SERIES TEMPORALES
Clase 2

MÁSTER EN BIG DATA,

DATA SCIENCE E INTELIGENCIA ARTIFICIAL

Juan Antonio Guevara Gil

juanguev@ucm.es

2.1. Introducción.

- Introducción.
- Función de autocorrelación simple y función de autocorrelación parcial.
- Modelo ARMA (p,q).
  - Modelo autorregresivo AR(p).
  - Modelo de medias móviles MA(q).
- Modelo ARIMA (p,d,q).
- Modelo ARIMA estacional.
- La metodología Box-Jenkins.
- Transformaciones para estabilidad la varianza.
- Identificación y estimación del modelo ARIMA.
- Diagnosis del modelo.
  - Significación estadística de los parámetros.
  - Análisis de los residuos.
  - Medidas de la adecuación del modelo.
- Cálculo de predicciones.

2.1. Introducción.

Una **serie temporal** es una realización de un proceso estocástico, donde los elementos están ordenados y corresponden a instantes equidistantes del tiempo. Un **proceso estocástico** es una sucesión de variables aleatorias que evolucionan en función de otra variable, generalmente el tiempo. Algunos **conceptos básicos de procesos estocásticos**:

- **Estacionarios**:

  Toman valores estables en el tiempo alrededor de un valor central, sin mostrar una tendencia o crecer o decrecer a lo largo del tiempo.

- **No estacionarios**:

  Pueden mostrar tendencia, estacionalidad y otros efectos evolutivos en el tiempo.

2.1. Introducción.

Un **proceso estocástico es estacionario** en sentido estricto cuando las distribuciones marginales de cualquier conjunto de *k* variables **son idénticas, en distribución y parámetros**.

Basta con que el proceso sea estacionario en sentido débil:

$$\begin{cases} \mu_t = \mu \quad \forall t \\ \sigma_t^2 = \sigma^2 \quad \forall t \\ Cov(x_t, x_{t+k}) = E\left[(x_t - \mu)(x_{t-k} - \mu)\right] = \gamma_k \quad \forall k \end{cases}$$

La media y la varianza permanecen constantes con el tiempo. La covarianza entre dos variables de la serie depende sólo de su separación en el tiempo.

$$\rho_k = \frac{Cov(x_t, x_{t-k})}{\sqrt{Var(x_t)Var(x_{t-k})}} = \frac{\gamma_k}{\gamma_0} \qquad \gamma_0 = \sigma_{x_t}^2 \quad \forall t$$

2.1. Introducción.

Una **propiedad** importante de los procesos estacionarios es que **son estables ante las combinaciones lineales**. En particular, los incrementos de una serie estacionaria son estacionarios.

$$\omega_t = x_t - x_{t-1}$$

Un proceso estocástico es un **ruido blanco** si:

$$E[X(t)] = 0 \quad V[X(t)] = \sigma^2 \quad \gamma_k = 0 \quad \forall k = 1,2, \ldots$$

Es decir:

- El valor esperado de la variable es cero en cualquier instante temporal, no teniendo sesgos hacia valores positivos o negativos.
- La amplitud de las fluctuaciones no cambia a lo largo del tiempo.
- Las correlaciones entre los valores de la serie en distintos momentos son nulas.

2.1. Introducción.

El **operador de retardo** es un operador **lineal** que, aplicado a una función temporal, describe como los valores cambian a lo largo del tiempo. Se define el operador de retardo **B**:

$$BX_t = X_{t-1} \quad B^2X_t = X_{t-2} \quad B^kX_t = X_{t-k}$$

El operador de retardo B te permite expresar de manera concisa y sistemática cómo los valores de una serie temporal cambian en función del tiempo pasado.

2.2. Función de autocorrelación simple (ACF) y Función de autocorrelación parcial (PACF).

Al observar un determinado proceso estacionario $X = \{x_1, x_2, \dots, x_T\}$, se pueden calcular los estimadores de los parámetros anteriores ($E[X], V[X], \gamma_k$):

- El estimador de la media es: $\hat{\mu} = \bar{x}$
- El estimador de la varianza es: $Var(\bar{x}) = \frac{1}{T}\left[\sigma^2 + 2\sum_{i=1}^{T-1}\left(1-\frac{i}{T}\right)\gamma_i\right] = \frac{\gamma_0}{T}\left[1 + 2\sum_{i=1}^{T-1}\left(1-\frac{i}{T}\right)\rho_i\right]$
- El estimador de la autocovarianza de orden *k* es: $\hat{\gamma}_k = \frac{1}{T}\sum_{t=k+1}^{T}(x_t - \bar{x})(x_{t-k} - \bar{x})$
- El estimador del coeficiente de correlación es: $r_k = \frac{\hat{\gamma}_k}{\hat{\gamma}_0}$

2.2. Función de autocorrelación simple (ACF) y Función de autocorrelación parcial (PACF).

La función de autocorrelación muestral (ACF) es una medida estadística que **representa la correlación de una serie temporal con ella misma** a diferentes rezagos (lags). La ACF se expresa en función de los coeficientes de autocorrelación muestrales, como γ_k, que miden la correlación entre la serie y sí misma desplazada por *k* periodos de tiempo. Su representación es el **correlograma**.

Estas cantidades miden la relación lineal entre las variables de la serie separadas por *k* posiciones.

La Función de Autocorrelación Parcial (PACF) es una medida de la correlación entre observaciones de una serie de tiempo que están separadas por *k* unidades de tiempo.

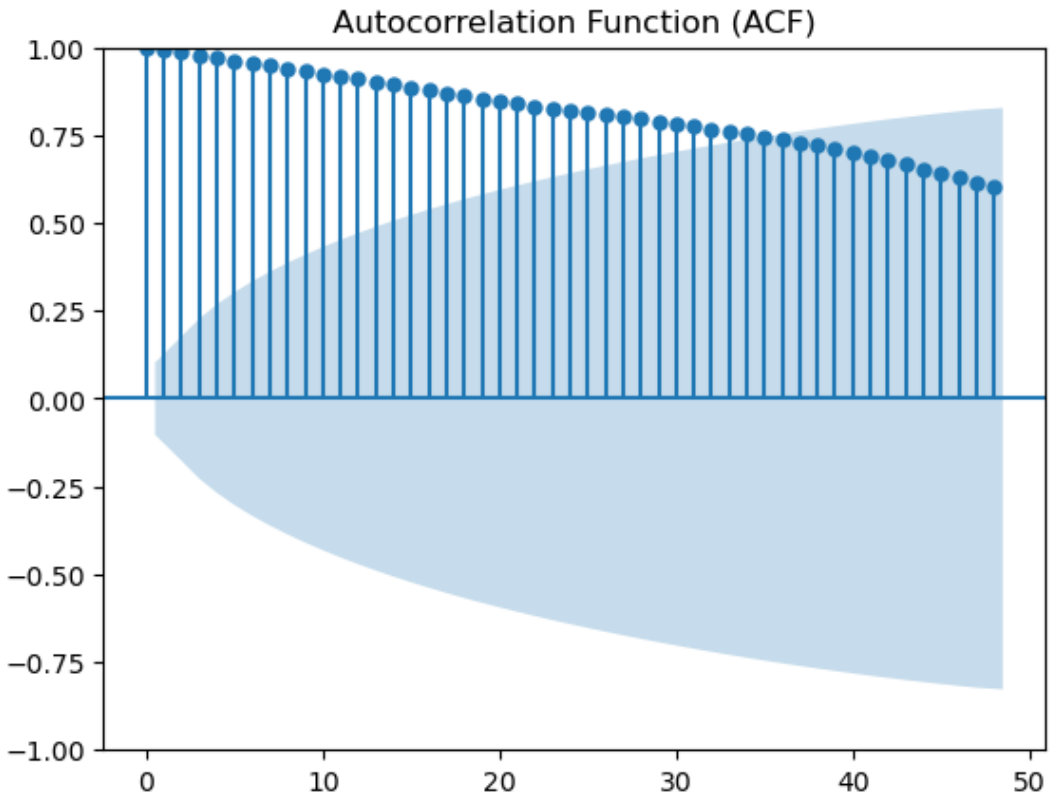
Se utiliza para examinar la correlación directa entre dos puntos en el tiempo, excluyendo las contribuciones de los rezagos intermedios y se estima mediante una regresión de X_t sobre $X_{t-1}, X_{t-2}, \dots, X_{t-k}$.

2.2. Función de autocorrelación simple (ACF) y Función de autocorrelación parcial (PACF).

La función de autocorrelación muestral (ACF) ayuda a identificar si una serie es o no estacionaria.

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

# Plot ACF

plot_acf(Bitcoin_A['PRECIO'], lags=48, alpha = 0.05)

plt.title('Autocorrelation Function (ACF)')

plt.show()
```

- Si el correlograma decrece lentamente, la serie no es estacionaria.
- Si se corta o decrece rápidamente, es estacionaria.
- Se valora a la vez que PACF para conocer qué modelo se ajusta mejor, y el orden de los parámetros.

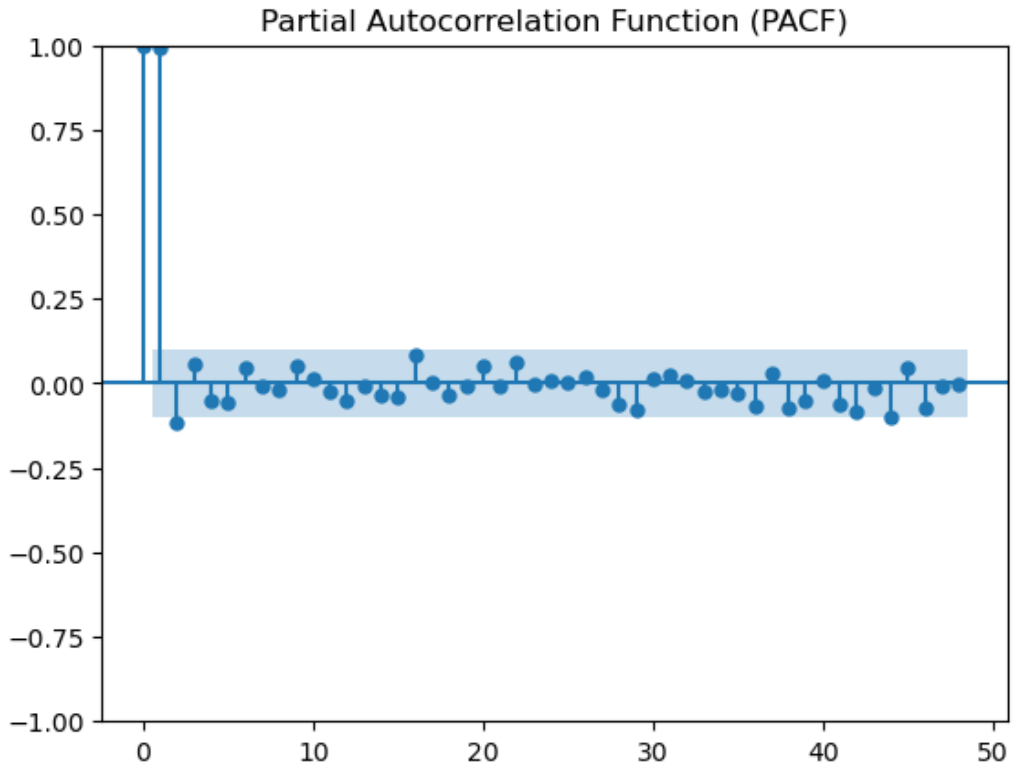


2.2. Función de autocorrelación simple (ACF) y Función de autocorrelación parcial (PACF).

La función de autocorrelación parcial (PACF) ayuda a identificar el modelo autorregresivo a utilizar, e.g.: AR(p) sólo tiene en cuenta los *p* primeros coeficientes de correlación parcial ≠ 0.

```
# Plot PACF
plot_pacf(Bitcoin_A['PRECIO'], lags=48, alpha = 0.05)
plt.title('Partial Autocorrelation Function (PACF)')
plt.show()
```

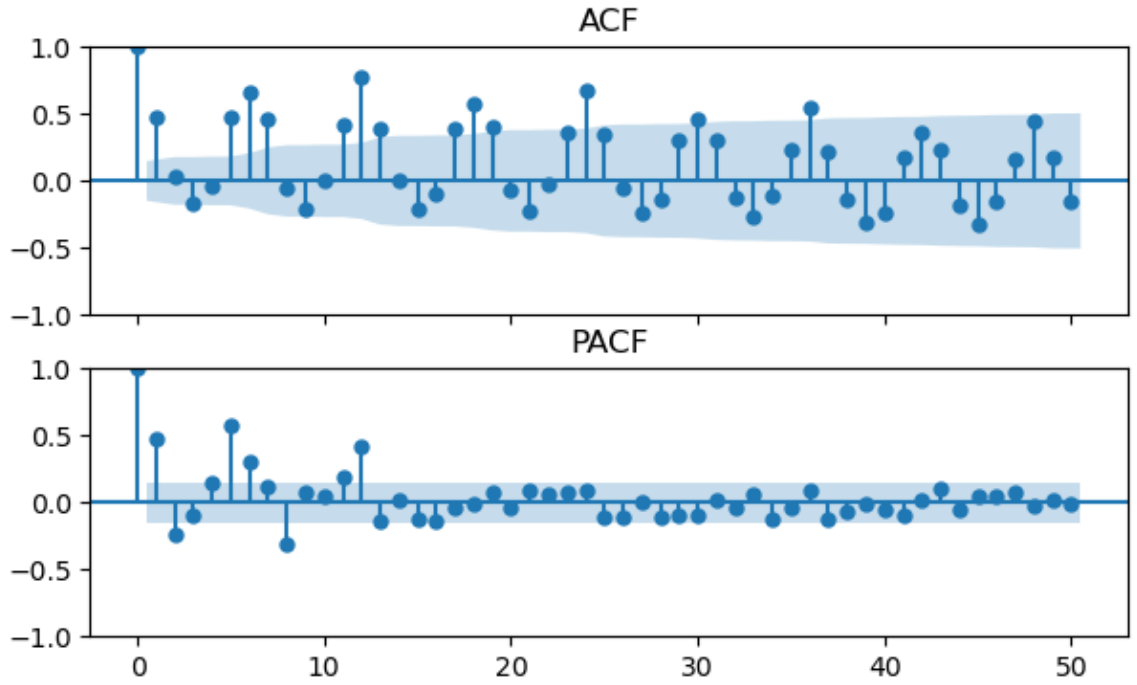
- El sombreado representa las bandas de confianza utilizando el error standard aproximado.
- Este intervalo de confianza es el que comprende los valores de la autocorrelación si el verdadero valor poblacional fuera 0, entendiendo el valor como 0 si se encuentra entre estas.



2.2. Función de autocorrelación simple (ACF) y Función de autocorrelación parcial (PACF).

ACF y PACF en una serie estacional (archivo Córdoba):

```
fig, axs = plt.subplots(nrows=2, ncols=1, figsize=(7, 4), sharex=True)
plot_acf(v_cordoba['V_Resident'], ax=axs[0], lags=50, alpha=0.05)
axs[0].set_title('ACF')
plot_pacf(v_cordoba['V_Resident'], ax=axs[1], lags=50, alpha=0.05)
axs[1].set_title('PACF');
```



Se observa un comportamiento repetitivo de las autocorrelaciones cada 12 meses en la ACF, observando como la autocorrelación más fuerte es en los retardos múltiplos de 12.

2.3. El modelo ARMA(p,q).

Modelo autorregresivo AR(p).

- El **modelo AR(p) generaliza la idea de regresión** para representar la relación entre la variable de interés y sus observaciones pasadas. Existe dependencia lineal entre las distintas observaciones de la variable.

$$AR(1) \equiv X_t = \Phi_1 X_{t-1} + a_t$$

Donde Φ es el coeficiente autorregresivo, a_t es un proceso de ruido blanco con varianza σ^2.

- Esta dependencia puede extenderse a las p observaciones pasadas, siendo un **modelo autorregresivo de orden *p:* AR(p).**

$$AR(p) \equiv X_t = \Phi_1 X_{t-1} + \Phi_2 X_{t-2} + \cdots \Phi_p X_{t-p} + a_t$$

El modelo AR(1) es un proceso estacionario si $|\Phi_1| < 1$.

La función de autocorrelación es: $\rho_k = \Phi_1^k$, la cual crece de forma geométrica.

2.3. El modelo ARMA(p,q).

Modelo autorregresivo AR(p).

❖Así, la ACF de un proceso AR(1) puede tener el siguiente aspecto:

$\Phi_1 > 0$ → La ACF será una función positiva y decreciente.

$\Phi_1 < 0$ → La ACF será una función alternada, y tendrá retardos pares positivas y retardos impares negativos.

❖La PACF:

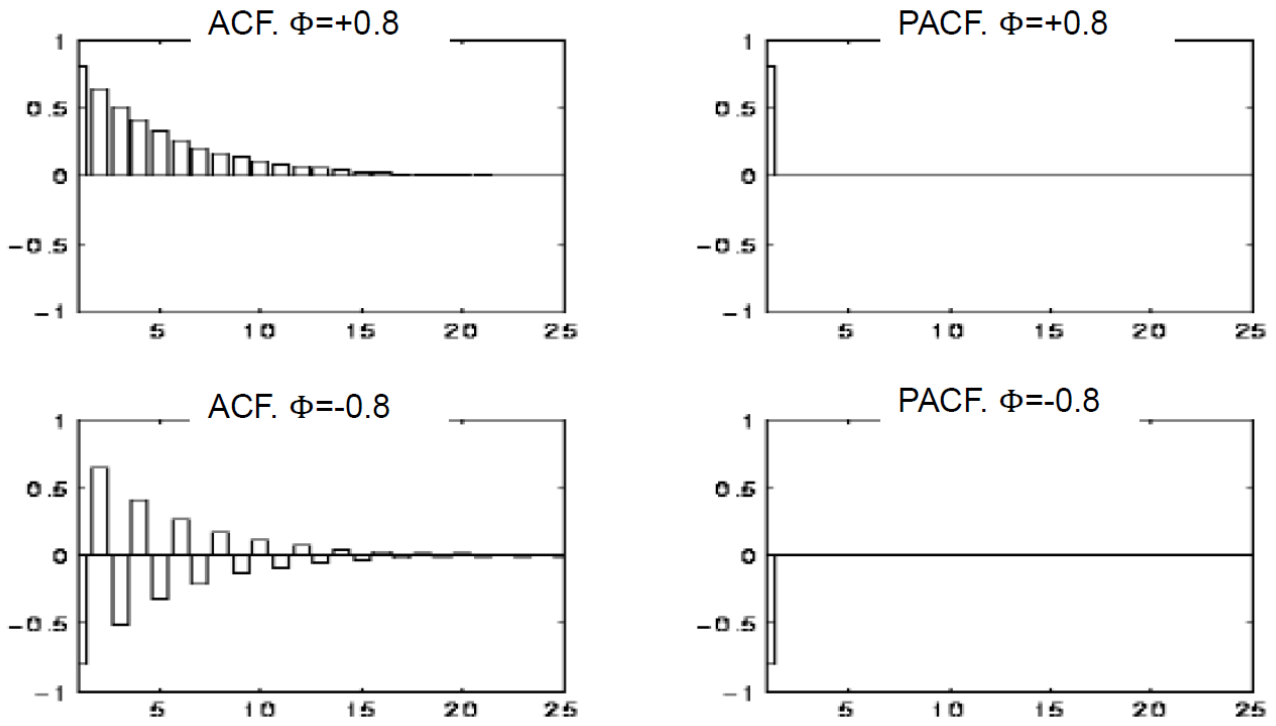
$\Phi_1 > 0$ → PACF tendrá un único retardo significativo y será positivo.

$\Phi_1 < 0$ → PACF tendrá un único retardo significativo y será negativo.

2.3. El modelo ARMA(p,q).

Modelo autorregresivo AR(p).

❖Ejemplos de funciones de autocorrelación para $\Phi_1 \pm 0.8$



2.3. El modelo ARMA(p,q).

Modelo de Medias Móviles MA(q).

Los **modelos de medias móviles** son modelos en el **que la variable se obtiene como un promedio de variables ruido blanco**.

Estos procesos son la suma de procesos estacionarios y por lo tanto sea cual sea el valor del parámetro es estacionario a diferencia de los modelos AR.

El proceso de media móvil de orden 1 MA(1) sólo utiliza el error inmediatamente anterior:

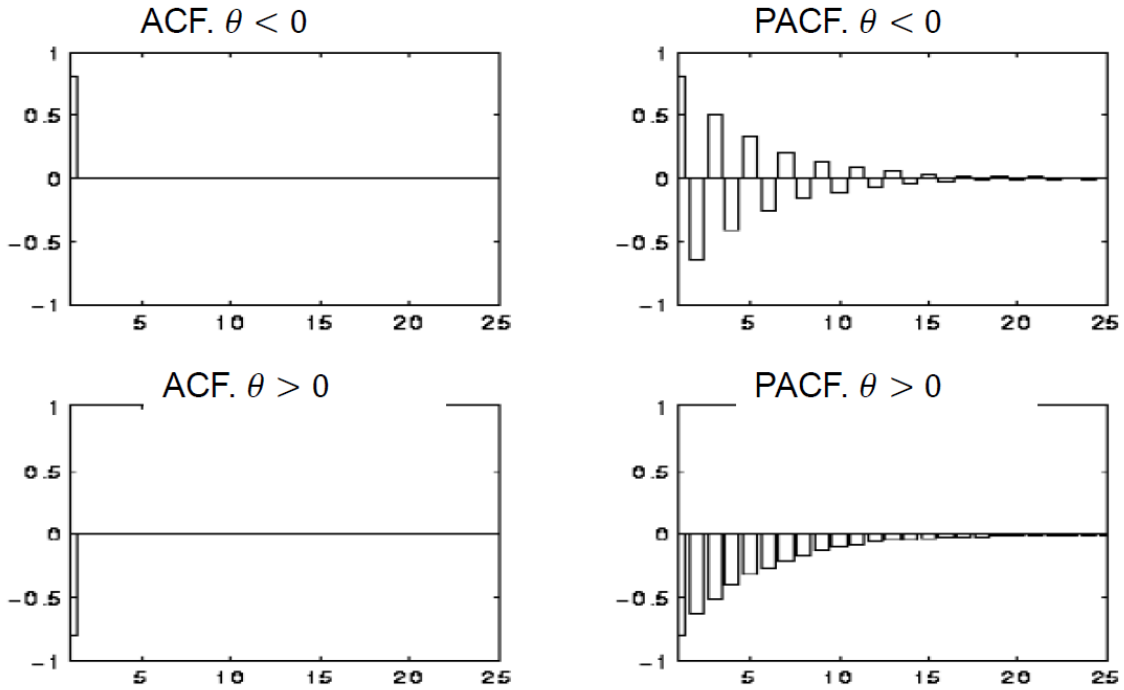
$$X_t = Z_t - \theta_1 Z_{t-1}$$

Donde Z_t es un término de error blanco, y θ_1 es el parámetro que representa la contribución del error Z_{t-1} al valor X_t.

2.3. El modelo ARMA(p,q).

Modelo de Medias Móviles MA(q).

$|\theta_1| < 1$ ➡ El proceso es ***invertible***.



Un proceso es invertible si presenta la capacidad de reconstruir el proceso utilizando información pasada y futura.

La invertibilidad es una propiedad importante porque garantiza que el proceso pueda ser representado de manera única por la combinación de errores pasados y futuros.

2.3. El modelo ARMA(p,q).

Modelo de Medias Móviles MA(q).

El proceso MA(q) es un modelo que tendrá en cuenta los *q* últimos errores, generalizando así la expresión a:

$$X_t = Z_t - \theta_1 Z_{t-1} - \theta_2 Z_{t-2} - \cdots - \theta_q Z_{t-q}$$

Este proceso es la suma de procesos estacionarios y, por lo tanto, sea cual sea el valor de los parámetros, es estacionario.

El proceso es invertible si las raíces del polinomio característico están fuera del círculo unidad (un círculo centrado en el origen con radio 1).

La función de autocorrelación de un proceso MA(q) tiene la misma "forma" que la función de autocorrelación parcial de un modelo AR(q).

Concluimos que **existe una dualidad entre los modelos AR y MA**, de manera que la ACF de un MA es como la PACF de un AR y viceversa.

2.3. El modelo ARMA(p,q).

Modelo ARMA(p,q).

Proceso	Función de autocorrelación (ACF)	Función de autocorrelación parcial (PACF)
MA(q)	Solo los **q** primeros coeficientes son significativos. El resto se anulan bruscamente (coef. 0 para **retardo >q**)	Decrecimiento rápido exponencial atenuado u ondas sinusoidales.
AR(p)	Decrecimiento rápido exponencial atenuado u ondas sinusoidales.	Solo los **p** primeros coeficientes son significativos. El resto se anulan bruscamente (coef. 0 para retardo >**p**)

- Para determinar el modelo ARMA hay que mirar **simultáneamente** la ACF y la PACF. Es necesario determinar el orden p (relativo a la parte autorregresiva AR) y el orden q (relativo a la parte de media móvil MA).
- **Para determinar p**: usar PACF. Si hay p valores (consecutivos, aunque puede haber valores intermedios menos importantes) significativos, el orden es p. A partir de p, la PACF decrece lentamente.
- En paralelo, **para determinar q**: usar ACF. Si hay q retardos significativos, el orden es q. Para ello hay que observar un decrecimiento rápido (exponencial o sinusoidal) en la ACF a partir del retardo siguiente a q.

2.4. El modelo ARIMA (p,d,q).

- Cuando el nivel de la serie no es constante en el tiempo, pudiendo en particular tener tendencia creciente o decreciente, se dice que **la serie no es estacionaria en la media**.
- Cuando la variabilidad se modifica en el tiempo, se dice que la serie **es no estacionaria en la varianza.**

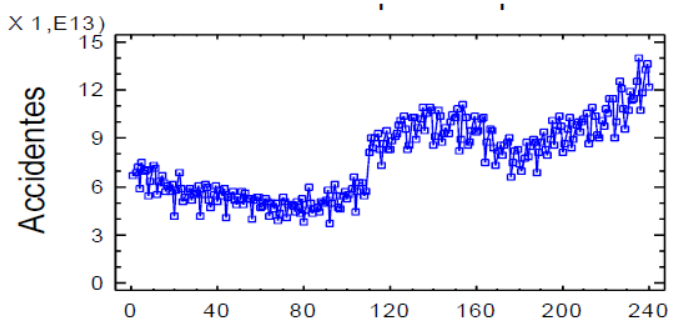
Los procesos no estacionarios más importantes son los procesos integrados, que tienen la propiedad fundamental de que, **al diferenciarlos, se obtienen procesos estacionarios**.

- En un modelo ARIMA(p,d,q), los órdenes p y q son los equivalentes al modelo ARMA(p,q), **tras haber diferenciado la serie previamente (d).**

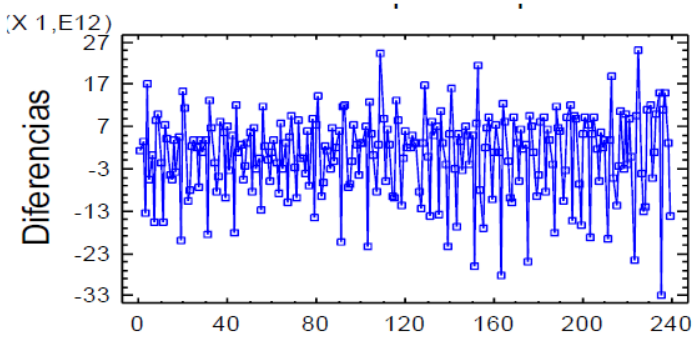
Así, se entiende por *d* como el número de diferenciaciones que hacen falta para convertir la serie no estacionaria en media, en una serie estacionaria.

2.4. El modelo ARIMA (p,d,q).

Frecuentemente las series económicas no son estacionarias pero sus diferencias relativas, o las diferencias cuando medimos la variable en logaritmos, son estacionarias. Por ejemplo, la serie número de accidentes es no estacionaria en media.



$$\omega_t = X_t - X_{t-1}$$



Se dice que un proceso es integrado de orden 1 si $\omega_t = \nabla X_t = X_t - X_{t-1}$ ya es estacionaria.

El modelo ARIMA se usa para **modelar series temporales que exhiben patrones de tendencia y estacionalidad**. El orden de los componentes p, d y q se determina a través de análisis gráficos (como ACF y PACF), pruebas estadísticas y métodos de ajuste.

2.4. El modelo ARIMA (p,d,q).

En los casos en los que es necesario realizar *d* diferencias, se habla de un proceso integrado de orden *d*:

$$\left(1-\phi_1 B-\phi_2 B^2-....-\phi_p B^p\right)\left(1-B\right)^d X_t=\left(1-\theta_1 B-\theta_2 B^2-....-\theta_q B^q\right)Z_t$$

❖**Parte autorregresiva** $\left(1-\Phi_1 B-\Phi_2 B^2-\cdots-\Phi_p B^p\right)$:

Representa el componente autorregresivo, donde Φ son los coeficientes autoregresivos. Multiplica a $(1-B)^d \mathrm{X_t}$, indicando que la relación autorregresiva se aplica después de la diferenciación.

❖**Parte de Diferenciación** $(1-B)^d$:

Este operador se utiliza para hacer la serie estacionaria.

❖**Parte de medias móviles** $\left(1-\theta_1 B-\theta_2 B^2-\cdots-\theta_q B^q\right)Z_t$:

Donde $\theta_1, \ldots \theta_q$ son los coeficientes de medias móviles.

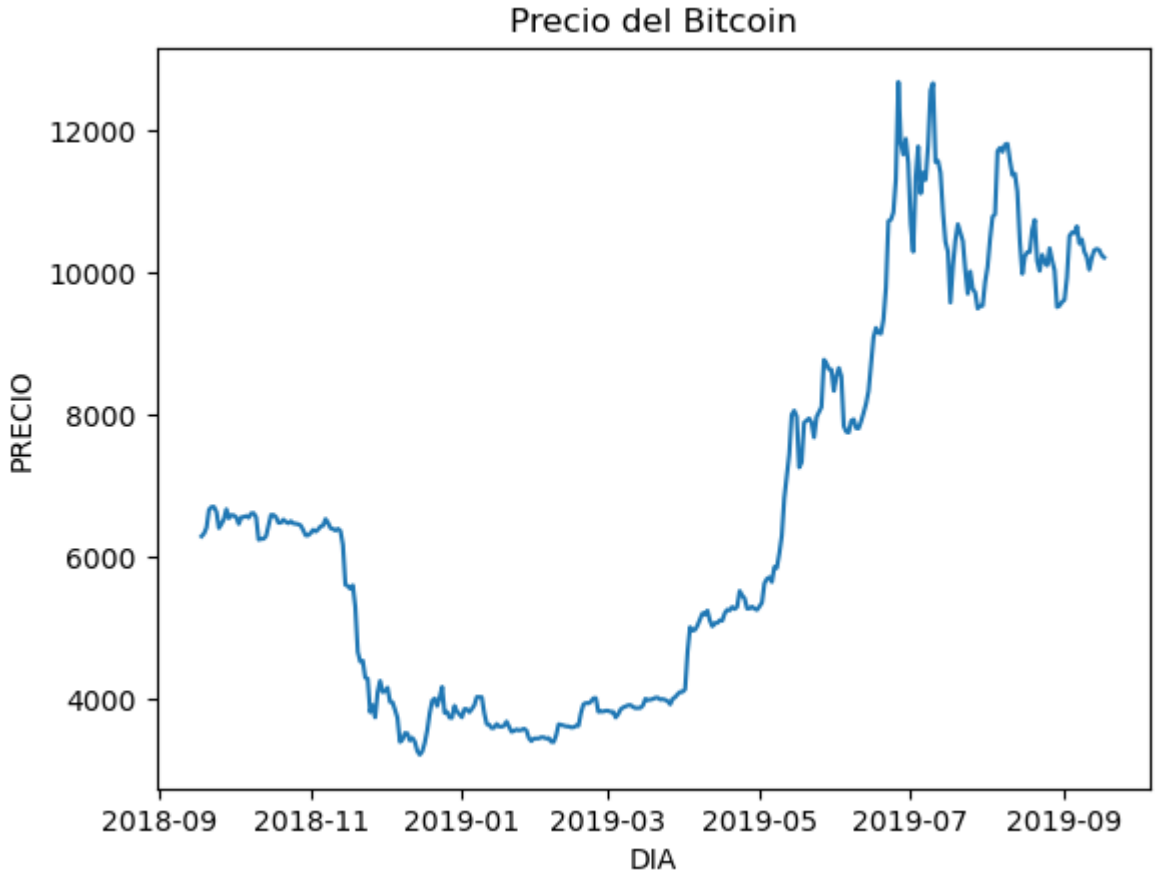
2.4. El modelo ARIMA (p,d,q).

Ejemplo práctico para diferenciar una serie:

precio = Bitcoin_A['PRECIO']

sns.lineplot(precio)

plt.title('Precio del Bitcoin')

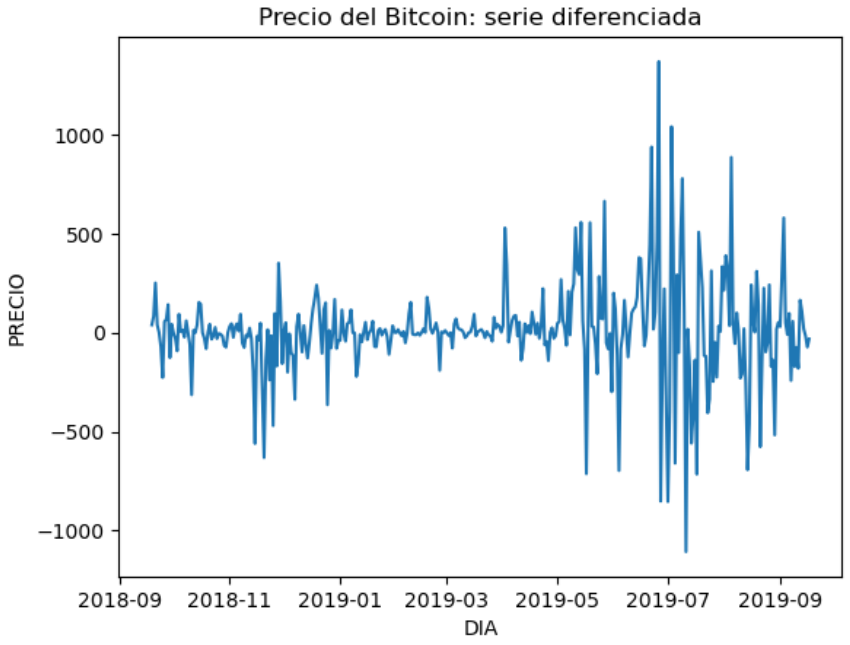


2.4. El modelo ARIMA (p,d,q).

Ejemplo práctico para diferenciar una serie :

A continuación, se representa la serie diferenciada. Recordemos que la diferenciación de una serie es un método que consiste en no hacer ninguna hipótesis sobre la forma de la tendencia a corto plazo y suponer simplemente que evoluciona lentamente en el tiempo. Asumimos que la tendencia en el instante t es muy próxima a la tendencia en el instante $t - 1$, y construimos una nueva serie $y_t = x_t - x_{t-1}$.

```
precio_diff = precio.diff().dropna()
sns.lineplot(precio_diff)
plt.title('Precio del Bitcoin: serie diferenciada')
```

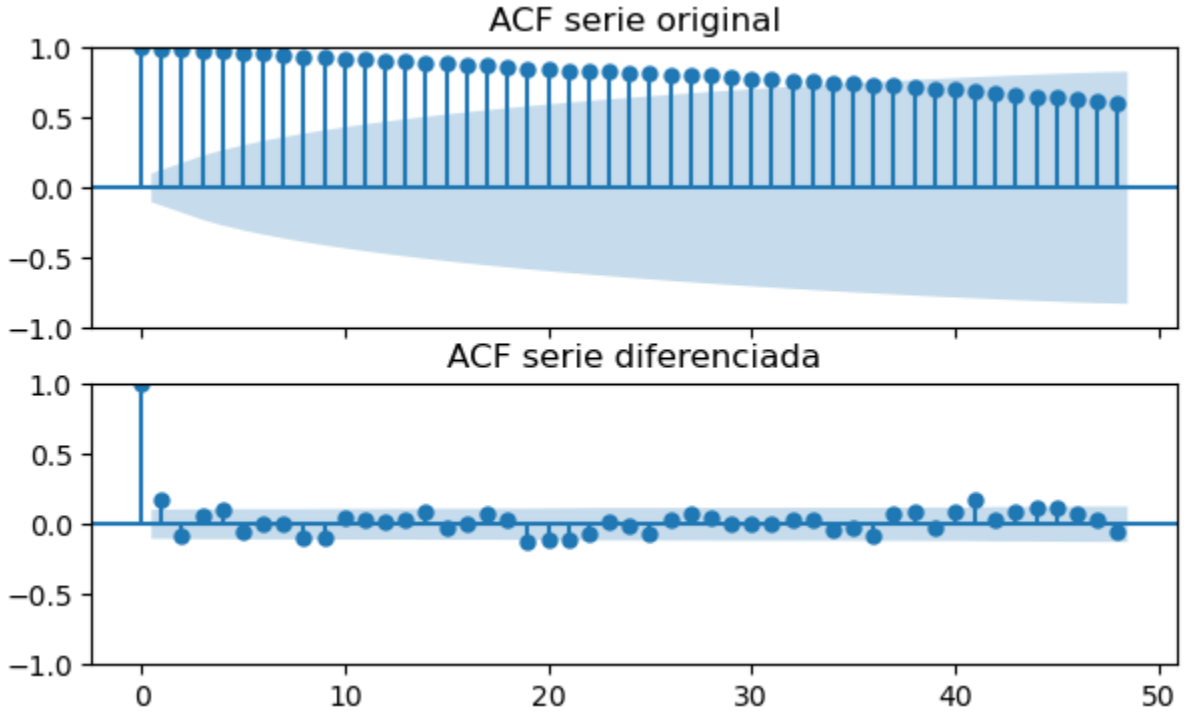


2.4. El modelo ARIMA (p,d,q).

Ejemplo práctico para diferenciar una serie :

A continuación, se calculan las autocorrelaciones simples de la serie original y diferenciada, observando que la serie no estacionaria decrece muy lentamente y de forma lineal:

```
fig, axs = plt.subplots(nrows=2, ncols=1, figsize=(7, 4), sharex=True)

plot_acf(precio, ax=axs[0], lags=48, alpha=0.05)
axs[0].set_title('ACF serie original')

plot_acf(precio_diff, ax=axs[1], lags=48, alpha=0.05)
axs[1].set_title('ACF serie diferenciada');
```



2.5. El modelo ARIMA estacional.

En el tema anterior de métodos descriptivos vimos que podíamos **eliminar la estacionalidad mediante diferencias con los índices estacionales**. Podemos convertir una serie con estacionalidad (con periodicidad o variación en un periodo) en estacionaria (estable a lo largo del tiempo) mediante las diferencias de orden s, siendo s el periodo de la serie.

Definimos el operador diferencia de periodo s o diferencia estacional de orden 1 como:

$$\nabla_s X_t = X_t - X_{t-1} = (1 - B^S)X_t$$

Un modelo estacional general $ARIMA(p,d,q)(P,D,Q)_s$

$$\left(1-\Phi_1 B^s-\Phi_2 B^{2s}-\ldots-\Phi_P B^{Ps}\right)\left(1-\phi_1 B-\phi_2 B^2-\ldots-\phi_p B^p\right)\left(1-B^s\right)^D\left(1-B\right)^d X_t =$$
$$\left(1-\Theta_1 B-\Theta_2 B^{2s}-\ldots-\Theta_Q B^{Qs}\right)\left(1-\theta_1 B-\theta_2 B^2-\ldots-\theta_q B^q\right)Z_t$$

2.5. El modelo ARIMA estacional.

Así, se tiene un **modelo que presenta tanto parte regular, como parte estacional**, permitiendo lidiar con la estacionalidad desde una perspectiva más efectiva al incorporar términos estacionales.

Con respecto a **los parámetros p,d,q y P,D,Q, es importante analizarlos y seleccionarlos por separado**. El análisis de la parte estacional se realiza de forma similar al de la parte regular, pero observando los **retardos correspondientes a la estacionalidad.** Aquí algunos ejemplos:

$$ARIMA(1,0,0)(1,0,0)_{12} \Rightarrow \left(1-\Phi_1 B^{12}\right)\left(1-\phi_1 B\right)X_t = Z_t$$

$$ARIMA(1,0,0)(0,0,1)_{12} \Rightarrow \left(1-\phi_1 B\right)X_t = \left(1-\Theta_1 B^{12}\right)Z_t$$

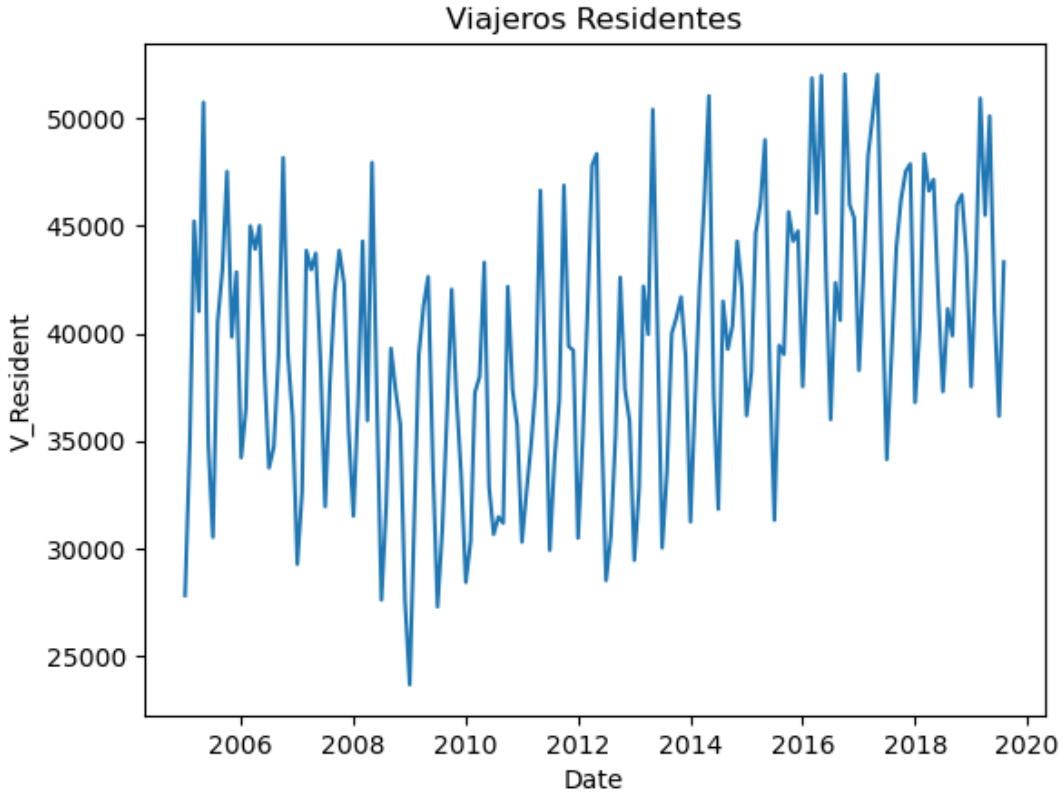
$$ARIMA(2,0,0)(1,0,0)_{12} \Rightarrow \left(1-\Phi_1 B^{12}\right)\left(1-\phi_1 B-\phi_2 B^2\right)X_t = Z_t$$

2.5. El modelo ARIMA estacional.

Ejemplo para determinar los parámetros con “Córdoba”.

```
sns.lineplot(v_cordoba['V_Resident'])
plt.title('Viajeros Residentes’)
```

Se observa que es claramente estacional,
Y “parece” estacionaria, si no fuese por su estacionalidad.

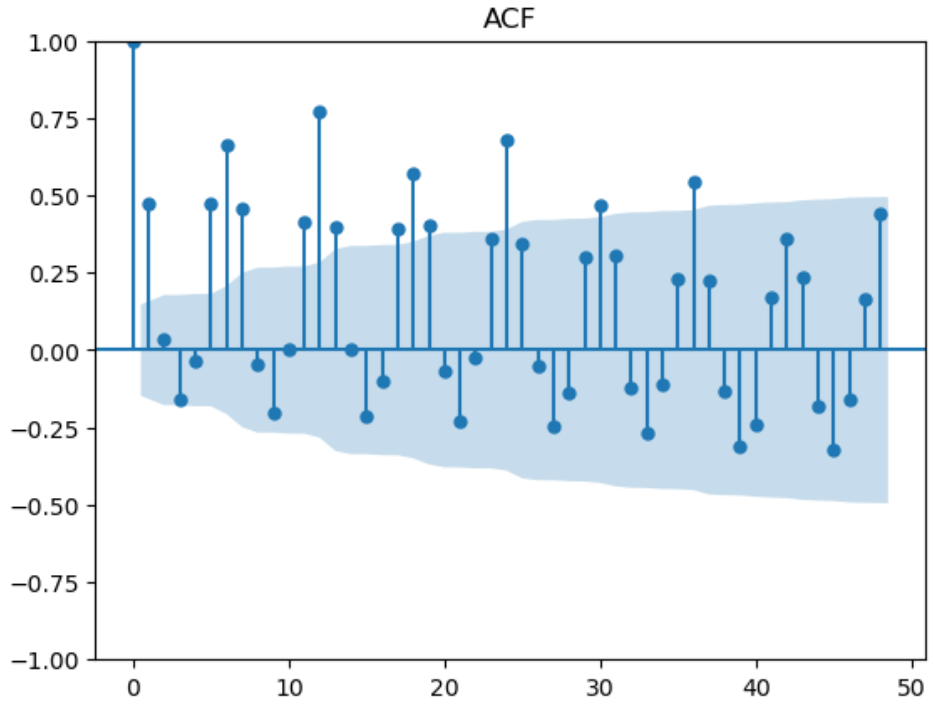


2.5. El modelo ARIMA estacional.

Ejemplo para determinar los parámetros con “Córdoba”.

Si representamos sus correlogramas se observa la estructura estacional y la no estacionariedad en media que se refleja en el gráfico de la serie.

```
plot_acf(v_cordoba['V_Resident'], lags=48, alpha = 0.05)
plt.title('ACF')
```

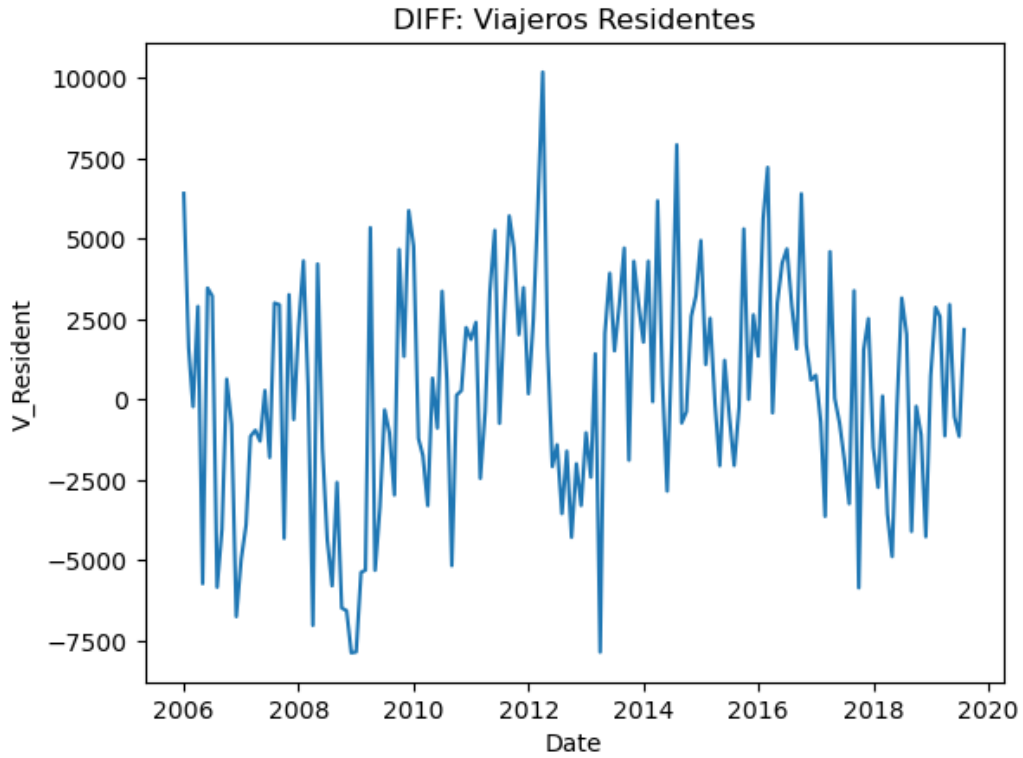


2.5. El modelo ARIMA estacional.

Ejemplo para determinar los parámetros con “Córdoba”.

Si diferenciamos la serie, mediante una diferenciación de orden estacional (12 meses, en este caso), y calculamos sus funciones de autocorrelación se tiene:

```
cordoba_diff = v_cordoba['V_Resident'].diff(12).dropna()
sns.lineplot(cordoba_diff)
plt.title('DIFF: Viajeros Residentes')
```

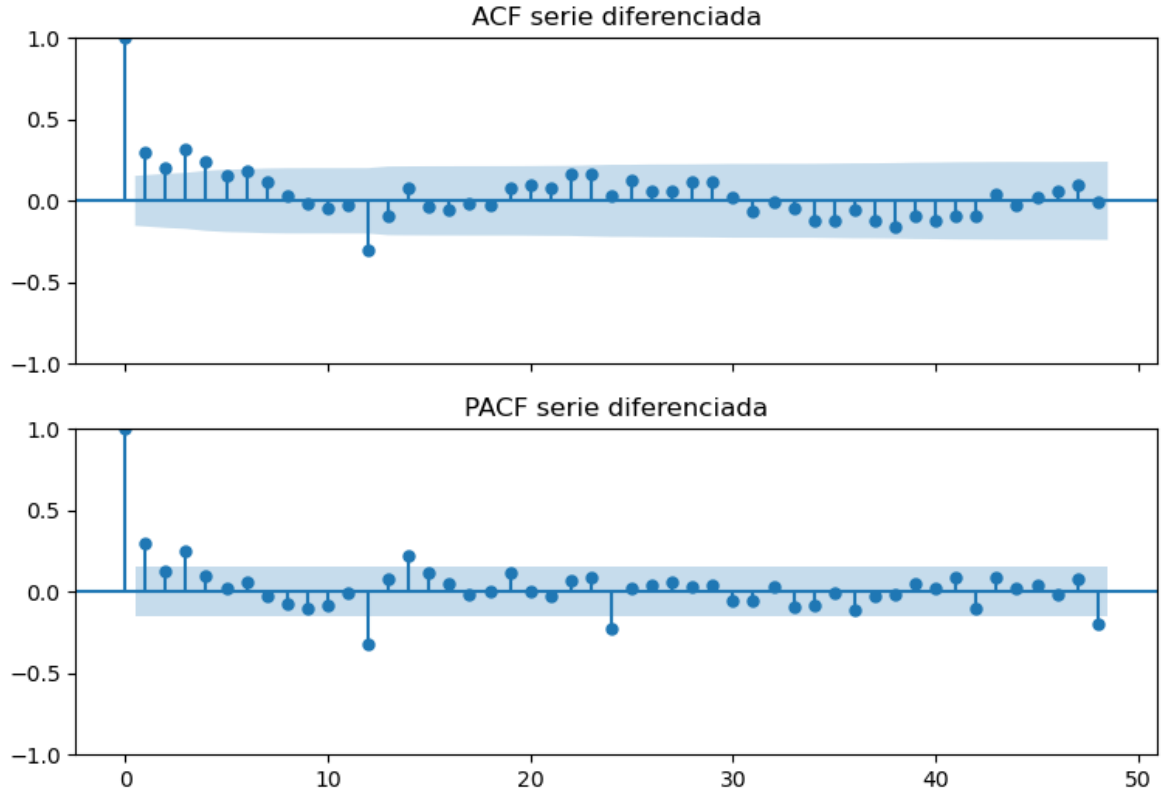


2.5. El modelo ARIMA estacional.

Ejemplo para determinar los parámetros con “Córdoba”.

De nuevo, se calculan las autocorrelaciones simples y parciales de la serie diferenciada:

```
fig, axs = plt.subplots(nrows=2, ncols=1,
figsize=(9, 6), sharex=True)

plot_acf(cordoba_diff, ax=axs[0],
lags=48, alpha=0.05)
axs[0].set_title('ACF serie diff’)

plot_pacf(cordoba_diff, ax=axs[1],
lags=48, alpha=0.05)
axs[1].set_title('PACF serie diferenciada');
```



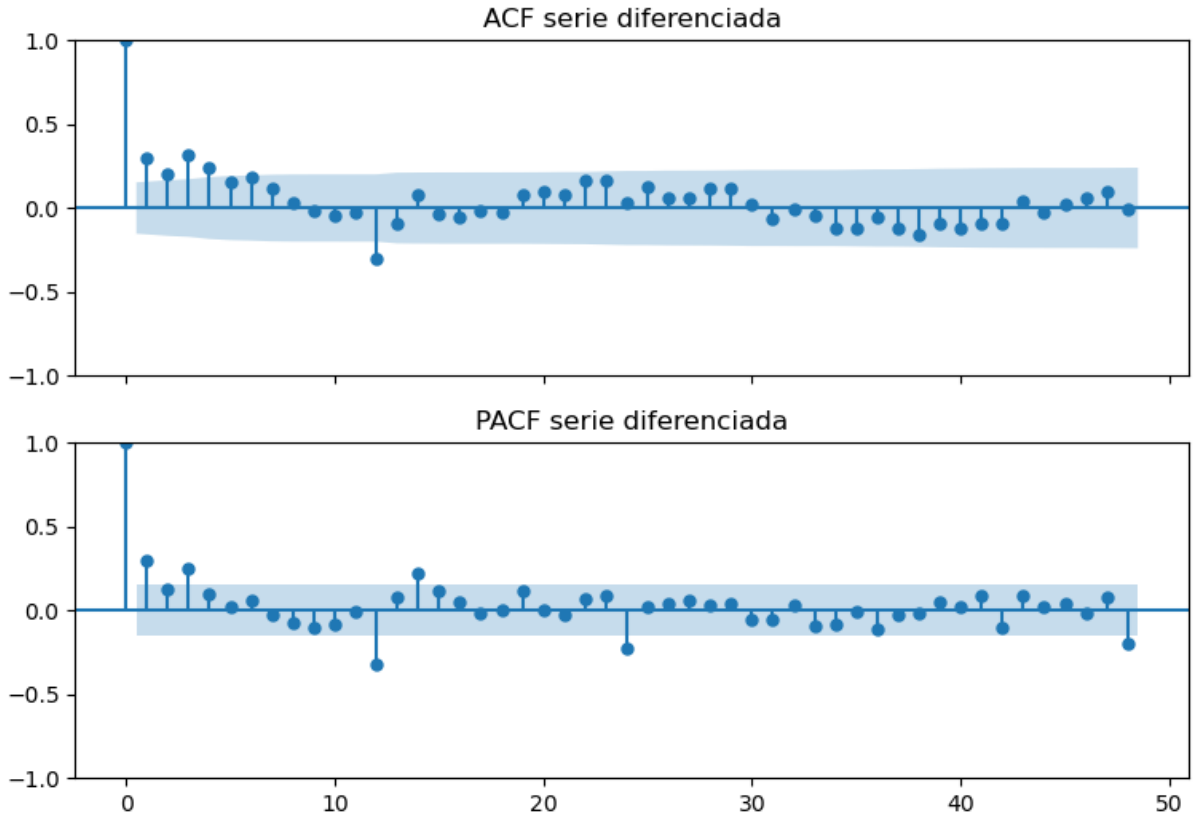
2.5. El modelo ARIMA estacional.

Ejemplo para determinar los parámetros con “Córdoba”.

- Se observa que el proceso ya es estacionario al diferenciar por 12 la serie, por lo que d = 0 y D = 1.
- Se observa PACF para el parámetro p, y las autocorrelaciones caen significativamente a partir del valor 3, por lo que p = 3 y P = 0 (Cada periodicidad de 12, se van observando autocorrelaciones positivas, aunque entre medias haya alguna que no lo sea).
- Se observa ACF para determinar el parámetro q. Al no observar decrecimientos rápidos y significativos a partir de un determinado retardo, q = 0 y Q = 1, al ver que, por cada estacionalidad, sí los hay.

Así, se optaría por un modelo ARIMA(3,0,0)(0,1,1).

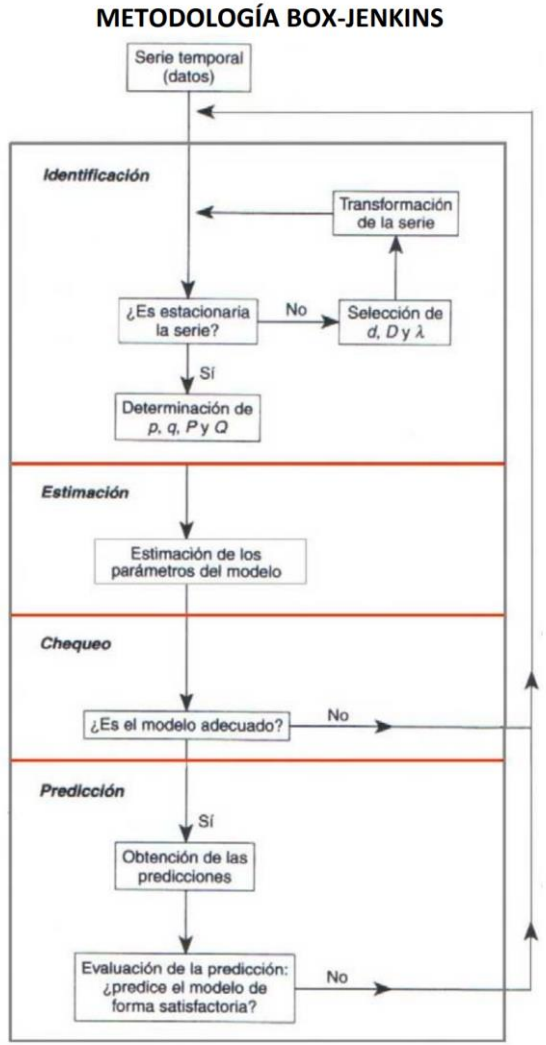
Nótese que es importante probar variaciones, y que no siempre está claro. Existen funciones en Python que optimizan la elección de parámetros a través de minimizar los errores predictivos (más adelante se explicará).



2.6. Metodología Box-Jenkins.

Box y Jenkins propusieron una metodología que permite ajustar los modelos vistos anteriormente a serie temporales reales:

- **Paso 1: Identificación del modelo.** Se utilizarán los datos históricos de la serie para identificar el modelo.
- **Paso 2: Estimación.** Se estiman los datos del modelo a través de los históricos de la serie.
- **Paso 3: Validación.** Se realizan diferentes contrastes para determinar si el modelo es el adecuado. En caso contrario, se vuelve al paso 1.
- **Paso 4: Predicción.** Una vez que se ha construido el modelo, y comprobada su adecuación, se utiliza para realizar predicciones.



2.7. Transformaciones para estabilizar la varianza.

Una **constancia en la media y la varianza son características deseadas** en una serie temporal para facilitar un modelo que ajuste. Si bien se ha visto cómo buscar estacionariedad, si esta viene acompañada de una varianza que aumenta o disminuye con la media (heterocedasticidad de varianza), se puede aplicar **la transformación box-cox**:

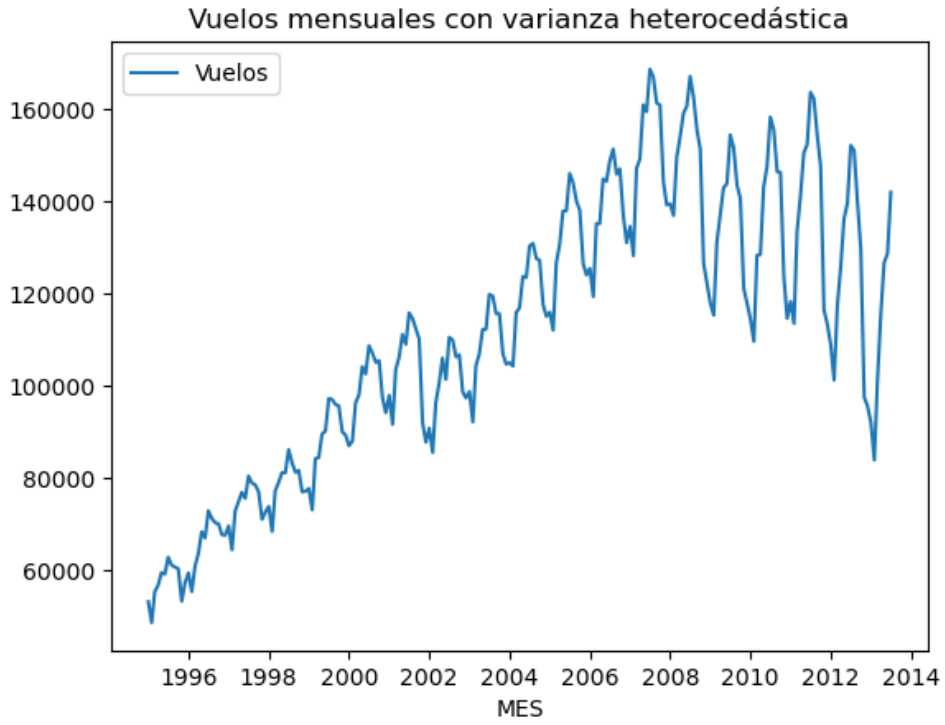
$$y^{(\lambda)} = \begin{cases} \frac{y^{\lambda}-1}{\lambda} & \text{si } \lambda \neq 0 \\ \log y & \text{si } \lambda = 0 \end{cases}$$

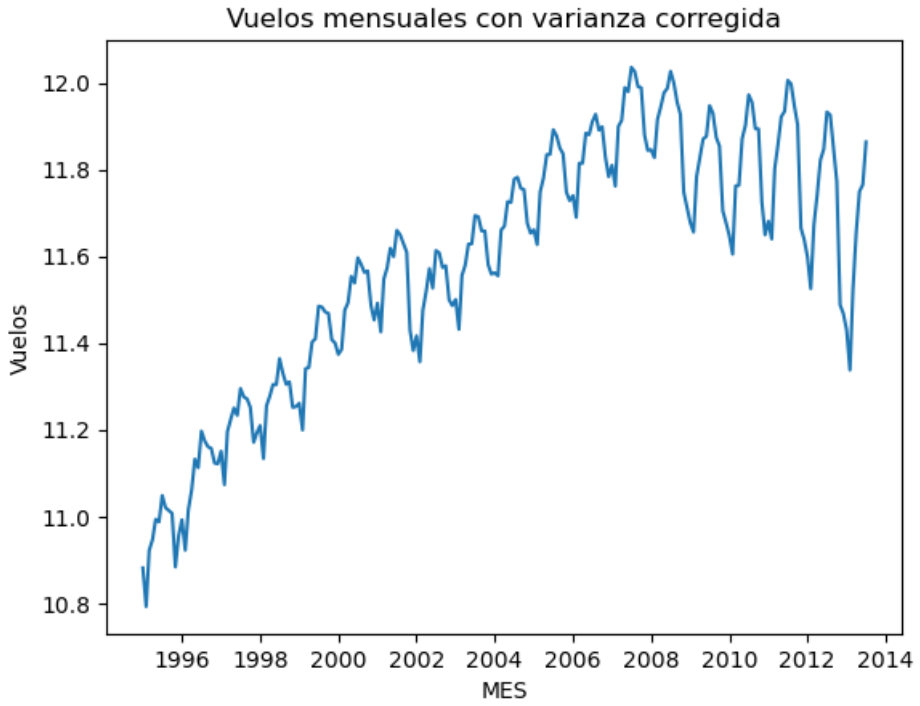
Las transformaciones box-cox permiten transformar la distribución de una variable en una normal.

Así, aplicando el logaritmo (se vio un ejemplo de esto anteriormente), se puede eliminar la heterocedasticidad de varianza, homogeneizando así la serie para un mejor ajuste.

2.7. Transformaciones para estabilizar la varianza.

Ejemplo práctico de estabilización de la varianza: Por ejemplo, la serie número de vuelos España desde Enero de 1995 (conjunto de datos “VUELOS.xlsx”)



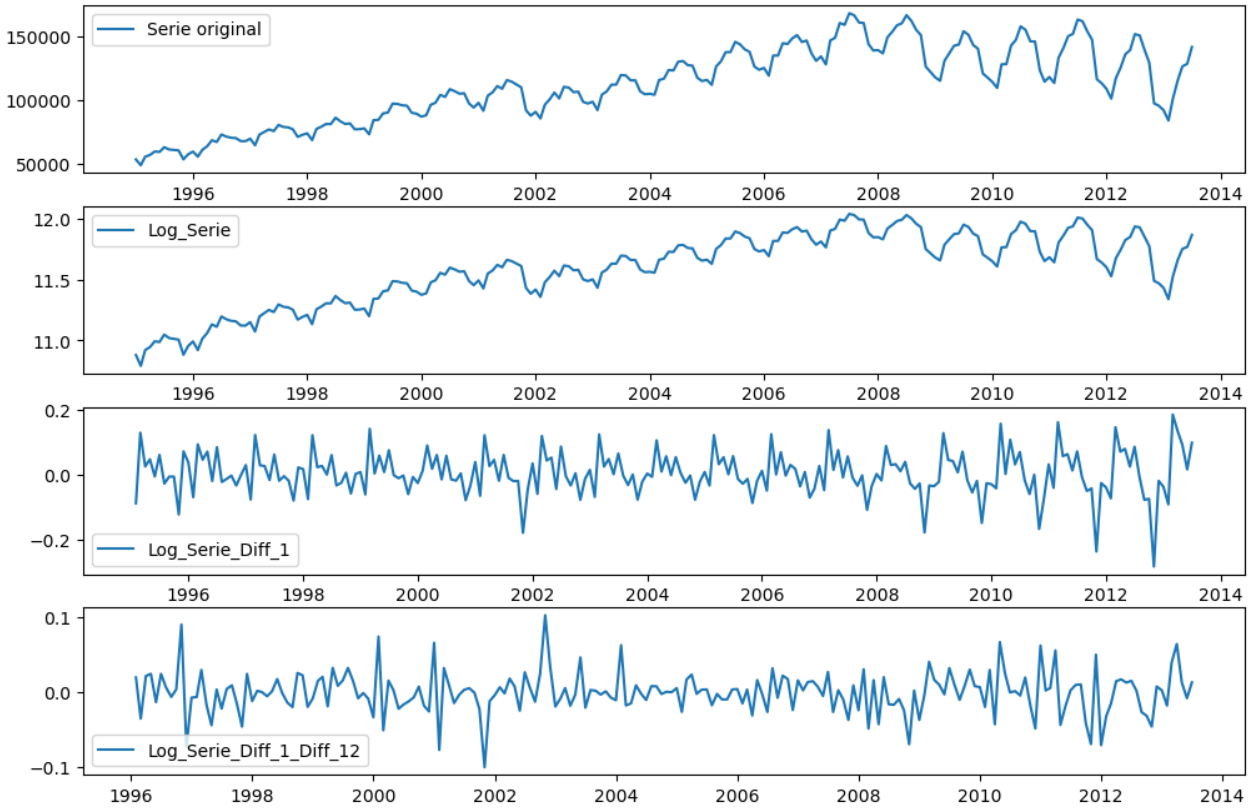


2.7. Transformaciones para estabilizar la varianza.

Ejemplo práctico de estabilización de la varianza: Por ejemplo, la serie número de vuelos España desde Enero de 1995 (conjunto de datos “VUELOS.xlsx”)

Ahora, representaremos la serie, el logaritmo de la serie, el logaritmo diferenciado y sobre ésta última, la diferenciación estacional:

```
plt.figure(figsize=(12, 8))
plt.subplot(4, 1, 1)
plt.plot(vuelos['Vuelos'], label='Serie original')
plt.legend(loc='upper left')
plt.subplot(4, 1, 2)
plt.plot(np.log(vuelos['Vuelos']), label='Log_Serie')
plt.legend(loc='upper left')
plt.subplot(4, 1, 3)
plt.plot(np.log(vuelos['Vuelos']).diff(1), label='Log_Serie_Diff_1')
plt.legend(loc='lower left')
plt.subplot(4, 1, 4)
plt.plot(np.log(vuelos['Vuelos']).diff(1).diff(12),
label='Log_Serie_Diff_1_Diff_12')
plt.legend(loc='lower left')
```



2.8. Resumen de todo y ejemplo completo.

NOTA: el código completo estará en un archivo aparte. La primera vez, intentar redactarlo por nuestra propia cuenta, simplemente siguiendo los pasos, y el código ya mostrado anteriormente.

```
import numpy as np
import pandas as pd
from io import StringIO
import contextlib
import re
import matplotlib.pyplot as plt
plt.style.use('seaborn-v0_8-darkgrid')

# pmdarima
# !pip install pmdarima
from pmdarima import ARIMA
from pmdarima import auto_arima
```

```
# statsmodels
from statsmodels.tsa.statespace.sarimax import
SARIMAX
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.stattools import kpss
from statsmodels.graphics.tsaplots import plot_acf,
plot_pacf
from statsmodels.tsa.seasonal import
seasonal_decompose
```

```
# skforecast
# !pip install skforecast
from skforecast.sarimax import Sarimax
from skforecast.recursive import ForecasterSarimax
from skforecast.model_selection import
backtesting_sarimax
from skforecast.model_selection import
grid_search_sarimax
from sklearn.metrics import mean_absolute_error
import seaborn as sns
import warnings
```

2.8. Resumen de todo y ejemplo completo.

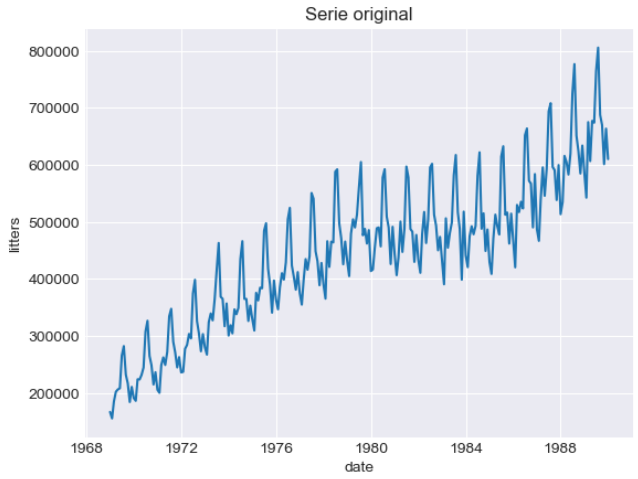
NOTA: el código completo estará en un archivo aparte. La primera vez, intentar redactarlo por nuestra propia cuenta, simplemente siguiendo los pasos, y el código ya mostrado anteriormente.

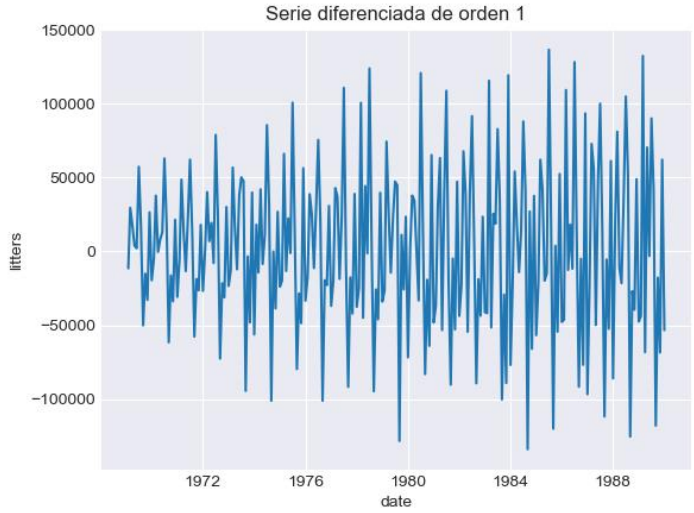
```
# Descarga datos
url = ('https://raw.githubusercontent.com/JoaquinAmatRodrigo/Estadistica-machine-learning-python/'
    'master/data/consumos-combustibles-mensual.csv')
datos_t = pd.read_csv(url, sep=',')
datos_t = datos_t[['Fecha', 'Gasolinas']]
datos_t = datos_t.rename(columns={'Fecha':'date', 'Gasolinas':'litters'})
datos_t['date'] = pd.to_datetime(datos_t['date'], format='%Y-%m-%d')
datos_t = datos_t.set_index('date')
datos = datos_t.loc[:'1980-01-01 00:00:00']
datos = datos.asfreq('MS')
datos = datos['litters']
datos.head()
```

2.8. Resumen de todo y ejemplo completo.

Análisis exploratorio.

Se requiere observar estacionariedad, eliminar la heterocedasticidad – si procede – en la varianza, análisis de autocorrelación y descomposición estacional.





No parece haber problemas de heterocedasticidad en la varianza. Aunque con diferenciación de orden 1 se observa la estacionariedad, existen pruebas de contraste de la misma.

2.8. Resumen de todo y ejemplo completo.

Paso 1: análisis exploratorio. Estacionariedad, autocorrelación y descomposición estacional.

La prueba Dickey-Fuller (adfuller() de statsmodels) parte de la hipótesis nula de que la serie temporal es no estacionaria. Así, se busca rechazar esta H_0 y aceptar la hipótesis alternativa de estacionariedad, con $p < 0{,}05$. adfuller()[0] ofrece su valor, adfuller()[1] el nivel de significación.

También existe la Prueba Kwiatkowski-Phillips-Schmidt-Shin (KPSS), kpps() que se puede aplicar de forma paralela, cuya hipótesis nula es que la serie es estacionaria, por lo que se buscan valores > 0.05. kpps()[0] ofrece su valor, kpps()[1] el nivel de significación.

```
Test estacionariedad serie original
ADF Statistic: -1.1596066639173295, p-value: 0.6907262160487554
KPSS Statistic: 1.8725434213925662, p-value: 0.01

Test estacionariedad serie diferenciada de orden 1
ADF Statistic: -3.641727690032322, p-value: 0.005011605002137 26
KPSS Statistic: 0.05268059379842924, p-value: 0.1
```

Conocer que la serie diferenciada en orden 1 es estacionaria, nos aporta el primer parámetro del modelo ARIMA, d = 1.

2.8. Resumen de todo y ejemplo completo.

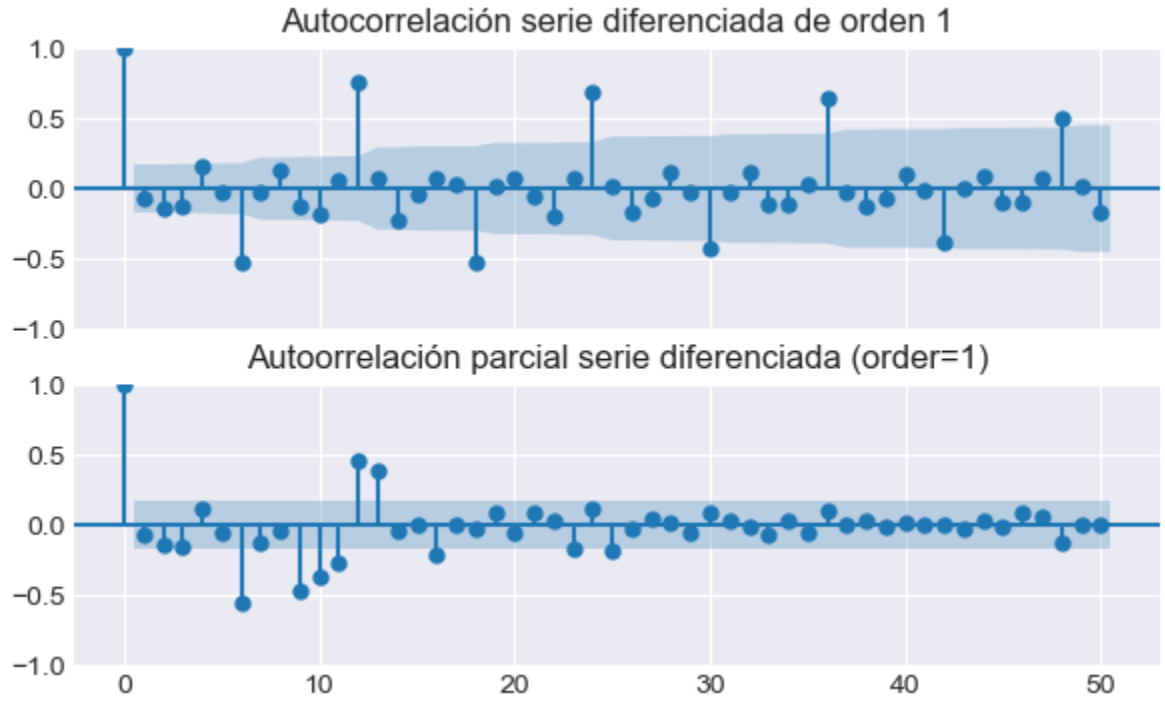
Paso 2: estimación del modelo.

Con ACF y PACF, se podrán determinar los parámetros p (PACF) y q (ACF):

La función de autocorrelación parcial sugiere un valor p = 0, así como la función de autocorrelación q = 0.

Dado que encontrar el modelo óptimo no siempre es sencillo, en ocasiones es recomendable asignar parámetros p y q > 0 para captar patrones sutiles o comportamientos no lineales.

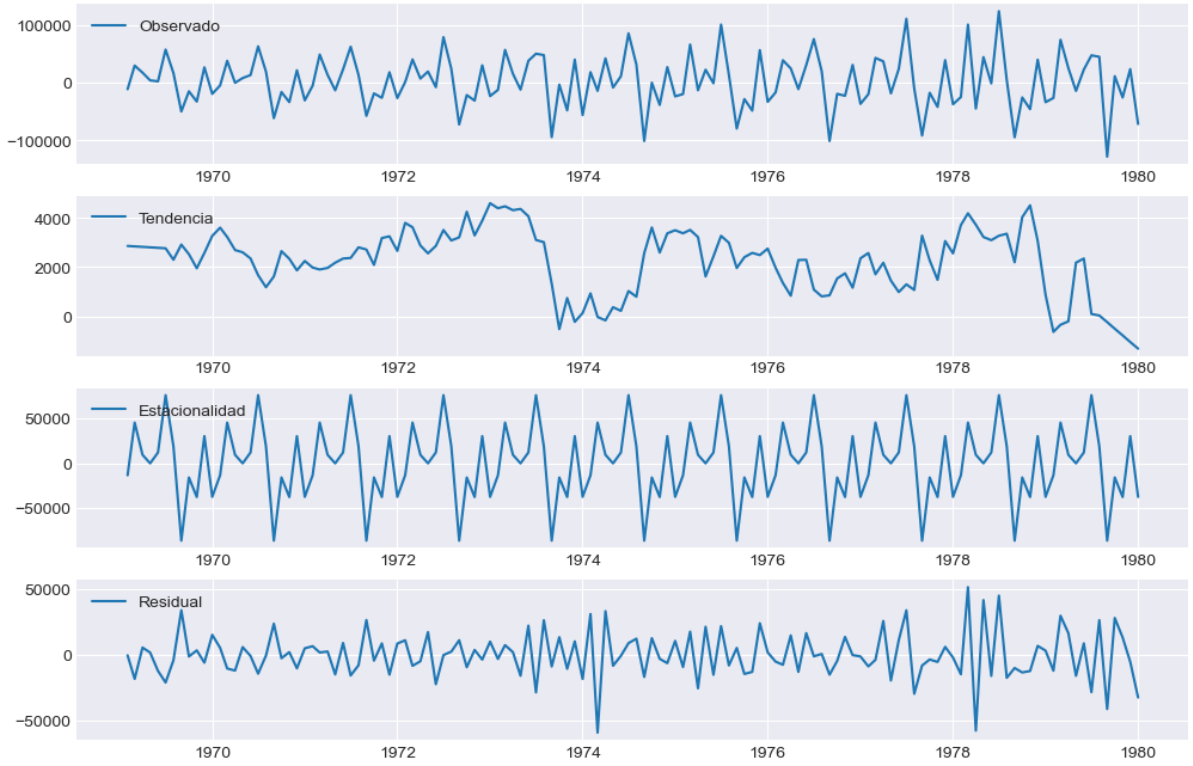
El primer modelo que se probará será p = 0 y q = 1, integrando la naturaleza autorregresiva al modelo



2.8. Resumen de todo y ejemplo completo.

Paso 2: estimación del modelo.

Integrar la descomposición con el análisis de la (ACF) y la (PACF), se logra una descripción integral que ayuda a comprender la estructura subyacente de los datos y a determinar los valores óptimos de los parámetros ARIMA.



Como se puede observar a primera vista, se encuentra un patrón recurrente en la estacionalidad cada 12 meses (anual), parámetro D del modelo.

A continuación, puede ser interesante observar ACF y PACF sobre una serie diferenciada de orden 12 para observar los posibles parámetros P y Q.

2.8. Resumen de todo y ejemplo completo.

Paso 2: estimación del modelo.

En primer lugar, se contrasta la estacionariedad de la serie diferenciada de orden 12:

```
Test estacionariedad serie de orden 12

Test estacionariedad serie diferenciada de orden 12
ADF Statistic: -4.387457230769959, p-value: 0.0003123773271126894
KPSS Statistic: 0.06291573421251051, p-value: 0.1
```

A continuación, se procede a la representación de las funciones ACF y PACF

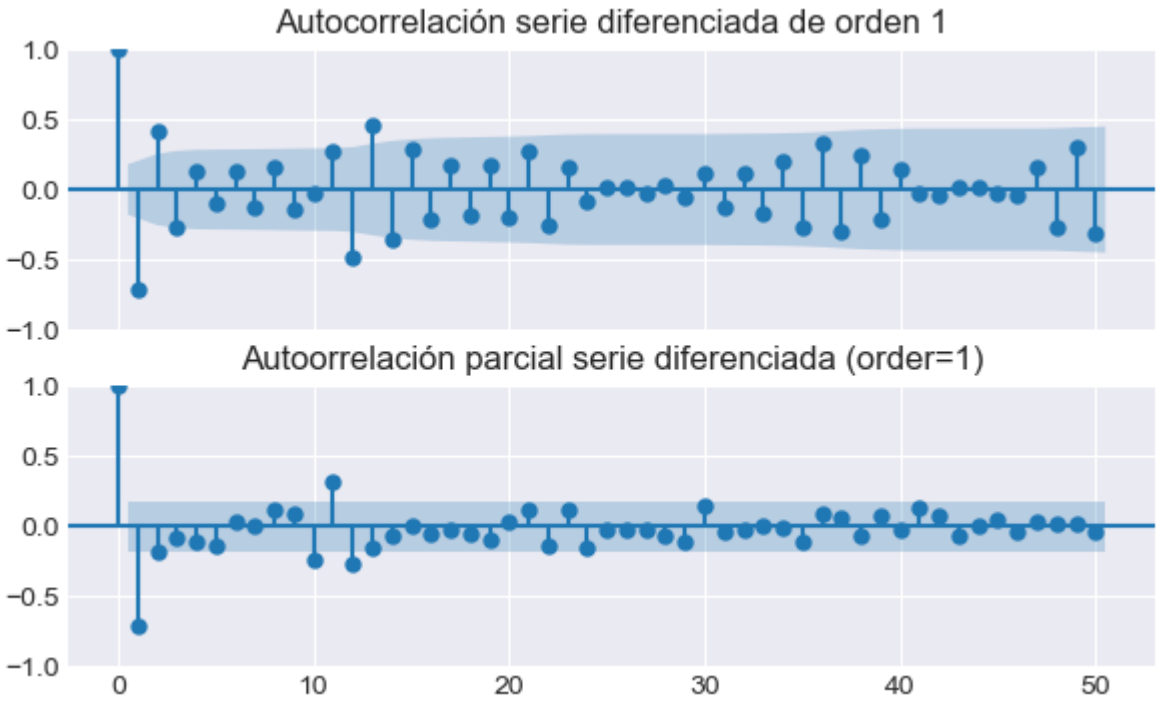
2.8. Resumen de todo y ejemplo completo.

Paso 2: estimación del modelo.

Las funciones parecen indicar un valor Q = 3 y un valor P = 1.

Así, parece que un buen modelo para empezar
es un modelo $ARIMA(0,1,1)(1,1,3)_{12}$

Es conveniente probar con varios modelos y ver cuál
Se ajusta mejor, dado que puede haber tendencias que
No se aprecien a simple vista



2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

```
warnings.filterwarnings("ignore", category=UserWarning, message='Non-invertible|Non-stationary')
modelo = SARIMAX(endog = datos, order = (0, 1, 1), seasonal_order = (1, 1, 3, 12))
modelo_res = modelo.fit(disp=0)
warnings.filterwarnings("default")
modelo_res.summary()
```

SARIMAX Results

Dep. Variable:	litters	**No. Observations:**	133
Model:	SARIMAX(0, 1, 1)x(1, 1, [1, 2, 3], 12)	**Log Likelihood**	-1358.014
Date:	Wed, 13 Dec 2023	**AIC**	2728.029
Time:	18:57:25	**BIC**	2744.754
Sample:	01-01-1969	**HQIC**	2734.821
	- 01-01-1980		
Covariance Type:	opg		

2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

El criterio de información de Akaike (AIC) y el criterio de información bayesiano (BIC), medidas de bondad de ajuste, están basados en el logaritmo de la función de verosimilitud utilizada para calcular los estimadores de la serie bajo el supuesto de Normalidad de los residuos.

$$AIC = -2\ln(L) + 2k$$

$$BIC = -2\ln(L) + \ln(n)k$$

Donde *L* es la función de verosimilitud de la serie, *k* el número de parámetros y *n* el número de residuos calculados.

AIC y BIC introducen un término de penalización para el número de parámetros en el modelo, pero la penalización es mayor para BIC.

No. Observations:	133
Log Likelihood	-1358.014
AIC	2728.029
BIC	2744.754
HQIC	2734.821

A la hora de comparar modelos, buscaremos los valores AIC y BIC más bajos.

2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

Se pueden observar los coeficientes del modelo explicado anteriormente, así como su significación asociada. En este caso, se observa que la elección de parámetros no parece haber sido la más acertada dada la significación encontrada.

Recuérdese que una p > 0,05 no permite concluir que el coeficiente sea distinto de 0.

El contraste Ljung Box nos permite comprobar si los residuos son incorrelados Siendo esta la hipótesis nula asociada a este contraste

	coef	std err	z	P>|z|	[0.025	0.975]
ma.L1	-0.4226	0.050	-8.448	0.000	-0.521	-0.325
ar.S.L12	-0.1853	0.578	-0.320	0.749	-1.319	0.948
ma.S.L12	-0.1996	0.595	-0.335	0.737	-1.367	0.967
ma.S.L24	-0.0731	0.228	-0.320	0.749	-0.520	0.374
ma.S.L36	0.1415	0.087	1.621	0.105	-0.030	0.313
sigma2	4.578e+08	1.09e-09	4.2e+17	0.000	4.58e+08	4.58e+08

Ljung-Box (L1) (Q):	9.30	**Jarque-Bera (JB):**	14.73
Prob(Q):	0.00	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.33	**Skew:**	-0.48
Prob(H) (two-sided):	0.38	**Kurtosis:**	4.42

2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

```
# Predicción
predicciones_statsmodels = modelo_res.get_forecast(steps=61).predicted_mean
predicciones_statsmodels.name = 'predicciones_statsmodels’

# se comprueba lo predicho con los valores reales
datos_futuro = datos_t.loc['1980-01-01 00:00:00':'1985-01-01 00:00:00']
datos_futuro = datos_futuro['litters']
```

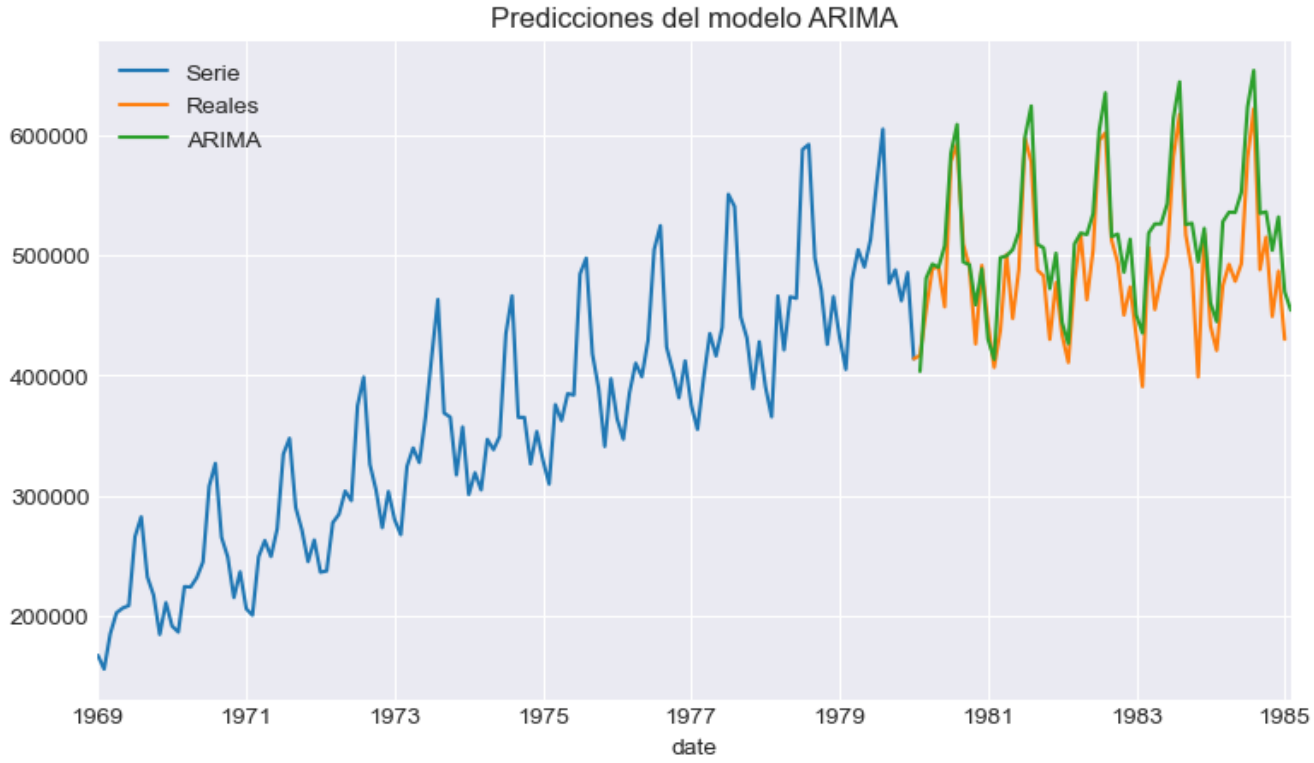
2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

```
# Plot predictions
fig, ax = plt.subplots(figsize=(9, 5))
datos.plot(ax=ax, label='Serie')
datos_futuro.plot(ax=ax, label='Reales')
predicciones_statsmodels.plot(ax=ax, label='ARIMA')
ax.set_title('Predicciones del modelo ARIMA')
ax.legend();
```

A pesar de que aparentemente el modelo se ajusta, se ha visto que no es el más adecuado.

Una opción es la función “autoarima”



2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

Cuando ajustamos un modelo calcularemos las medidas de adecuación de su ajuste basadas en los residuos del modelo para las T observaciones de que disponemos.

$$\varepsilon_t = X_t - \hat{X}_t$$

El valor total de estos residuos se resume en diferentes estadísticos que presentamos a continuación y cuya interpretación en general es sencilla: buscamos el menor error total posible medido de diferentes formas.

El Error Absoluto Medio: $$MAE = \sum_{t=1}^{T} \frac{|e_t|}{T}$$

La desviación estándar del error también llamada raíz de la media de los errores al cuadrado.

$$RMSE = \sqrt{\sum_{t=1}^{T} \frac{\hat{e}_t^2}{T-k}}$$

2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

```
# Mean Squared Error (MSE)
mse = mean_squared_error(datos_futuro,
predicciones_statsmodels)
print(f'Mean Squared Error (MSE): {mse}')

# Root Mean Squared Error (RMSE)
rmse = np.sqrt(mse)
print(f'Root Mean Squared Error (RMSE): {rmse}')

# Mean Absolute Error (MAE)
mae = mean_absolute_error(datos_futuro,
predicciones_statsmodels)
print(f'Mean Absolute Error (MAE): {mae}')
```

```
Mean Squared Error (MSE): 4219458717.059663
Root Mean Squared Error (RMSE): 64957.360761192125
Mean Absolute Error (MAE): 52742.63993757908
```

2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

```
modelo = auto_arima(
        y               = datos,
        start_p         = 0,
        start_q         = 0,
        max_p           = 3,
        max_q           = 3,
        seasonal        = True,
        test            = 'adf',
        m               = 12, # periodicidad de la estacionalidad
        d               = None, # El algoritmo determina 'd'
        D               = None, # El algoritmo determina 'D'
        trace           = True,
        error_action    = 'ignore',
        suppress_warnings = True,
        stepwise        = True)
```

```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(1,1,1)[12]             : AIC=2761.331, Time=0.12 sec
 ARIMA(0,1,0)(0,1,0)[12]             : AIC=2787.327, Time=0.02 sec
 ARIMA(1,1,0)(1,1,0)[12]             : AIC=2721.601, Time=0.12 sec
 ARIMA(0,1,1)(0,1,1)[12]             : AIC=2726.070, Time=0.14 sec
 ARIMA(1,1,0)(0,1,0)[12]             : AIC=2735.413, Time=0.03 sec
 ARIMA(1,1,0)(2,1,0)[12]             : AIC=2715.646, Time=0.33 sec
 ARIMA(1,1,0)(2,1,1)[12]             : AIC=2716.922, Time=0.53 sec
 ARIMA(1,1,0)(1,1,1)[12]             : AIC=2719.493, Time=0.19 sec
 ARIMA(0,1,0)(2,1,0)[12]             : AIC=2753.562, Time=0.25 sec
 ARIMA(2,1,0)(2,1,0)[12]             : AIC=2718.130, Time=0.52 sec
 ARIMA(1,1,1)(2,1,0)[12]             : AIC=2718.076, Time=0.42 sec
 ARIMA(0,1,1)(2,1,0)[12]             : AIC=2723.616, Time=0.32 sec
 ARIMA(2,1,1)(2,1,0)[12]             : AIC=2719.094, Time=1.37 sec
 ARIMA(1,1,0)(2,1,0)[12] intercept   : AIC=2717.278, Time=0.36 sec

Best model:  ARIMA(1,1,0)(2,1,0)[12]
Total fit time: 4.749 seconds
```

2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

Al ajustar el nuevo modelo:

Ojo, porque encontramos residuos correlados Ljung-Box = 4.92, p < 0.05. Esto indica que hay patrones sistemáticos no capturados por el modelo.

Existen métodos de entrenamiento más sofisticados (entrenamiento, validación y test) que mejoran la calidad de la creación de modelos, mejorando significativamente la calidad de la predicción.

Esto se verá más adelante en el master. No os olvidéis de series temporales cuando lo veáis, para poder aplicarlo!

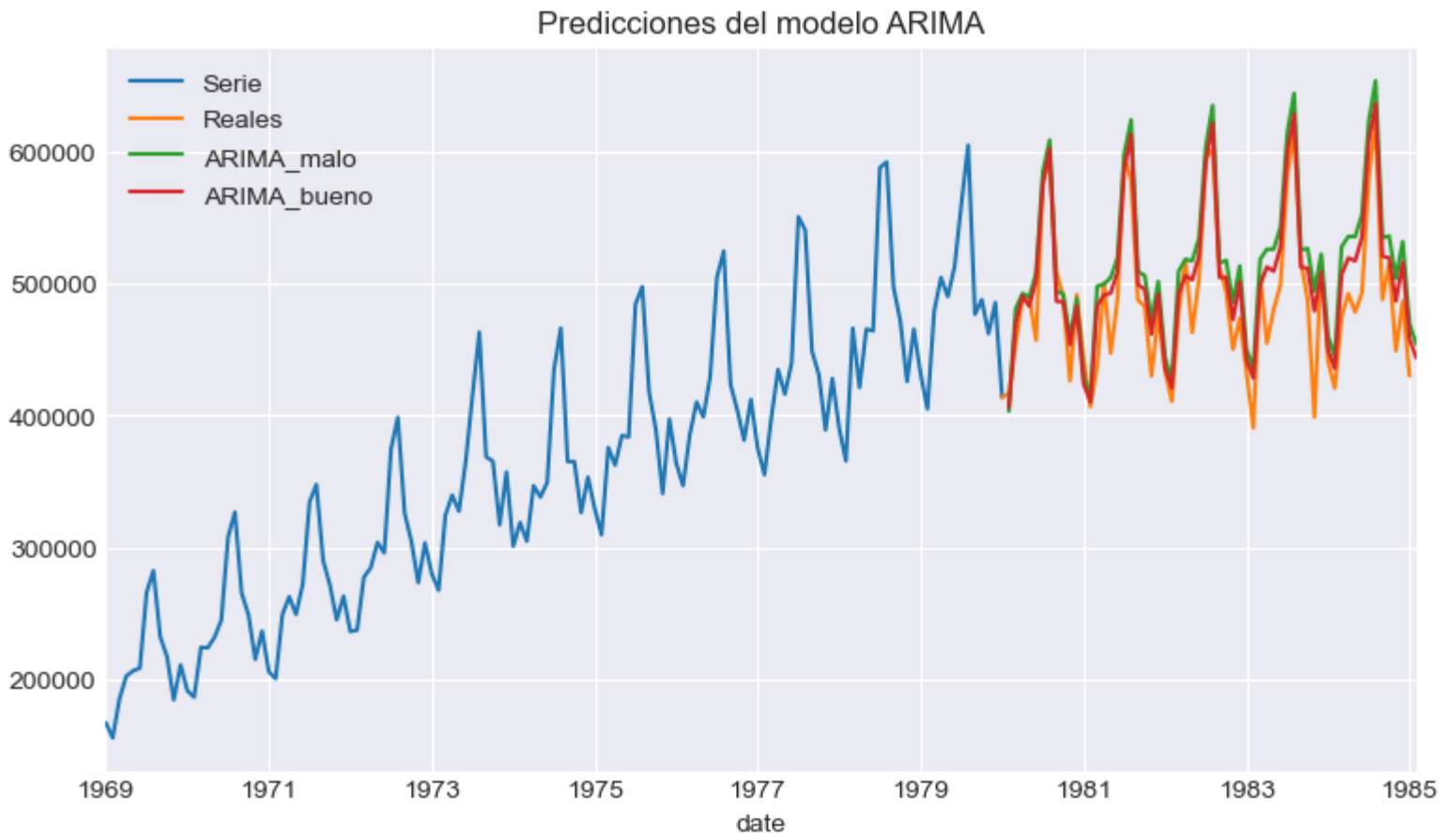
	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.4822	0.037	-13.034	0.000	-0.555	-0.410
ar.S.L12	-0.3631	0.043	-8.488	0.000	-0.447	-0.279
ar.S.L24	-0.1989	0.042	-4.705	0.000	-0.282	-0.116
sigma2	3.428e+08	2.33e-11	1.47e+19	0.000	3.43e+08	3.43e+08

Ljung-Box (L1) (Q):	4.92	**Jarque-Bera (JB):**	10.66
Prob(Q):	0.03	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.16	**Skew:**	-0.38
Prob(H) (two-sided):	0.64	**Kurtosis:**	4.24

2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

Al ajustar el nuevo modelo:



2.8. Resumen de todo y ejemplo completo.

Paso 3: validación del modelo.

Al comparar el MSE, MEA y RMSE entre modelos, se observa lo siguiente:

Primer modelo:

```
Mean Squared Error (MSE): 4219458717.059663
Root Mean Squared Error (RMSE): 64957.360761192125
Mean Absolute Error (MAE): 52742.63993757908
```

Segundo modelo:

```
Mean Squared Error (MSE): 3494914516.4535637
Root Mean Squared Error (RMSE): 59117.802026577105
Mean Absolute Error (MAE): 48320.11325530279
```

Concluyendo, como mejor opción, el modelo $ARIMA(1,1,0)(2,1,0)_{12}$ sobre $ARIMA(0,1,1)(1,1,3)_{12}$.

3. Bibliografía.

- Librería statsmodels de Python
- Análisis de series temporales. Daniel Peña. ISBN:978-84-206-6945-8
- https://cienciadedatos.net/documentos/py51-modelos-arima-sarimax-python#Estacionariedad