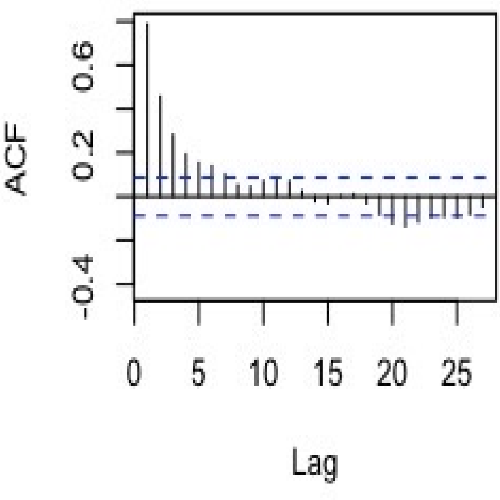
f) En R simule el proceso $\{X_t\}$ para un tamaño de muestra n, grafique la serie de tiempo y los correlogramas ACF y PACF.

```
n<-500
phi<-0.6
theta<-0.9
```

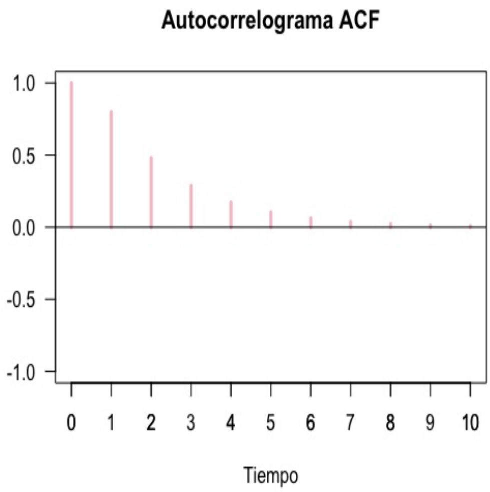
```
serie<-arima.sim(list(order=c(1,0,1),ar=phi,ma=theta),n=n)
tsdisplay(serie)
```



Compare los correlogramas simulados con el proceso original.
Comparemos los correlogramas.



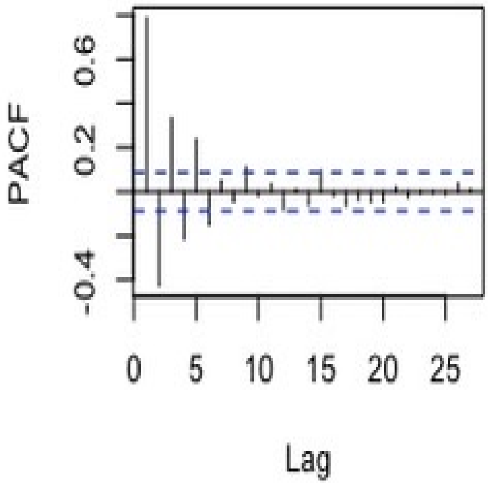
(a) **ACF** del proceso original



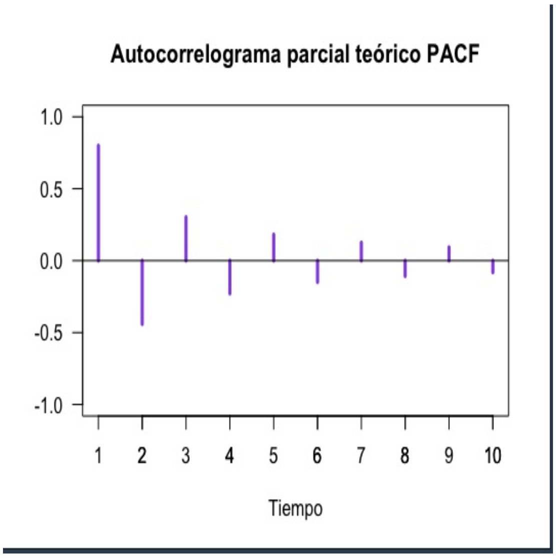
(b) **ACF** del proceso simulado

(1) Podemos notar que el patrón es bastante similar, observemos que los primeros siguen un patrón bastante parecido, tomando en cuenta que va decreciendo, notemos también que en la gráfica (a) no se muestra como primer valor el correspondiente a $0 = 1$; y podemos ver que si es incluido en la grafica en la figura (a).

Ahora para los PACF:



(a) **PACF** del proceso original



(b) **ACF** del proceso simulado

(1) Podemos notar que el patrón es bastante similar, observemos que los valores crecen y decrecen, alternandose uno y uno, entre más va creciendo el lag, los valores de PACF se van haciendo cada vez más pequeños.

3. Considere el siguiente proceso:

$$X_t = X_{t-1} - 0.5X_{t-2} + Z_t - Z_{t-1}$$

a) Escriba el proceso X_t en su forma de polinomio de retraso, donde Z_t es un ruido blanco.
Solución: La expresión principal la podmeos reescribir como:

$$X_t - X_{t-1} + 0.5X_{t-2} = Z_t - Z_{t-1}$$

$$(1 - \mathbf{B} + 0.5\mathbf{B}^2)X_t = (1 - \mathbf{B})Z_t$$

$$X_t = \frac{(1 - \mathbf{B})Z_t}{(1 - \mathbf{B} + 0.5\mathbf{B}^2)}$$

Por lo que el polinomio de retraso para X_t es:

$$\frac{(1 - \mathbf{B})}{(1 - \mathbf{B} + 0.5\mathbf{B}^2)}$$

b) Clasifique el modelo como un AR(p), MA(q) o ARMA(p, q) y defina el orden.
Solución: De la ecuación principal, la podemos reescribir como:

$$X_t - X_{t-1} + 0.5X_{t-2} = Z_t - Z_{t-1}$$

$$(1 - \mathbf{B} + 0.5\mathbf{B}^2)X_t = (1 - \mathbf{B})Z_t$$

Por definición, podemos clasificar las dos ecuaciones de la siguiente manera:

$$\phi(\mathbf{B}) = 1 - \mathbf{B} + 0.5\mathbf{B}^2$$

$$\theta(\mathbf{B}) = 1 - \mathbf{B}$$

Podemos notar que $\phi(\mathbf{B})$ es de grado 2 y $\theta(\mathbf{B})$ es de grado 1
Por lo tanto, al proceso original lo podemos clasificar como un modelo *ARMA(2,1)*
ya que la ecuacion orginal la podemos reescribir como:

$$\phi_2(\mathbf{B})X_t = \theta_1(\mathbf{B})Z_t$$

c) Determina si el proceso es estacionario, invertible y causal.
Solución:

I. Invertible. Por preposión, el modelo ARMA será invertible si las soluciones a la ecuacion $\theta_1(\mathbf{B}) = 0$ son mayores a 1.
Notemos que:

$$\theta(\mathbf{B}) = 1 - \mathbf{B} = 0 \Leftrightarrow \mathbf{B} = 1$$

Su raiz no es mayor a 1, por lo tanto no cumple la invertibilidad.

II. Causal. Por preposión, el modelo ARMA será causal si las soluciones a la ecuacion $\phi_2(\mathbf{B}) = 0$ son mayores a 1.
Notemos que:

$$\phi(\mathbf{B}) = 1 - \mathbf{B} + 0.5\mathbf{B}^2 = 0 \Leftrightarrow \mathbf{B} = 1 + 1, \mathbf{B} = 1 - i$$

Sus raices son mayor a 1, por lo tanto el proceso es causal.

III. Estacionario. El modelo ARMA es estacionario Por defecto, los modelos MA siempre serán estacionarios, faltará demostrar que la parte AR es estacionaria.

Un modelo AR es estacionario si todas sus raices son mayores al modulo unitario.
Recordemos que el polinomio asociado al modelo AR es:

$$\phi(\mathbf{B}) = 1 - \mathbf{B} + 0.5\mathbf{B}^2$$

Donde sus raices las obtuvimos en el punto anterior, las cuales son mayores al modulo unitario.
Por lo tanto, la parte AR es estacionaria.
Por lo tanto, el modelo ARMA(2,1) es estacionario.

d) Obtenga la representacion de un AR(∞) y un MA(∞) respectivamente, si es que existe **Solución:**

1. Como no existe la invertibilidad, no existe $AR(\infty)$
2. Como es causal, existe una expresion para $MA(\infty)$ donde:

$$X_t = \frac{\theta(\mathbf{B})}{\phi(\mathbf{B})} \cdot Z_t$$

$$\Leftrightarrow X_t = \frac{1 - \mathbf{B}}{1 - \mathbf{B} + 0.5\mathbf{B}^2} \cdot Z_t$$

$$\Leftrightarrow X_t = \frac{1 - \mathbf{B}}{(\mathbf{B} - 1 - i)(\mathbf{B} - 1 + i)} \cdot Z_t$$

Finalmente, utilizando el software *wolfram alpha* podemos ver que dicho polinomio racional tiene una expansión en serie. De tal manera tenemos que

$$X_t = \left(\sum_{n=0}^{\infty} \left[\left(\frac{1}{2} - \frac{i}{2} \right)^{1+n} + \left(\frac{1}{2} + \frac{i}{2} \right)^{1+n} \right] \mathbf{B}^n \right) Z_t = \sum_{n=0}^{\infty} \left[\left(\frac{1}{2} - \frac{i}{2} \right)^{1+n} + \left(\frac{1}{2} + \frac{i}{2} \right)^{1+n} \right] Z_{t-n}$$