

Índice

1. Funciones de autocovarianzas y de autocorrelación (ACF)	6
2. Función de autocorrelación parcial (PACF)	6
3. Densidad espectral	7
4. ACF, PACF y densidad espectral de algunos modelos lineales	8
4.1. Procesos lineales causales	8
4.2. Proceso de Ruido Blanco	9
4.3. Proceso de media móvil $MA(q)$	10
4.4. Proceso de media móvil $MA(1)$	11
4.4.1. $MA(1)$ con raíz positiva	11
4.4.2. $MA(1)$ con raíz negativa	12
4.5. Proceso de media móvil $MA(2)$	12
4.5.1. $MA(2)$ con dos parámetros positivos $(+, +)$	13
4.5.2. $MA(2)$ con parámetros de distinto signo $(-, +)$	14
4.5.3. $MA(2)$ con dos parámetros negativos $(-, -)$	15
4.5.4. $MA(2)$ con parámetros de distinto signo $(+, -)$	16
4.5.5. $MA(2)$ con dos raíces reales, una positiva y la otra negativa	16
4.5.6. $MA(2)$ con un par de raíces complejas	17
4.5.7. $MA(2)$ con dos raíces reales y negativas	18
4.5.8. $MA(2)$ con dos raíces reales y positivas	19
5. (Resumen) características de procesos $MA(q)$	19
5.1. Gráfico raíces $MA(2)$	20

Lección 4. ACF, PACF y densidad espectral de modelos MA

Marcos Bujosa

20 de agosto de 2025

Resumen

Veremos tres funciones que describen las características correspondientes a los momentos de segundo orden de los procesos lineales: la función de autocorrelación (ACF), la función de autocorrelación parcial (PACF) y la densidad espectral (estas tres funciones son transformaciones de la función de autocovarianzas del proceso). Veremos la estructura de estas funciones en procesos lineales genéricos y hoy lo particularizaremos a procesos de ruido blanco y procesos $MA(q)$.

- ([slides](#)) — ([html](#)) — ([pdf](#)) — ([mybinder](#))

Carga de algunos módulos de python y creación de directorios auxiliares

```
# Para trabajar con los datos y dibujarlos necesitamos cargar algunos módulos de python
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib as mpl
# definimos parámetros para mejorar los gráficos
mpl.rc('text', usetex=False)
import matplotlib.pyplot as plt # data visualization
```

- Directorio auxiliar para albergar las figuras de la lección: para publicar la lección como pdf o página web, necesito los gráficos como ficheros `.png` alojados algún directorio específico:

```
imagenes_leccion = "./img/lecc04" # directorio para las imágenes de la lección
import os
os.makedirs(imagenes_leccion, exist_ok=True) # crea el directorio si no existe
```

Gráficos para las ACF, PACF y densidades espectrales teóricas

```
# Módulos necesarios
import statsmodels.api as sm
from statsmodels.tsa.arima_process import ArmaProcess
from scipy import signal
from scipy.signal import freqz
from numpy.polynomial import Polynomial
import math
```

La siguiente función que calcula la PACF a partir de la ACF resolviendo las [ecuaciones de Yule-Walker](#) recursivamente con el algoritmo [Levinson-Durbin](#); véase *Pourahmadi, M. (2001, Capítulo 7)* o *Brockwell & Davis (1991, Capítulo 5)*).

```

def theoretical_pacf_from_acf(acf_vals):
    """
    Calcula la PACF teórica (para lags >= 0) de forma paramétrica usando el algoritmo de Durbin-Levinson.
    Se asume que acf_vals es un array tal que acf_vals[0] corresponde a lag 0.

    Retorna un array donde el coeficiente en la posición k corresponde a la PACF en el lag k.
    """
    nlags = len(acf_vals)-1
    # Matriz para almacenar los coeficientes (phi)
    phi = np.zeros((nlags+1, nlags+1))
    # v guarda la varianza de error en cada paso
    v = np.zeros(nlags+1)
    pacf = np.zeros(nlags+1)
    v[0] = acf_vals[0]
    pacf[0] = 1.0 # Por convención, PACF(0)=1.

    for k in range(1, nlags+1):
        if k == 1:
            phi[1,1] = acf_vals[1] / acf_vals[0]
            pacf[1] = phi[1,1]
            v[1] = v[0]*(1 - phi[1,1]**2)
        else:
            sum_val = 0.0
            # Sumar phi[k-1,j]*acf_vals[k-j] para j=1,...,k-1.
            for j in range(1, k):
                sum_val += phi[k-1, j] * acf_vals[k - j]
            phi[k, k] = (acf_vals[k] - sum_val) / v[k-1]
            pacf[k] = phi[k, k]
            for j in range(1, k):
                phi[k, j] = phi[k-1, j] - phi[k, k] * phi[k-1, k - j]
            v[k] = v[k-1] * (1 - phi[k, k]**2)
    return pacf # Incluye PACF en lag 0 con valor 1.

```

La siguiente función pinta las ACF, PACF y densidad espectral teóricas.

```

def plot_arma_parametric_diagnostics(ar_params=[1,], ma_params=[1,], sigma2=1, lags=20):
    """
    Genera y retorna una figura con tres subgráficas para un modelo ARMA:
    - ACF teórica (omitiendo retardo 0)
    - PACF teórica paramétrica (calculada vía el algoritmo de Durbin-Levinson, omitiendo retardo 0)
    - Densidad espectral teórica

    Parámetros:
    ar_params : array-like
        Coeficientes del polinomio AR (incluye el 1 inicial).
    ma_params : array-like
        Coeficientes del polinomio MA (incluye el 1 inicial).
    sigma2 : float
        Varianza del ruido U.
    lags : int, opcional
        Número de retardos considerados (por defecto 20).

    Retorna:
    fig : objeto matplotlib.figure.Figure
    """
    phi = Polynomial(ar_params, symbol='B')
    theta = Polynomial(ma_params, symbol='B')
    # Crear el proceso ARMA
    arma_process = ArmaProcess(ar=phi.coef, ma=theta.coef)

    # 1. Calcular la ACF teórica
    acf_theo = arma_process.acf(lags=lags)
    lags_theo = np.arange(len(acf_theo))

```

```

# Omitir el retardo 0 para gráficos de ACF y PACF:
acf_plot = acf_theo[1:]
lags_plot = lags_theo[1:]

# 2. Calcular la PACF teórica de forma paramétrica a partir de la ACF
pacf_theo = theoretical_pacf_from_acf(acf_theo)
pacf_plot = pacf_theo[1:]
lags_pacf = np.arange(len(pacf_theo))[1:]

# 3. Calcular la densidad espectral teórica usando freqz
w, h = freqz(theta.coef, phi.coef, worN=1024)
freq = w / (2 * np.pi) # Conversión de radianes/muestra a ciclos/muestra
spectrum = (sigma2 / (2 * np.pi)) * np.abs(h)**2

# 4. Graficar los tres subgráficos
fig, axs = plt.subplots(1, 3, figsize=(18, 3))

# ACF teórica
axs[0].stem(lags_plot, acf_plot, basefmt=" ")
axs[0].axhline(0, color='black', lw=0.5, linestyle='--')
axs[0].set_xlabel('Retardo')
axs[0].set_ylabel('ACF')

# PACF teórica paramétrica
axs[1].stem(lags_pacf, pacf_plot, basefmt=" ")
axs[1].axhline(0, color='black', lw=0.5, linestyle='--')
axs[1].set_xlabel('Retardo')
axs[1].set_ylabel('PACF')

# Densidad espectral teórica
axs[2].plot(freq, spectrum)
axs[2].axhline(0, color='black', lw=0.5, linestyle='--')
axs[2].set_xlabel('Frecuencia')
axs[2].set_ylabel('Densidad Espectral')

plt.tight_layout()
plt.close(fig)
return fig

```

Breve explicación de la anterior función `plot_arma_parametric_diagnostics`:

1. La función recibe los parámetros del modelo ARMA y el número de retardos.
2. Se calcula la ACF teórica y a partir de ella la PACF con `theoretical_pacf_from_acf`; en la gráfica se omite el retardo cero.
3. La densidad espectral teórica $S(\omega)$ se obtiene mediante la función `freqz` y la fórmula

$$S(\omega) = \frac{\sigma^2}{2\pi} \left| \frac{\sum_{j=0}^q \theta_j e^{-i2\pi\omega j}}{\sum_{k=0}^p \phi_k e^{-i2\pi\omega k}} \right|^2.$$

Donde:

- σ^2 es la varianza del término de error (innovación) del modelo.
- $(\theta_j \mid j = 0 : q)$ son los coeficientes del polinomio de medias móviles (MA), de modo que el polinomio MA es: $\theta(z) = \sum_{j=0}^q \theta_j z^j$.
- $(\phi_k \mid k = 0 : p)$ son los coeficientes del polinomio autorregresivo (AR), de modo que el polinomio AR es: $\phi(z) = \sum_{k=0}^p \phi_k z^k$.

- Al evaluar estos polinomios en $z = e^{-i2\pi\omega}$ se obtiene la respuesta en frecuencia del modelo:

$$H(e^{-i2\pi\omega}) = \frac{\theta(e^{-i2\pi\omega})}{\phi(e^{-i2\pi\omega})}.$$

- El módulo al cuadrado de $H(e^{-i2\pi\omega})$ multiplicado por $\frac{\sigma^2}{2\pi}$ es la densidad espectral.

Esta fórmula es fundamental porque permite conocer la distribución de la varianza del proceso a lo largo de las diferentes frecuencias, lo cual es clave para analizar las propiedades dinámicas en modelos ARMA.

4. Se crean tres subgráficos, y se añade una línea horizontal en $y=0$ en cada uno (`axhline`).
5. Por último, la función retorna el objeto figura.

Función que simula un modelo ARIMA y pinta la serie temporal, las ACF, PACF y densidad espectral estimadas con los datos simulados.

```
def plot_arma_analysis(ar_params=[1,], ma_params=[1,], sigma2=1, lags=20, n=400, seed=0):

    lags=math.floor(n/2) if n/2 < lags else lags

    phi = Polynomial(ar_params, symbol='B')
    theta = Polynomial(ma_params, symbol='B')
    arma_process = sm.tsa.ArmaProcess(phi.coef, theta.coef)

    # Simulación de la serie temporal
    if seed:
        np.random.seed(seed)
    data = arma_process.generate_sample(nsample=n)

    # Crear la figura
    fig = plt.figure(figsize=(18, 7))

    # Subgráfico de la serie temporal
    ax1 = fig.add_subplot(2, 1, 1)
    ax1.plot(data, color='blue')
    ax1.set_xlabel('t')

    # Crear una fila de 3 subgráficos para ACF, PACF y periodograma
    ax2 = fig.add_subplot(2, 3, 4) # Fila 2, Columna 1
    sm.graphics.tsa.plot_acf(data, lags=lags, ax=ax2, title='ACF Estimada', zero=False, auto_ylims=True)
    ax2.set_xlabel('Retardo')

    ax3 = fig.add_subplot(2, 3, 5) # Fila 2, Columna 2
    sm.graphics.tsa.plot_pacf(data, lags=lags, ax=ax3, title='PACF Estimada', zero=False, auto_ylims=True)
    ax3.set_xlabel('Retardo')

    ax4 = fig.add_subplot(2, 3, 6) # Fila 2, Columna 3
    f, Pxx = signal.welch(data, fs=1, nperseg=256)
    ax4.plot(f, Pxx, color='purple')
    ax4.set_title('Periodograma Estimado')
    ax4.set_xlabel('Frecuencia')
    #ax4.set_ylabel('Densidad Espectral')
    ax4.set_xlim(0, 0.5) # Limitar el eje x a 0.5

    plt.tight_layout()
    plt.close(fig)
    return fig
```

1. Funciones de autocovarianzas y de autocorrelación (ACF)

Si un proceso estocástico es estacionario, la covarianza entre X_t y X_{t+k} no depende de t ; tan solo depende de la distancia temporal k entre ambas variables aleatorias: $Cov(X_t, X_{t-k}) = \gamma_k$.

- La secuencia $\gamma = (\gamma_k \mid k \in \mathbb{Z})$ se denomina *función de autocovarianzas*
- Dividiendo γ por γ_0 obtenemos la secuencia $\rho = (\rho_k \mid k \in \mathbb{Z})$ donde

$$\rho_k = \frac{Cov(X_t, X_{t-k})}{\sqrt{Var(X_t)Var(X_{t-k})}} = \frac{\gamma_k}{\gamma_0},$$

que se denomina *función de autocorrelación* (ACF).

Veamos otras transformaciones de γ que subrayan diferentes características de la interdependencia temporal en un proceso estocástico.

2. Función de autocorrelación parcial (PACF)

Dado un conjunto de variables $\mathbf{Z} = (Z_1, Z_2, \dots, Z_k)$, la *correlación condicional* $\rho_{X,Y|\mathbf{Z}}$ entre las variables X e Y es

$$\rho_{X,Y|\mathbf{Z}} = Corr(X - E(X \mid \mathbf{Z}), Y - E(Y \mid \mathbf{Z}))$$

El problema es que no siempre es fácil conocer las esperanzas condicionadas $E(X \mid \mathbf{Z})$ y $E(Y \mid \mathbf{Z})$.

Pero es sencillo restar de X y de Y los ajustes $\hat{X}$ y $\hat{Y}$ correspondientes a sus respectivas regresiones lineales sobre $\mathbf{Z}$. La correlación entre ambos residuos se denomina *correlación parcial* entre X e Y ; y mide la correlación entre ambas variables una vez “descontado” el efecto lineal de las variables $\mathbf{Z}$.

$$Corr((X - \hat{X}), (Y - \hat{Y}))$$

La *correlación parcial* entre dos variables X_t y X_{t-k} de un proceso estacionario $\mathbf{X}$ mide su correlación una vez descontado el efecto de las variables $X_{t-1}, \dots, X_{t-(k-1)}$ que median entre ambas.

$$X_t, \overbrace{X_{t-1}, \dots, X_{t-(k-1)}}, X_{t-k}$$

Si denotamos con $\hat{X}_t$ y $\widetilde{X_{t-k}}$ los ajustes de las respectivas regresiones de X_t y X_{t-k} sobre $X_{t-1}, \dots, X_{t-(k-1)}$; la *correlación parcial* π_k entre X_t y X_{t+k} es la correlación entre los residuos de sendas regresiones:

$$\pi_k = Corr((X_t - \hat{X}_t), (X_{t-k} - \widetilde{X_{t-k}}))$$

En un proceso débilmente estacionario las correlaciones parciales solo dependen de la distancia k , lo que permite definir la siguiente secuencia.

La función de autocorrelación parcial (PACF) de un proceso estocástico estacionario $\mathbf{X}$ es la siguiente secuencia π simétrica ($\pi_{-k} = \pi_k$):

$$\boldsymbol{\pi} = (\pi_k \mid k \in \mathbb{Z}) = \begin{cases} \pi_0 &= \rho_0 = 1 \\ \pi_1 &= \text{Corr}(X_t, X_{t-1}) = \rho_1 = \frac{\gamma_1}{\gamma_0} \\ \pi_k &= \text{Corr}((X_t - \widehat{X}_t), (X_{t-k} - \widetilde{X}_{t-k})) \end{cases},$$

donde $\widehat{X}_t$ y $\widetilde{X}_{t-k}$ son los ajustes de las respectivas regresiones de X_t y X_{t-k} sobre $X_{t-1}, \dots, X_{t-(k-1)}$; i.e., las proyecciones ortogonales sobre $\bar{sp}(1, X_{t-1}, \dots, X_{t-(k-1)})$.

La magnitud de la correlación parcial π_k **refleja la mejora en la predicción de $\widehat{X}_t$ si** en lugar de usar una combinación lineal con solo los $k-1$ primeros retardos, $\widehat{X}_t = \widehat{\alpha}_0 + \sum_{j=1}^{k-1} \widehat{\alpha}_j X_{t-j}$, **empleamos k retardos** (i.e., un retardo más).

Así, la PACF nos ayudará a identificar el orden de procesos autoregresivos.

Hay una correspondencia uno-a-uno entre la función de autocovarianzas γ y la PACF $\boldsymbol{\pi}$.

Es decir, es posible reconstruir una de las secuencias a partir de la otra (por ejemplo resolviendo las [ecuaciones de Yule-Walker](#) recursivamente con el algoritmo [Levinson-Durbin](#); véase *Pourahmadi, M. (2001, Capítulo 7)* o *Brockwell & Davis (1991, Capítulo 5)*)

Así, la PACF $\boldsymbol{\pi}$ puede verse como una reparametrización de la función de autocovarianzas γ .

Veamos otra transformación de γ que también arrojará luz sobre las propiedades de un proceso estocástico.

3. Densidad espectral

Si $\mathbf{X}$ es un proceso estocástico con función de autocovarianzas $\gamma \in \ell^1$, es decir, con función de autocovarianzas absolutamente sumable,

$$\sum_{k \in \mathbb{Z}} |\gamma_k| < \infty,$$

entonces definimos la *densidad espectral* de $\mathbf{X}$ como

$$f(\omega) = \frac{1}{2\pi} \sum_{k \in \mathbb{Z}} \gamma_k e^{-ik\omega} = \frac{1}{2\pi} \sum_{k \geq 0} \gamma_k \cos(k\omega), \quad \omega \in [-\pi, \pi],$$

(donde ω son las frecuencias de oscilación de la serie).

La *densidad espectral* $f(\omega)$ satisface las siguientes propiedades:

1. $f(\omega) = f(-\omega)$
2. $f(\omega) \geq 0$
3. $\int_{-\pi}^{\pi} f(\omega) d\omega < \infty$

Es más, partiendo de la densidad espectral se pueden calcular las covarianzas

$$\gamma_k = \int_{-\pi}^{\pi} f(\omega) e^{-ik\omega} d\omega; \quad k \in \mathbb{Z}$$

Consecuentemente, para $k = 0$

$$\sigma^2 = \gamma_0 = \int_{-\pi}^{\pi} f(\omega) d\omega.$$

Por tanto podemos interpretar la densidad espectral como una descomposición de la varianza en sumas de oscilaciones con distintas frecuencias (o periodos).

($Periodo = \frac{1}{Frecuencia}$, donde la Frecuencia = N.º de ciclos por unidad de tiempo).

La ACF, la PACF y la *densidad espectral* son funciones que dependen únicamente de los dos primeros momentos de la distribución.

Su estimación y posterior análisis son la herramienta fundamental para elegir un modelo ARMA para una serie temporal “*estacionaria*”.

Para entenderlo debemos ver cómo son estas funciones en algunos modelos lineales concretos.

4. ACF, PACF y densidad espectral de algunos modelos lineales

4.1. Procesos lineales causales

Sea $\mathbf{X} = \psi * \mathbf{U}$, donde $\mathbf{U} \sim WN(0, \sigma^2)$ y $\psi \in \ell^2$ es una serie formal:

$$X_t = \sum_{j \geq 0} \psi_j U_{t-j}.$$

Por la [lección anterior](#) sabemos que $E(X_t) = 0$ y que

$$\gamma = \sigma^2 \psi(z) * \psi(z^{-1})$$

Es decir,

$$\gamma_k = \sigma^2 \sum_{j=0}^{\infty} \psi_{j+|k|} \cdot \psi_j.$$

Por otra parte, $\gamma_k \rightarrow 0$ cuando $k \rightarrow \infty$

Demostración: aplicando la desigualdad de Cauchy-Schwarz

$$|\gamma_k| \leq \sigma^2 \left(\sum_{j=0}^{\infty} |\psi_{j+k}|^2 \right)^{\frac{1}{2}} \left(\sum_{j=0}^{\infty} |\psi_j|^2 \right)^{\frac{1}{2}} = \sigma^2 \left(\sum_{j=k}^{\infty} |\psi_j|^2 \right)^{\frac{1}{2}} \left(\sum_{j=0}^{\infty} |\psi_j|^2 \right)^{\frac{1}{2}}$$

y puesto que ψ es de cuadrado sumable, la cola $\sum_{j=k}^{\infty} |\psi_j|^2$ converge a cero cuando $k \rightarrow \infty$.

A partir de γ obtenemos las otras tres funciones:

ACF: $\rho = \frac{1}{\gamma_0}\gamma$

Dens. espectral: $f(\omega) = \frac{1}{2\pi} \sum_{k \geq 0} \gamma_k \cos(k\omega); \quad \omega \in [-\pi, \pi]$

PACF: $\pi \in \ell^2$, i.e., $\pi_k \rightarrow 0$ cuando $k \rightarrow \infty$

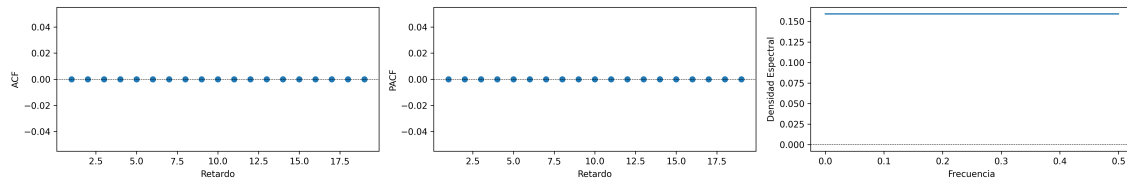
(Cuando $\pi \in \ell^2$ se dice que el proceso es *no-determinista*.)

4.2. Proceso de Ruido Blanco

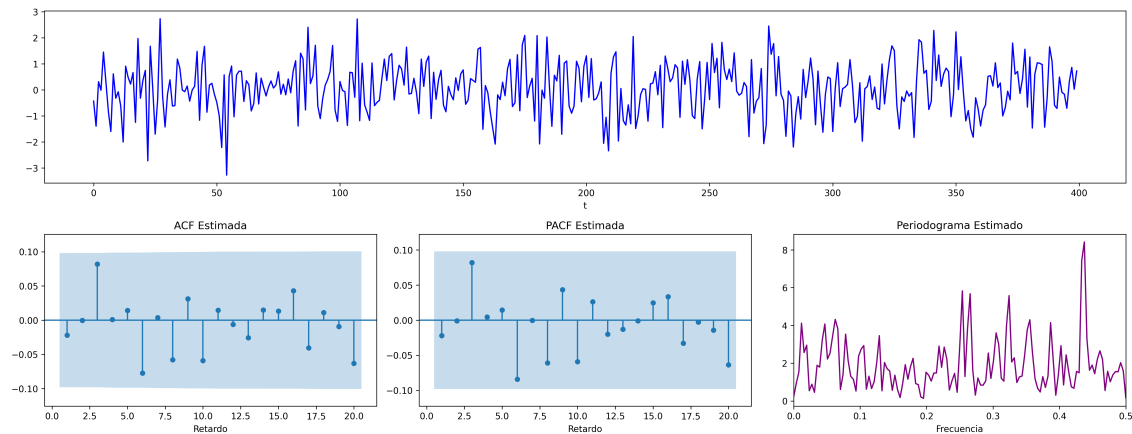
Sea $\mathbf{X} = 1 * \mathbf{U}$, donde $\mathbf{U} \sim WN(0, \sigma^2)$; es decir $X_t = U_t$ para $t \in \mathbb{Z}$. Entonces

- $\gamma = \sigma^2(1z^0 * 1z^0) = \sigma^2 z^0 = (\dots, 0, 0, \sigma^2, 0, 0, \dots)$
- $\rho = \frac{1}{\gamma_0}\gamma = \frac{1}{\sigma^2}\gamma = 1z^0 = (\dots, 0, 0, 1, 0, 0, \dots)$
- $f(\omega) = \frac{1}{2\pi} \sum_{k \geq 0} \gamma_k \cos(k\omega) = \frac{\sigma^2}{2\pi} \cos(0\omega) = \frac{\sigma^2}{2\pi}; \quad \omega \in [-\pi, \pi]$
- $\pi = 1z^0 = (\dots, 0, 0, 1, 0, 0, \dots)$

```
ar_params = [1,]
ma_params = [1,]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, 1, lags=20)
fig.savefig('./img/lecc04/ACFwhitenoise.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params, seed=2026)
fig.savefig('./img/lecc04/SimulWhitenoise.png', dpi=300, bbox_inches='tight')
```



4.3. Proceso de media móvil MA(q)

Sea $\mathbf{X} = \boldsymbol{\theta} * \mathbf{U}$, con $\mathbf{U} \sim WN(0, \sigma^2)$ y donde $\boldsymbol{\theta}$ es un polinomio de orden $q > 0$, con $\theta_0 = 1$ y con todas sus raíces fuera del círculo unidad ($\boldsymbol{\theta}^{-\triangleright} \in \ell^1$), es decir

$$X_t = \sum_{j=1}^q \theta_j U_{t-j} + U_t.$$

Este requisito no es necesario para que el proceso sea estacionario, pero por conveniencia se usan modelos MA que puedan tener una representación AR(∞) causal (es decir, usando observaciones del pasado “para construir” el presente):

$$\mathbf{X} = \boldsymbol{\theta} * \mathbf{U} \quad \rightarrow \quad \frac{1}{\boldsymbol{\theta}} * \mathbf{X} = \mathbf{U} \quad \Rightarrow \quad U_t = \sum_{j=0}^{\infty} \psi_j X_{t-j}$$

donde $\boldsymbol{\psi} = \boldsymbol{\theta}^{-\triangleright} = \boldsymbol{\theta}^{-1} \in \ell^1$. Entonces se dice que el modelo MA es “invertible”. Cualquier modelo MA sin raíces en el círculo unidad se puede reparametrizar para obtener un modelo invertible con la misma función de autocovarianzas “ $\boldsymbol{\gamma}$ ” (Véase Sección 2.6 de Fuller (1996)) y, consecuentemente, las mismas ACF, PACF y densidad espectral que el modelo original. Para tener una asociación uno a uno entre ACFs y procesos estocásticos, siempre asumiremos que las raíces de los polinomios MA, $\boldsymbol{\theta}$, son mayores que uno en valor absoluto.

Entonces $E(X_t) = 0$ y

$$\boldsymbol{\gamma} = \sigma^2 \boldsymbol{\theta}(z) * \boldsymbol{\theta}(z^{-1}) = \sigma^2 \sum_{j=0}^q \theta_{j+k} \theta_j z^k \text{ (grado } q; \text{ cogrado } -q) \text{ (Lecc. 02)}$$

$$\boldsymbol{\rho} = \frac{1}{\gamma_0} \boldsymbol{\gamma}$$

$$f(\omega) = \frac{\sigma^2}{2\pi} \boldsymbol{\theta}(e^{-i\omega}) \cdot \boldsymbol{\theta}(e^{i\omega}) = \frac{1}{2\pi} \sum_{h=0}^q \gamma_h \cos(h\omega); \quad \omega \in [-\pi, \pi] \text{ (} q \text{ cosenos } +1)$$

π con grado ∞ y cogrado $-\infty$. Su magnitud decae paulatinamente cuando $k \rightarrow \infty$

Un proceso MA es siempre estacionario.

Tipos de representación del proceso

Como suma ponderada finita (número finito de parámetros)

$$\mathbf{X} = \boldsymbol{\theta} * \mathbf{U} \quad \Rightarrow \quad X_t = \boldsymbol{\theta}(\mathbf{B}) U_t$$

Como suma ponderada infinita (solo existe si es invertible):

$$\frac{1}{\boldsymbol{\theta}} * \mathbf{X} = \mathbf{U} \quad \Rightarrow \quad \frac{1}{\boldsymbol{\theta}}(\mathbf{B}) X_t = U_t$$

Para ser invertible raíces $\boldsymbol{\theta}$ fuera del círculo unidad ($\boldsymbol{\theta}^{-\triangleright} = \boldsymbol{\theta}^{-1} \in \ell^1$)

4.4. Proceso de media móvil MA(1)

Sea $\mathbf{X} = (1 - \theta z) * \mathbf{U}$ con $|\theta| < 1$ y con $\mathbf{U} \sim WN(0, \sigma^2)$, es decir

$$X_t = U_t - \theta U_{t-1}.$$

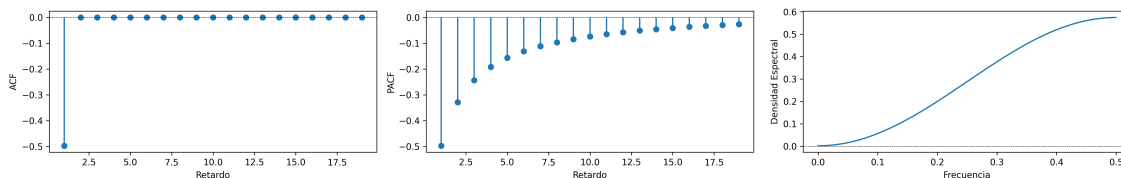
- $\gamma = \sigma^2(1 - \theta z) * (1 - \theta z^{-1}) = \sigma^2(\dots, 0, -\theta, (1 + \theta^2), -\theta, 0, \dots)$
- $\rho = \frac{1}{\gamma_0}\gamma = \frac{1}{\sigma^2(1+\theta^2)}\gamma = (\dots, 0, \frac{-\theta}{1+\theta^2}, 1, \frac{-\theta}{1+\theta^2}, 0, \dots)$
- $f(\omega) = \frac{\sigma^2}{2\pi}\theta(e^{-i\omega}) \cdot \theta(e^{i\omega}) = \frac{\sigma^2}{2\pi}((1 + \theta^2) - 2\theta \cos(\omega)); \quad \omega \in [-\pi, \pi]$
- $\pi = \sum_{k \in \mathbb{Z}} \frac{-\theta^k(1-\theta^2)}{1-\theta^{2(k+1)}} z^k$

4.4.1. MA(1) con raíz positiva

$$\theta(z) = 1 - 0,9z \Rightarrow X_t = (1 - 0,9B)U_t \quad (\theta > 0)$$

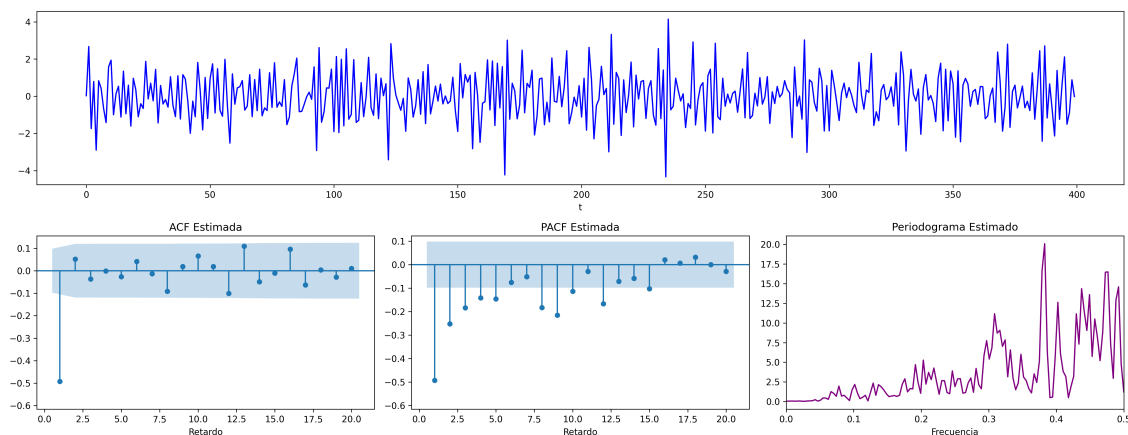
P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
1.111111	0.0	1.111111	0.0	Inf	1

```
ar_params = [1,]
ma_params = [1, -0.9]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA1p.png', dpi=300, bbox_inches='tight')
```



Ejemplo de uso

```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA1p.png', dpi=300, bbox_inches='tight')
```

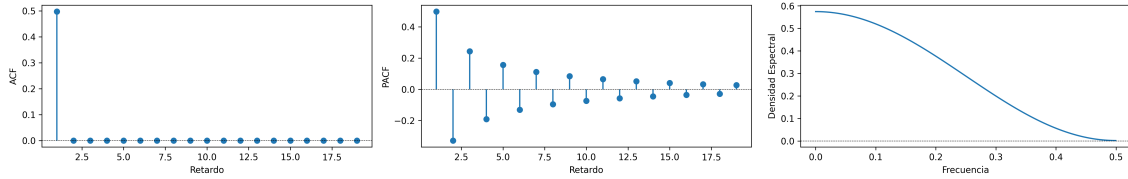


4.4.2. MA(1) con raíz negativa

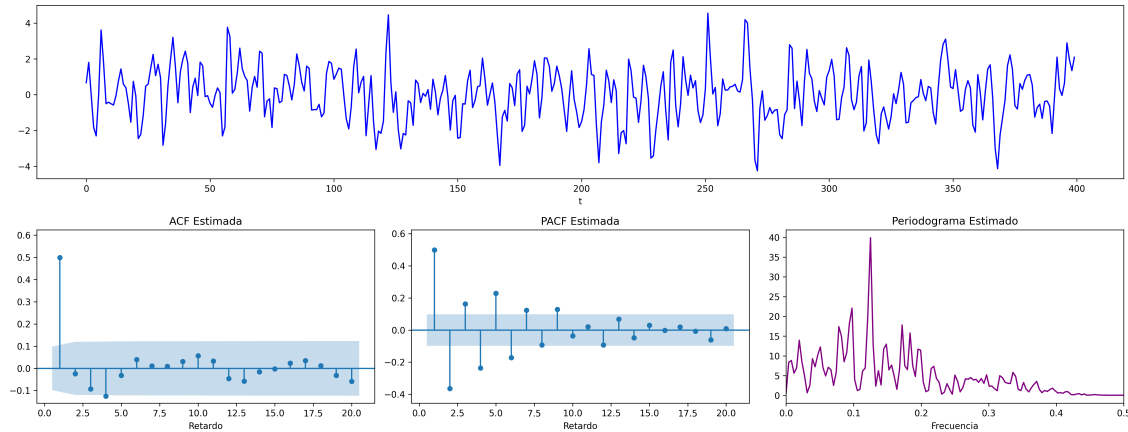
$$\theta(z) = 1 + 0,9z \Rightarrow X_t = (1 + 0,9B)U_t \quad (\theta < 0)$$

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
-1.111111	0.0	1.111111	0.5	2.0	1.0

```
ar_params = [1,]
ma_params = [1, 0.9]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA1n.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA1n.png', dpi=300, bbox_inches='tight')
```



4.5. Proceso de media móvil MA(2)

Sea $\mathbf{X} = (1 - \theta_1 z - \theta_2 z^2) * \mathbf{U}$, con $\mathbf{U} \sim WN(0, \sigma^2)$ y con

$$\theta_1 + \theta_2 < 1; \quad \theta_2 - \theta_1 < 1; \quad |\theta_2| < 1; \quad (\text{i.e., invertible})$$

es decir $X_t = U_t - \theta_1 U_{t-1} - \theta_2 U_{t-2}$.

- $\gamma = \sigma^2(1 - \theta_1 z - \theta_2 z^2) * (1 - \theta_1 z^{-1} - \theta_2 z^{-2})$
 $\gamma_0 = \sigma^2(1 + \theta_1^2 + \theta_2^2); \quad \gamma_1 = \sigma^2(-\theta_1(1 - \theta_2)); \quad \gamma_2 = \sigma^2(-\theta_2)$
 $\gamma_3 = \gamma_4 = \dots = 0 \quad (\text{y es simétrica})$
- $\rho = \frac{1}{\gamma_0} \gamma = \frac{1}{\sigma^2(1 + \theta_1^2 + \theta_2^2)} \gamma$

- $f(\omega) = \frac{\sigma^2}{2\pi} \left((1 + \theta_1^2 + \theta_2^2) - 2\theta_1(1 + \theta_2)\cos(\omega) - 2\theta_2\cos(2\omega) \right)$
- π tiene una expresión complicada, pero su magnitud decae paulatinamente.

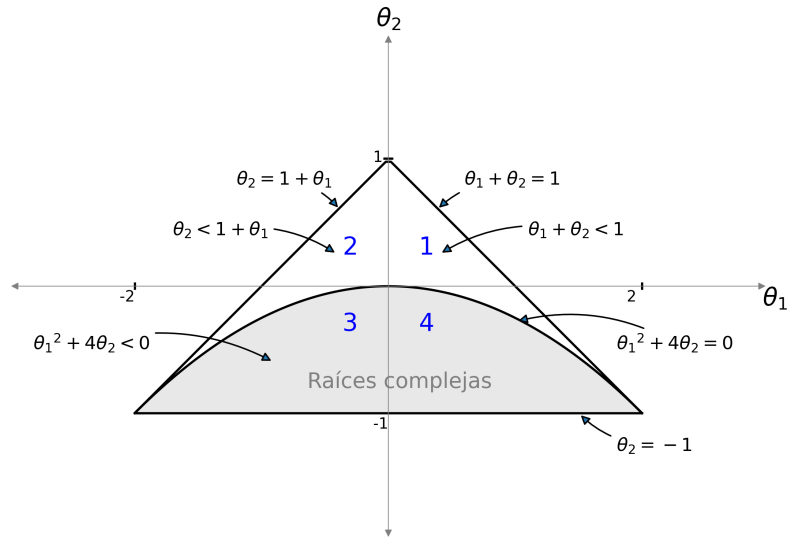


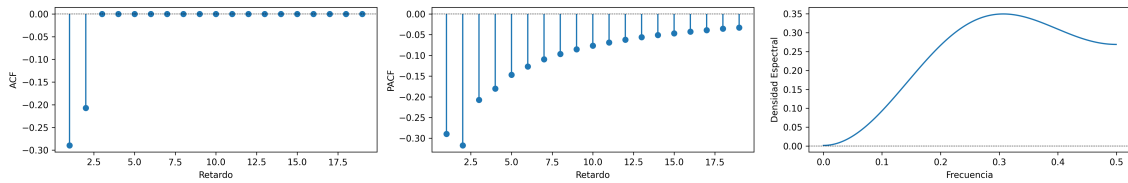
Figura 1: El interior del triángulo es la región paramétrica donde un MA(2) es invertible.

4.5.1. MA(2) con dos parámetros positivos (+, +)

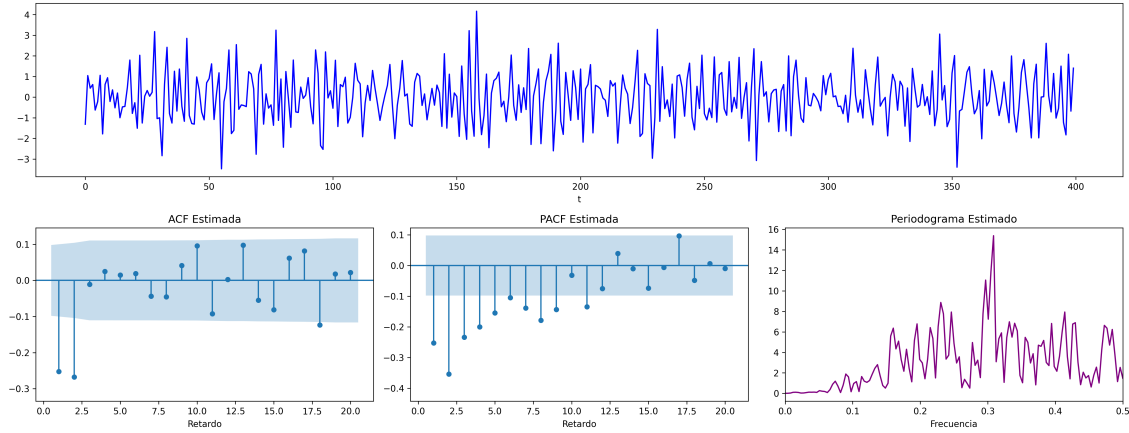
$$\theta(z) = 1 - 0,6z - 0,3z^2 \Rightarrow X_t = (1 - 0,6B - 0,3B^2)U_t; \quad (\theta_1 > 0 \text{ y } \theta_2 > 0)$$

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
1.081666	0.0	1.081666	0.0	Inf	1
-3.081666	0.0	3.081666	0.5	2.0	1

```
ar_params = [1,]
ma_params = [1, -0.6, -0.3]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA2thpp.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA2thpp.png', dpi=300, bbox_inches='tight')
```



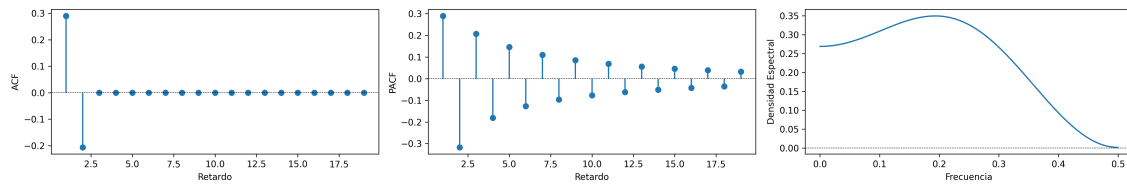
4.5.2. MA(2) con parámetros de distinto signo (-, +)

$$\theta(z) = 1 + 0,6z - 0,3z^2 \Rightarrow X_t = (1 + 0,6B - 0,3B^2)U_t \quad (\theta_1 < 0 \text{ y } \theta_2 > 0)$$

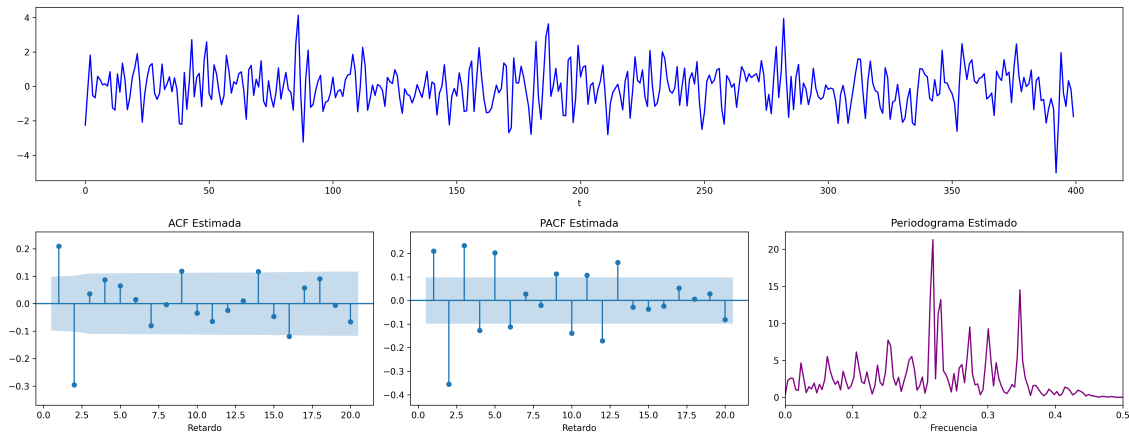
#: 79133a7c-9a4e-439f-aac2-9d8737653e65

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
3.081666	0.0	3.081666	0.0	Inf	1
-1.081666	0.0	1.081666	0.5	2.0	1

```
ar_params = [1,]
ma_params = [1, +0.6, -0.3]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA2thnp.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA2thnp.png', dpi=300, bbox_inches='tight')
```

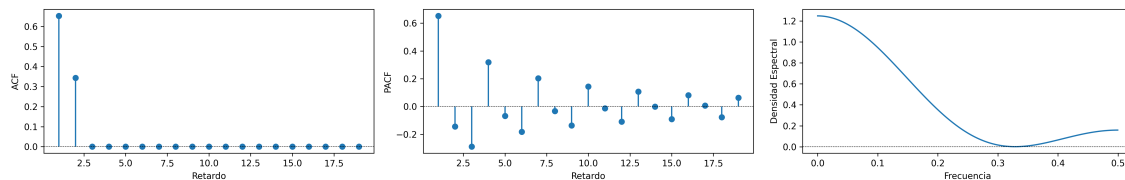


4.5.3. MA(2) con dos parámetros negativos (−, −)

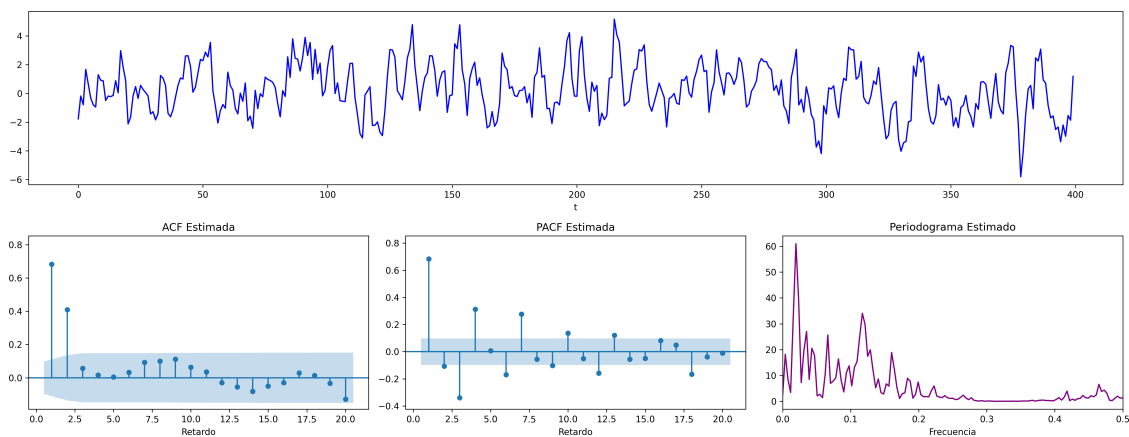
$$\theta(z) = 1 + 0,9z + 0,9z^2 \Rightarrow X_t = (1 + 0,9B + 0,9B^2)U_t; \quad (\theta_1 < 0 \text{ y } \theta_2 < 0)$$

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
-0.5	-0.927961	1.054093	0.328657	3.042687	1.0
-0.5	0.927961	1.054093	0.328657	3.042687	1.0

```
ar_params = [1,]
ma_params = [1, +0.9, 0.9]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA2thnn.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA2thnn.png', dpi=300, bbox_inches='tight')
```

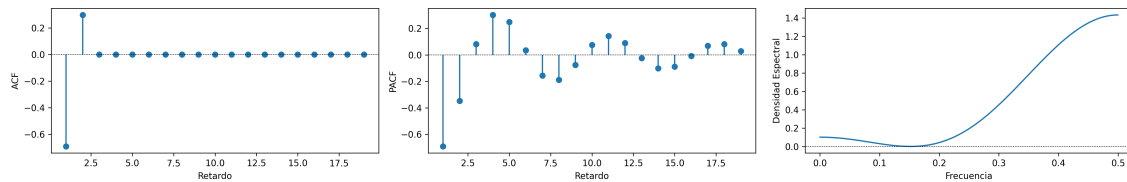


4.5.4. MA(2) con parámetros de distinto signo (+, -)

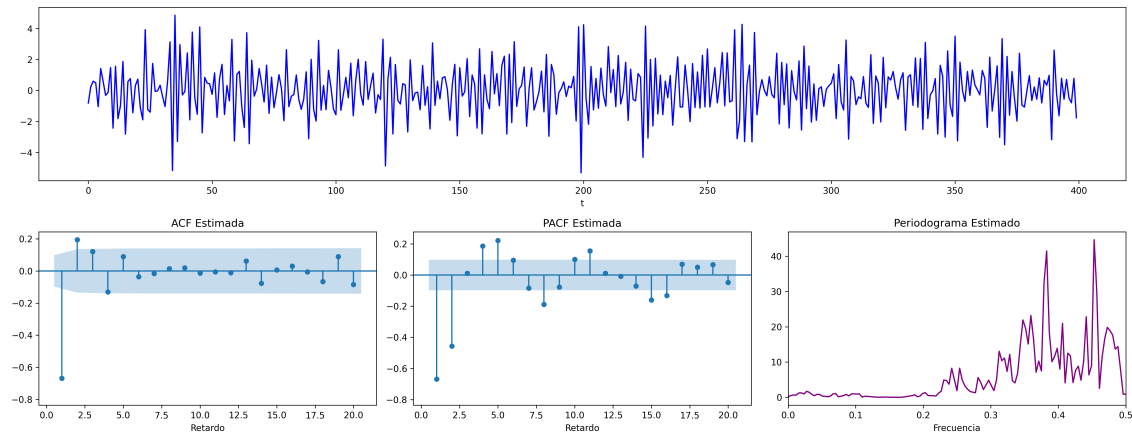
$$\theta(z) = 1 - 1,1z + 0,9z^2 \Rightarrow X_t = (1 - 1,1B + 0,9B^2)U_t \quad (\theta_1 > 0 \text{ y } \theta_2 < 0)$$

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
0.611111	-0.858868	1.054093	0.151575	6.597397	1.0
0.611111	0.858868	1.054093	0.151575	6.597397	1.0

```
ar_params = [1,]
ma_params = [1, -1.1, 0.9]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA2thpn.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA2thpn.png', dpi=300, bbox_inches='tight')
```

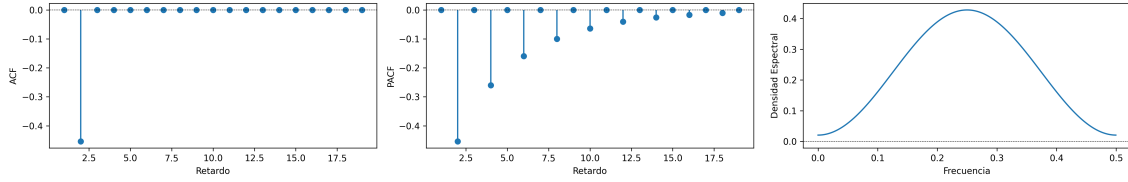


4.5.5. MA(2) con dos raíces reales, una positiva y la otra negativa

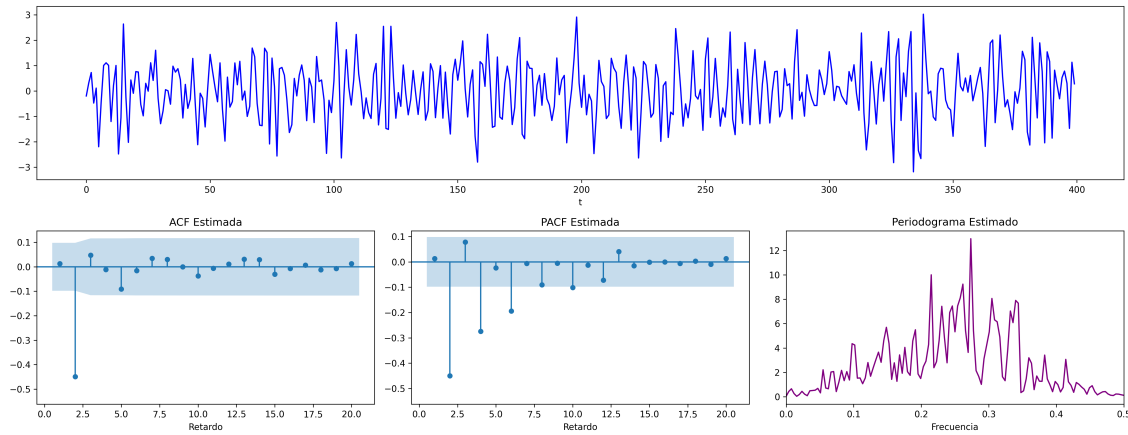
$$\theta(z) = 1 - 0,64z^2 \Rightarrow X_t = (1 - 0,64B^2)U_t \quad (\text{raíces reales: } +, -)$$

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
1.25	0.0	1.25	0.0	Inf	1
-1.25	0.0	1.25	0.5	2.0	1

```
ar_params = [1,]
ma_params = [1, 0, -0.64]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA2pn.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA2pn.png', dpi=300, bbox_inches='tight')
```

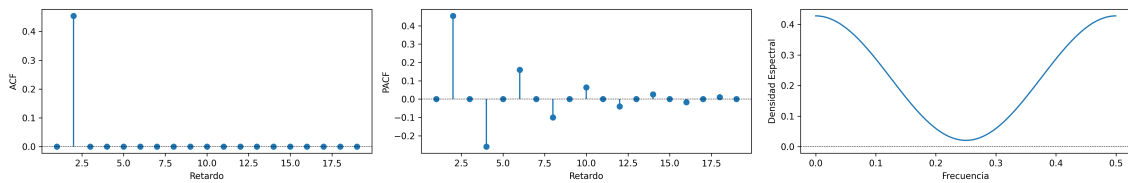


4.5.6. MA(2) con un par de raíces complejas

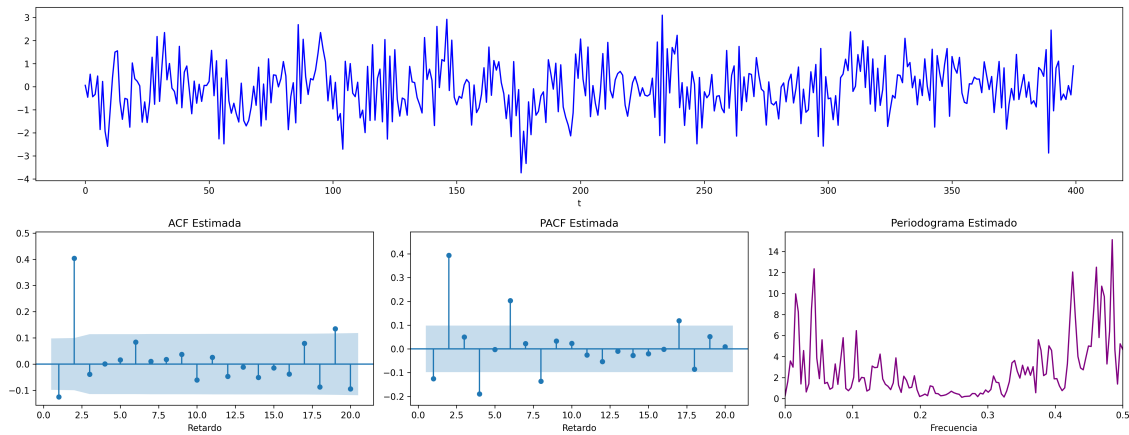
$$\theta(z) = 1 + 0,64z^2 \Rightarrow X_t = (1 + 0,64B^2)U_t \text{ (par de raíces complejas)}$$

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
0.0	-1.25	1.25	0.25	4.0	1.0
0.0	1.25	1.25	0.25	4.0	1.0

```
ar_params = [1,]
ma_params = [1, 0, 0.64]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA2np.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA2np.png', dpi=300, bbox_inches='tight')
```



4.5.7. MA(2) con dos raíces reales y negativas

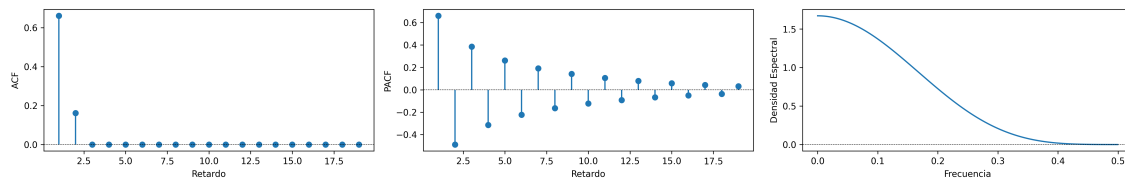
$$\theta(z) = 1 + 1,6z + 0,64z^2 \Rightarrow X_t = (1 + 1,6B + 0,64B^2)U_t \text{ (r. reales: -, -)}$$

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
-1.25	0.0	1.25	0.5	2.0	2.0

```

ar_params = [1,]
ma_params = [1, 1.6, 0.64]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA2rojo.png', dpi=300, bbox_inches='tight')

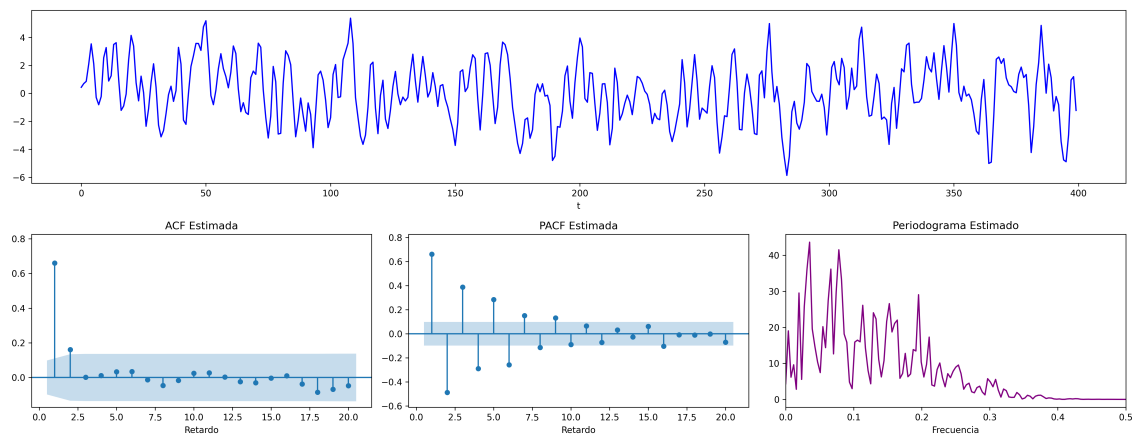
```



```

fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA2rojo.png', dpi=300, bbox_inches='tight')

```

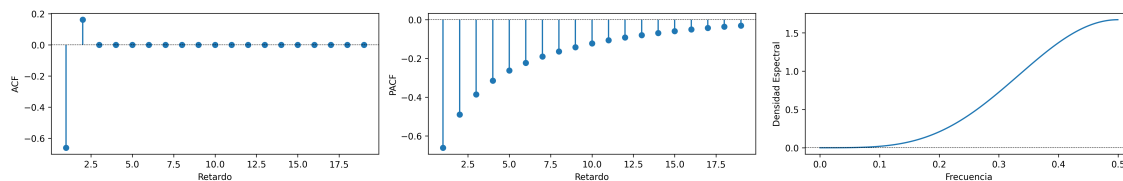


4.5.8. MA(2) con dos raíces reales y positivas

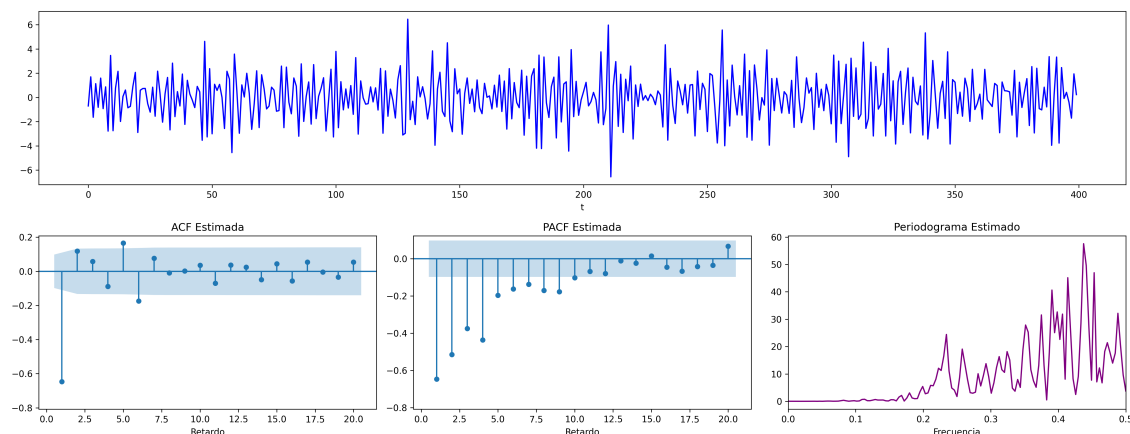
$$\theta(z) = 1 - 1,6z + 0,64z^2 \Rightarrow X_t = (1 - 1,6B + 0,64B^2)U_t \text{ (r. reales: +, +)}$$

P. real	P. imag.	Módulo	Frecuencia	Periodo	Multip.
1.25	0.0	1.25	0.0	Inf	2

```
ar_params = [1,]
ma_params = [1, -1.6, 0.64]
fig = plot_arma_parametric_diagnostics(ar_params, ma_params, sigma2=1, lags=20)
fig.savefig('./img/lecc04/ACF-MA2violeta.png', dpi=300, bbox_inches='tight')
```



```
fig = plot_arma_analysis(ar_params, ma_params)
fig.savefig('./img/lecc04/Sim-MA2violeta.png', dpi=300, bbox_inches='tight')
```



5. (Resumen) características de procesos MA(q)

Siempre estacionario.

Para ser invertible raíces θ fuera del círculo unidad ($\theta^{-\triangleright} = \theta^{-1} \in \ell^1$)

Tipos de representación del proceso

Como suma ponderada finita (número finito de parámetros)

$$\mathbf{X} = \theta * \mathbf{U} \Rightarrow X_t = \theta(B)U_t$$

Como suma ponderada infinita (solo existe si es invertible):

$$\frac{1}{\theta} * X = U \Rightarrow \frac{1}{\theta}(B)X_t = U_t$$

Comportamiento de las funciones asociadas a los segundos momentos

ACF (ρ): Grado (q) y cogrado ($-q$)

PACF (π): Grado (∞) y cogrado ($-\infty$): Combinación de exponenciales y/o sinusoidales amortiguadas.

Densidad espectral Suma de dos cosenos más una constante.

5.1. Gráfico raíces MA(2)

```
def MA2_param_plot():  
  
    fig, ax = plt.subplots(figsize=(10, 6))  
    ax.set_aspect('equal')  
  
    # Set axis  
    xmin, ymin = -3, -2  
    xmax, ymax = -xmin, -ymin  
    plt.axis([xmin, xmax, ymin, ymax])  
  
    # Set axis labels  
    ax.set(xticks=[], yticks=[])  
    ax.set_xlabel(r'$\theta_2$', fontsize=16)  
    ax.xaxis.set_label_position('top')  
    ax.set_ylabel(r'$\theta_1$', rotation=0, fontsize=16)  
    ax.yaxis.set_label_position('right')  
  
    # Draw (t1, t2) points  
    p1 = np.linspace(-2, 2, 100)  
    ax.plot(p1, -abs(p1) + 1, c='black')  
    ax.plot(p1, np.full_like(p1, -1), c='black')  
    ax.plot(p1, -(p1**2 / 4), c='black')  
  
    # Fill the region below the parabola and above the base of the triangle  
    ax.fill_between(p1, -(p1**2 / 4), -1, where=-(p1**2 / 4) >= -1, color='lightgray', alpha=0.5)  
  
    # Turn normal axes off  
    for spine in ['left', 'bottom', 'top', 'right']:  
        ax.spines[spine].set_visible(False)  
  
    # Add arrows to represent axes  
    axes_arrows = {'arrowstyle': '<|-|>', 'lw': 0.5, 'color': 'gray'} # Hacerlas grises y más finas  
    ax.annotate('', xy=(xmin, 0), xytext=(xmax, 0), arrowprops=axes_arrows)  
    ax.annotate('', xy=(0, ymin), xytext=(0, ymax), arrowprops=axes_arrows)  
  
    # Annotate the plot with equations  
    plot_arrowsl = {'arrowstyle': '-|>', 'connectionstyle': 'arc3, rad=-0.2'}  
    plot_arrowsr = {'arrowstyle': '-|>', 'connectionstyle': 'arc3, rad=0.2'}  
    ax.annotate(r'$\theta_1 + \theta_2 < 1$', xy=(0.4, 0.25), xytext=(1.1, 0.4),  
               arrowprops=plot_arrowsr, fontsize='12')  
    ax.annotate(r'$\theta_1 + \theta_2 = 1$', xy=(0.38, 0.6), xytext=(0.6, 0.8),  
               arrowprops=plot_arrowsr, fontsize='12')
```

```

ax.annotate(r'$\theta_2 < 1 + \theta_1$', xy=(-0.4, 0.25), xytext=(-1.7, 0.4),
            arrowprops=plot_arrowsl, fontsize='12')
ax.annotate(r'$\theta_2 = 1 + \theta_1$', xy=(-0.38, 0.6), xytext=(-1.2, 0.8),
            arrowprops=plot_arrowsl, fontsize='12')
ax.annotate(r'$\theta_2 = -1$', xy=(1.5, -1), xytext=(1.8, -1.3),
            arrowprops=plot_arrowsl, fontsize='12')
ax.annotate(r'$\{\theta_1\}^2 + 4\theta_2 = 0$', xy=(1.0, -0.28),
            xytext=(1.8, -0.5), arrowprops=plot_arrowsr, fontsize='12')
ax.annotate(r'$\{\theta_1\}^2 + 4\theta_2 < 0$', xy=(-.9, -0.6),
            xytext=(-2.8, -0.5), arrowprops=plot_arrowsl, fontsize='12')

# Label complex region
ax.text(0.09, -0.8, 'Raíces complejas', ha='center', fontsize=14, color='gray')

# Label cases
ax.text(.3, 0.25, '1', ha='center', fontsize=16, color='blue')
ax.text(-.3, 0.25, '2', ha='center', fontsize=16, color='blue')
ax.text(-.3, -0.35, '3', ha='center', fontsize=16, color='blue')
ax.text(.3, -0.35, '4', ha='center', fontsize=16, color='blue')

# Add small marker to y-axis
ax.axhline(y=1.005, xmin=0.495, xmax=0.505, c='black')
ax.text(-0.12, -1.12, '-1', fontsize=10)
ax.text(-0.12, 0.98, '1', fontsize=10)

# Add small markers to x-axis
ax.axvline(x=-2, ymin=0.495, ymax=0.505, c='black')
ax.axvline(x=2, ymin=0.495, ymax=0.505, c='black')
ax.text(-2.12, -0.12, '-2', fontsize=10)
ax.text(1.88, -0.12, '2', fontsize=10)

return fig
MA2_param_plot().savefig('./img/lecc04/MA2roots.png', dpi=300, bbox_inches='tight')

```
