Análisis y Predicción de Series Temporales

Minería de datos y Modelización Predictiva

Bartolomé Mestre Fons

Miércoles, 21 de Enero de 2025

Minería de datos y Modelización Predictiva
2024-2025
UCM - Universidad Complutense de Madrid

Índice

1. Objetivos

En el siguiente documento se pretende realizar un análisis descriptivo y predictivo de series temporales sobre la temperatura media mensual en el territorio español de la ciudad de Barcelona desde Febrero de 2010 a Diciembre del 2024. Tras descomponer los datos en función de sus componentes estacional, tendencia y residual, se dividirá el conjunto en una parte dedicada al entreno del modelo y otra para su testeo. A continuación, se utilizarán técnicas de suavizado para ajustar el modelo óptimo y predecir las temperaturas medias para aquellos datos dedicados al testeo. Seguidamente, se utilizarán técnicas de análisis de series temporales ARIMA (manual y automática) para comparar los resultados obtenidos con los suavizados y escoger el modelo ganador. Con este último, se predecirá la temperatura media para los siguientes 12 meses. Finalmente, se comparará el rendimiento de los modelos estadísticos y su precisión.

2. Introducción

La serie de datos a analizar corresponde a los 180 registros de la temperatura media mensual de Barcelona desde Febrero de 2010 a Diciembre de 2024. El conjunto de datos se ha descargado de la fuente citada en [1], y se compone de un total de 2820 observaciones donde se indica el año de la observación ("Any") y la temperatura media por mes en las siguientes columnas ("Temp_Mitjana_Gener", "Temp_Mitjana_Febrer", "Temp_Mitjana_Marc", etc.). Solo se pretende analizar los últimos 180 registros de la temperatura media, por lo que primero se importan las librerías necesarias y el documento *csv* con el que trabajar:

```
# Se importan aquellas librerías y módulos a utilizar ahora y a lo largo del documento.
import pandas as pd
import matplotlib.pyplot as plt
import pmdarima as pm
import statsmodels.api as sm
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.api import ExponentialSmoothing
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import adfuller
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

df = pd.read_csv('download.csv')
```

Se modifica el documento para que las observaciones se distribuyan verticalmente en función de la fecha:

```
df_melted = df.melt(id_vars=['Any'],
                    var_name='Mes',
                    value_name='Temperatura')

df_melted['Mes'] = df_melted['Mes'].str.replace('Temp_Mitjana_', '')
df_melted['Mes'] = df_melted['Mes'].map(month_translation)

month_order = {
    'Gener': 1, 'Febrer': 2, 'Marc': 3, 'Abril': 4, 'Maig': 5, 'Juny': 6,
    'Juliol': 7, 'Agost': 8, 'Setembre': 9, 'Octubre': 10, 'Novembre': 11, 'Desembre': 12
}
df_melted['Numero_Mes'] = df_melted['Mes'].map(month_order)

df_melted['dt'] = pd.to_datetime(
   df_melted[['Any', 'Numero_Mes']].rename(
columns={'Any': 'year', 'Numero_Mes': 'month'}
).assign(day=1)
)

df_melted = df_melted[['dt', 'Temperatura']]
df_melted['dt'] = df_melted['dt'].dt.to_period('M')
df_melted = df_melted.sort_values(by='dt').reset_index(drop=True)
```

Tras preparar los datos, se conservan los últimos 180 registros y se guardan en un archivo *csv* externo con el que trabajar de ahora en adelante:

```
df = df_melted[-180:]
df = df.reset_index(drop = True)
df.to_csv('Barcelona_avg_temp.csv')
```

Junto al presente documento se adjunta el archivo *csv* obtenido en la anterior operación. Se toma la variable "dt" como índice:

```
df = df.set_index('dt')
```

Se representa a continuación gráficamente la serie temporal:

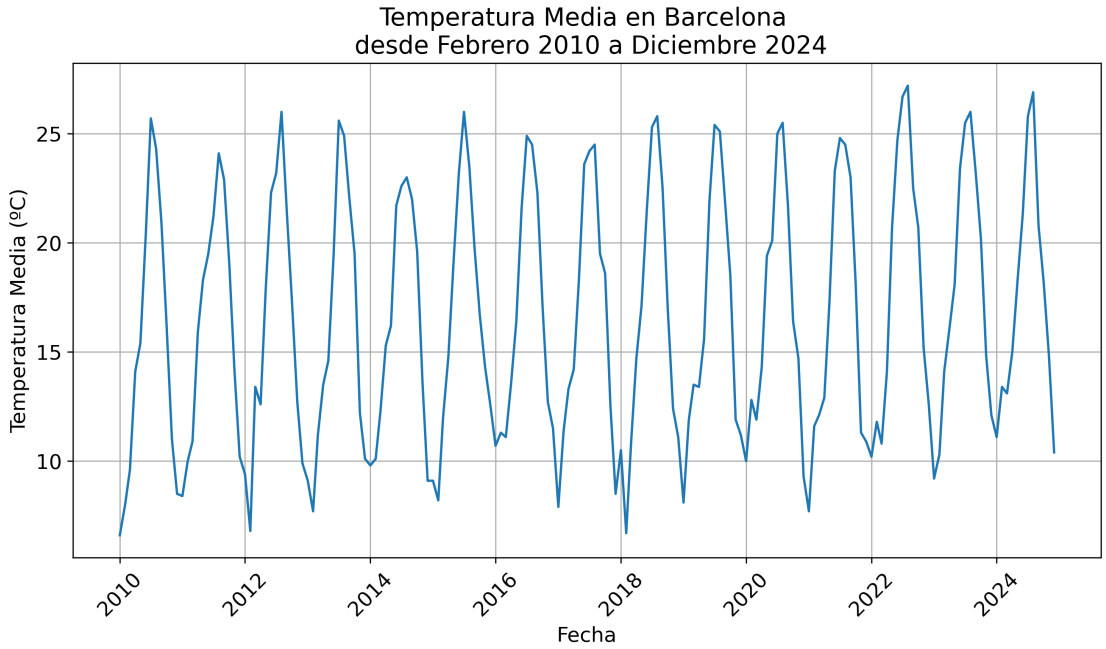

Figura 1: Temperatura Media en Barcelona desde Febrero 2010 a Diciembre 2024.

Se identifica un patrón cíclico que corresponde a la anualidad de los datos, empezando por los meses de Enero y terminando por los meses de Diciembre, por lo que el periodo de dicho ciclo es de 12 observaciones. Los máximos se localizan en los meses de verano y los mínimos en los de inverno. Así, se desglosan las componentes estacional, tendencia y residual con el siguiente código, incluyendo su respectiva representación gráfica:

```
multiplicative_decomposition = seasonal_decompose(df, model = 'multiplicative', period = 12)
plt.rc('figure', figsize = (12,8))
plt.rc('font', size = 13)
plt.rc('title')
fig = additive_decomposition.plot()
fig.suptitle('Tendencia, Componente Estacional y Residuales de \n
 la Temperatura Media desde Febrero 2010 a Diciembre 2024')
plt.subplots_adjust(top=0.9)
plt.show()
```

Se obtiene lo siguiente:

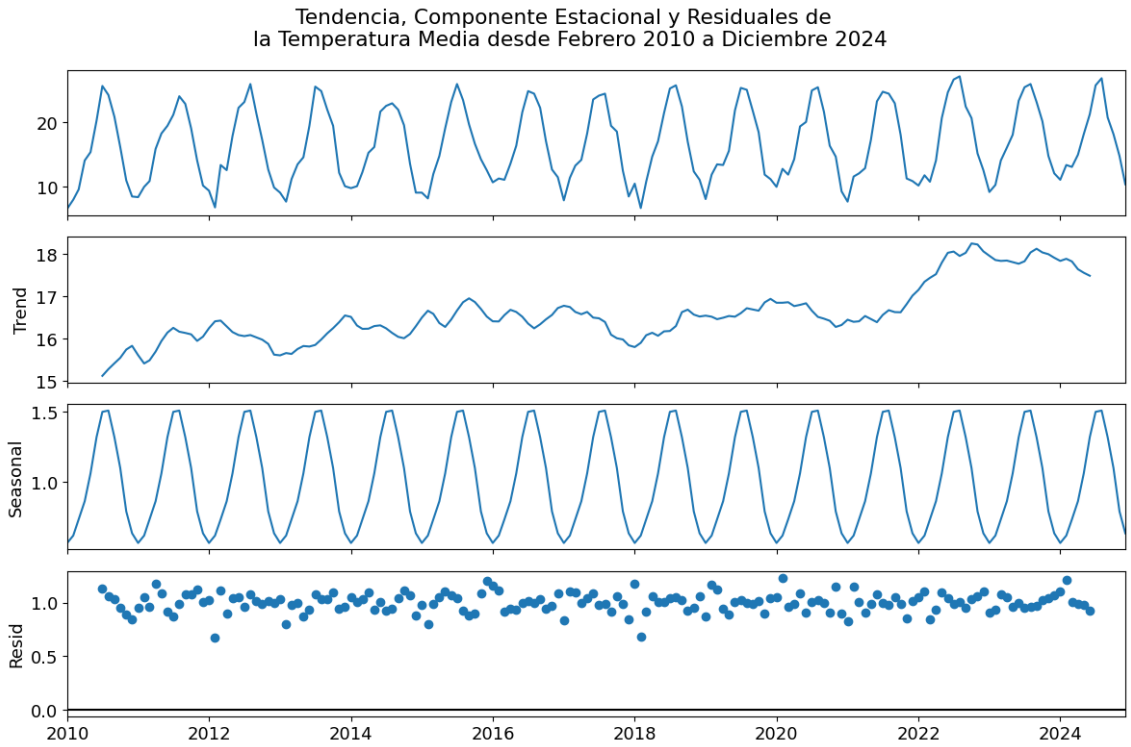

Figura 2: Tendencia, Componente Estacional y Residuales de la Temperatura Media desde Febrero 2010 a Diciembre 2024.

En el primer gráfico se muestra la serie temporal con la temperatura media en el eje vertical y la fecha en el horizontal. En el segundo, la componente de la tendencia, que sigue un patrón irregular aunque creciente. En el tercero, se ilustra la componente estacional, que sigue un patrón sinusoidal al identificar una ciclicidad anual sobre los datos. Finlamente, en el último se muestra la distribución de los residuales (la componente aleatoria de los datos), que siguen una distribución aleatoria alrededor de una media de 1 (con un método aditivo la media habría sido 0 al tener que en el modelo aditivo se modela la predicción como Tendencia + Factor Estacional + Residuos, mientras que con el método multiplicativo se calcula con Tendencia · Factor Estacional · Residuos). De las anteriores proposiciones se extrae que existe una fuerte componente estacional en los datos, por lo que se requerirá de una futura diferenciación de periodo 12 en el método ARIMA. Además, la creciente tendencia apunta a un incremento consistente generalizado de las temperaturas. Véase que se ha escogido un análisis estacional multiplicativo, tal que las contribuciones estacionales, tendencia y residuales son multiplicadas para reproducir el dato real. Se ha elegido esta opción frente a la estacionalidad aditiva debido a que no se disponen de temperaturas mensulaes medias iguales a 0º, por lo que la primera opción rendirá mejor en un principio.

Como añadido, se adjuntan los coeficientes de estacionalidad para los 12 primeros registros:

Fecha	Coeficiente de Estacionalidad
2010-01-01	0.564536
2010-02-01	0.617197
2010-03-01	0.732488
2010-04-01	0.861510
2010-05-01	1.060309
2010-06-01	1.318561
2010-07-01	1.498655
2010-08-01	1.507664
2010-09-01	1.317867
2010-10-01	1.098529
2010-11-01	0.789259
2010-12-01	0.633424

Cuadro 1: Factores de Estacionalidad para los primeros 12 registros.

Los coeficientes anteriores se situan alrededor de 1 y muestran cuán alejado se halla cada registro de la media. Así como ejemplo, la primera lectura tan solo representaría el 56.5 % de la media aproximadamente, mientras que la última el 63.3 %.

3. Modelización

3.1. División de los datos en *Train/ Test*

Se dividirá el dataset anterior en dos subdivisiones, dedicada una al entreno del modelo y la otra al testeo de su eficiencia. Esta última permite obtener los errores estimados del modelo respecto a los datos reales. Es importante aquí mantener el orden temporal debido a la dependencia temporal de la serie. Así, se escoge dedicar 144 lecturas para la subdivisión *Train* y 36 para la *Test* (si la extensión de los datos de prueba es demasiado pequeña, las métricas de precisión pueden verse afectadas por la variabilidad de unos pocos ejemplos, por lo que se toma el 20 % de datos como datos de testeo). Se acomete dicha tarea con el siguiente código:

```
x_train = df[:-36]
x_test = df[-36:]
```

Se representan dichas subdivisiones de los datos en un gráfico:

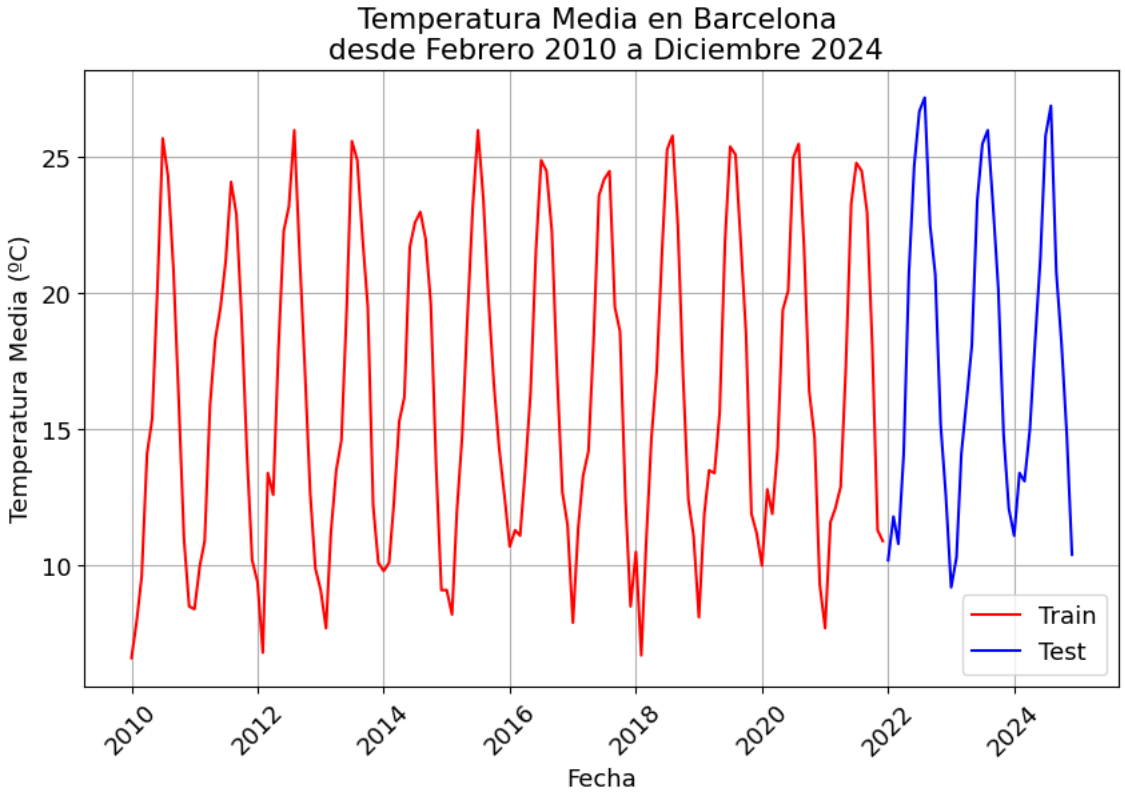

Figura 3: Subdivisiones *Train* y *Test* de la Temperatura Media desde Febrero 2010 a Diciembre 2024.

3.2. Método de Suavizado

Se escoge aplicar el método de suavizado exponencial sobre los datos de entreno, asociando pesos que decrecen con el tiempo exponencialmente. Así, se eliminan las fluctuaciones aleatorias y permanece la componente de tendencia-estacionalidad. Se implementa dicho algoritmo utilizando el *método de Holt-Winters*. A diferencia de otros métodos como el de *alisado simple* (orientado a series cuya tendencia es nula o fluctua mínimamente) o *Holt* (incluye una dependencia lineal en la tendencia), el de *Holt-Winters* recoge el comportamiento estacional de la serie incluyéndolo mediante un coeficiente que multiplica a la componente de tendencia. Se implementa dicha función con el siguiente código:

```
model1 = ExponentialSmoothing(x_train, seasonal_periods = 12, trend = 'mul',
                              seasonal = 'mul', initialization_method = 'estimated').fit()
fcast1 = model1.forecast(36)
```

En la rutina anterior se ha escogido una longitud del periodo de 12 (ciclo de 12 meses), así como añadir los efectos de tendencia y estacionalidad multiplicativamente. Por otra parte, se ha creado la variable "fcast1", que recoge la predicción de los siguientes 36 registros de temperatura media a partir de la última observación de los datos de entreno. En la siguiente tabla se recogen los parámetros del modelo, obtenidos con model1.params_formatted:

Nombre del parámetro	Abreviatura	Valor
Smoothing Level	α	$1{,}75 \cdot 10^{-8}$
Smoothing Trend	β	$1{,}75 \cdot 10^{-8}$
Smoothing Seasonal	γ	$7{,}06 \cdot 10^{-18}$
Initial Level	l_0	12,72
Initial Trend	b_0	1,00
Initial Season 0	s_0	0,68
Initial Season 1	s_1	0,74
Initial Season 2	s_2	0,90
Initial Season 3	s_3	0,11
Initial Season 4	s_4	1,33
Initial Season 5	s_5	1,62
Initial Season 6	s_6	1,86
Initial Season 7	s_7	1,87
Initial Season 8	s_8	1,65
Initial Season 9	s_9	1,36
Initial Season 10	s_{10}	0,97
Initial Season 11	s_{11}	0,78

Cuadro 2: Parámetros Optimizados del Método de Suavizado Exponencial.

Para entender mejor los coeficientes anteriores, se grafican el Nivel, Tendencia y Estacionalidad del modelo suavizado (obtenidos con model1.level, model1.trend y model1.season, respectivamente):

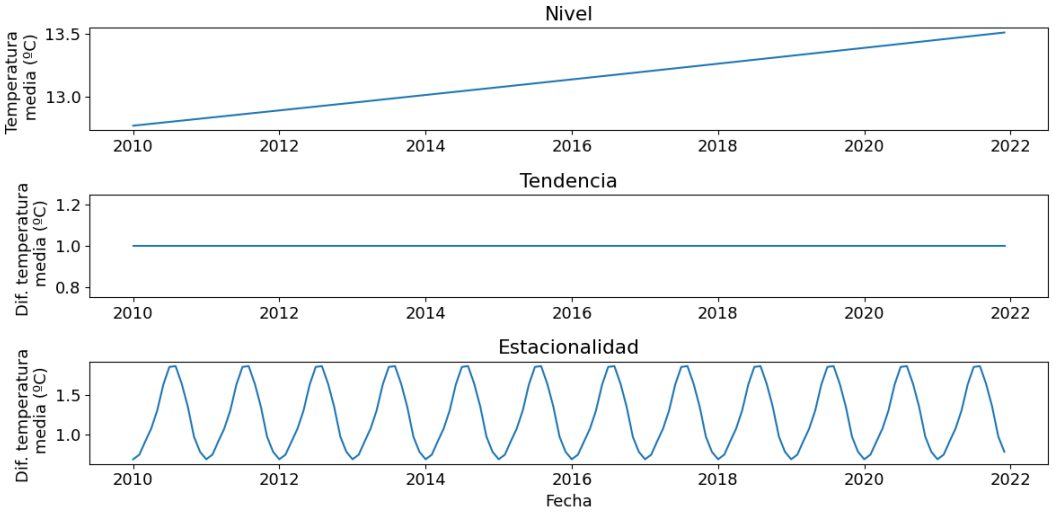

Figura 4: Componentes de Nivel, Tendencia y Estacionalidad del Modelo Suavizado.

El Nivel es la componente no influenciada por las fluctuaciones estacionales o tendencias. Para esta componente, se observa un crecimiento suave y constante general a lo largo del tiempo (a causa de los parámetros "smoothing_level" (que indica el peso a dar a los datos recientes) e "initial_level" (punto de partida del Nivel)), lo que refleja un aumento gradual de las temperaturas promedio en territorio barcelonés. Por otra parte, la Tendencia es casi constante y con pendiente mínima (por los parámetros "smoothing_trend" (cuanto peso se le da a los datos recientes para la tendencia) e "initial_trend" (punto de partida de la tendencia)), mientras que en la parte Estacional se aprecia una ciclicidad propia de la anualidad de los datos (determinada por los parámetros "smoothing_seasonal" (indica como se actualiza la componente estacional en función de los nuevos datos) e "initial_seasons" (s_0, s_1, etc.) (definen el patrón cíclico de 12 meses)), inalterada a lo largo del tiempo. Con esto, se concluye que el modelo asocia mayores pesos a los valores iniciales tanto para el Nivel como para la Tendencia y Estacionalidad, con mínimos ajustes sobre ellos debido a los bajos coeficientes "smoothing" . Esto explica la ausencia de ruido o inestabilidades notables en los gráficos anteriores.

A continuación, se grafican los resultados del Modelo Suavizado y sus predicciones para la subdivisión *Test* del dataset, donde la parte suavizada de la subdivisión *Train* se ha obtenido con model1.fittedvalues:

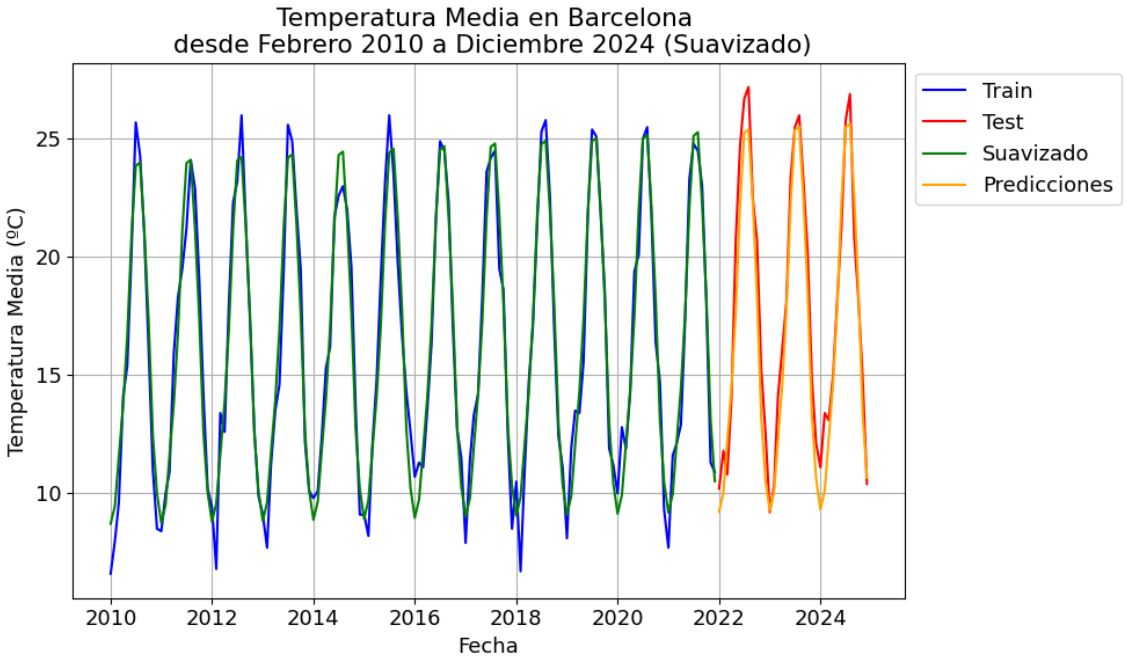

Figura 5: Comparativa entre las Temperaturas Medias reales y las ofrecidas por el Modelo Suavizado para las subdivisiones *Train* y Test.

Se observa para la subdivisión *Train* que las temperaturas medias coinciden en buena parte con las ofrecias por el propio dataset. Además, se predice con éxito la tendencia en aquellas temperaturas de la subdivisión *Test*. Se calcula la precisión del modelo con el Promedio de la Suma de Errores (MSE), el Promedio de Error Absoluto (MAE) y R^2 mediante las siguientes rutinas:

```
mse = mean_squared_error(x_test.values, fcast1)
mae = mean_absolute_error(x_test.values, fcast1)
r2 = r2_score(x_test.values, fcast1)
```

Se obtiene lo siguiente:

MSE Test	**MAE Test**	R^2 **Test**
2.20	1.23	0.93

Cuadro 3: Estadísticos de precisión del modelo de Suavizado.

Es interesante notar que en contra del criterio multiplicativo escogido anteriormente, si se hubiese optado por un método aditivo se habría obtenido un MAE de 1.19 y un MSE de 1.99, ligeramente por debajo de los resultados del modelo multiplicativo (cabría realizar validación cruzada para comprobar la eficacia de un método u otro con solidez). Finalmente, se adjuntan en la siguiente tabla las 12 primeras predicciones (se omite mostrar el resto, las 24 predicciones restantes) del modelo con los datos de testeo:

Fecha	Pred.Suavizado	Test
2021-09	22.1	23.0
2021-10	18.4	18.1
2021-11	13.5	11.3
2021-12	10.8	10.9
2022-01	9.6	10.2
2022-02	10.3	11.8
2022-03	12.5	10.8
2022-04	14.7	14.1
2022-05	17.8	20.7
2022-06	22.2	24.7
2022-07	25.3	26.7
2022-08	25.4	27.2

Cuadro 4: Comparativa entre las predicciones del Modelo Suavizado y los datos de *Test*.

3.3. Método ARIMA

3.3.1. Manual

Se procede a realizar un análisis predictivo utilizando técnicas de modelaje ARIMA siguiendo la estrategia Box-Jenkins, seleccionando manualmente los parámetros que lo determinan. Para ello, se requiere que el proceso sea estacionario, en el sentido de que la media y la varianza deben permanecer constantes con el tiempo y la covarianza de la serie depender únicamente de su separación en el tiempo. Dada la ciclicidad de las temperaturas en un periodo de 12 meses, uno consigue la estacionariedad de los datos restando la temperatura por su decimosegundo regazo, esto es, $\hat{x}_t = x_t - x_{t-12}$. Se consigue con el siguiente código:

```
diff = df.diff(12)
```

Se representa a continuación la serie estacionaria:

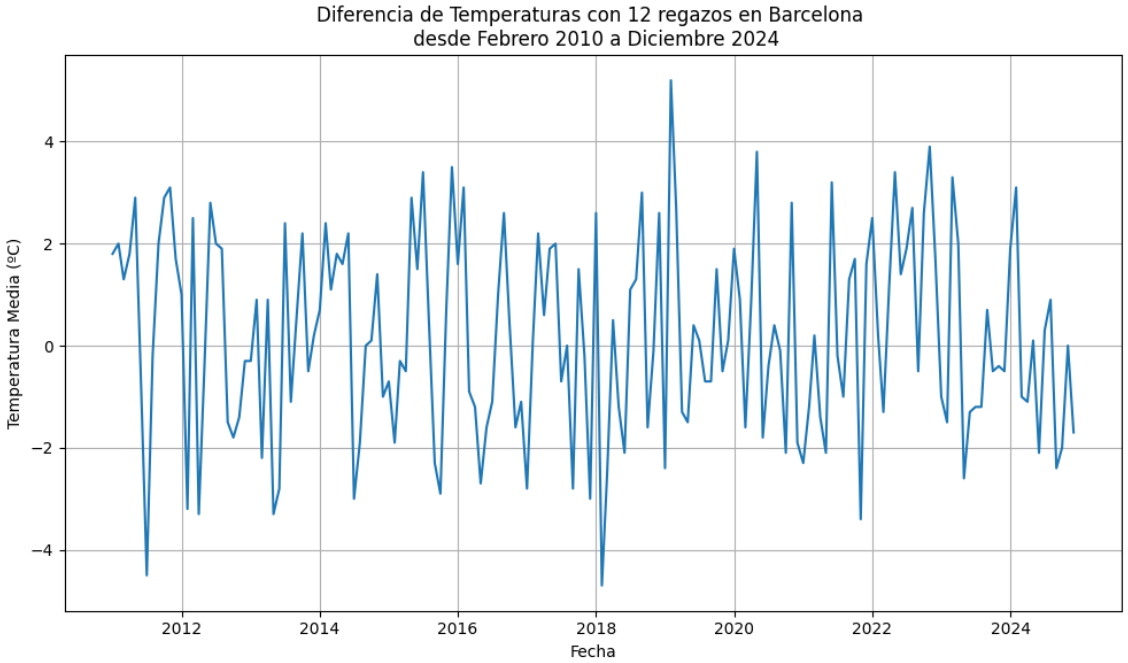

Figura 6: Temperaturas Medias diferenciadas con sus respectivos últimos 12 regazos.

Si bien un análisis visual permite confirmar cualitativamente la estacionariedad de los datos, se realiza un test de hipótesis para poner a prueba la proposición. Escogiendo un intervalo de confianza del 95 %, se pone

a prueba la hipótesis nula de que los datos no son estacionarios calculando el estadístico P con el siguiente código:

```
result = adfuller(diff.dropna())
```

Accediendo al resultado de interés con result[1], se obtiene un valor del estadístico P de 0.014, por debajo de 0.05 que es el indicado para el intervalo de confianza deseado, por lo que se rechaza la hipótesis nula. Se refuerza la hipótesis mostrando los correlogramas de las funciones ACF y PACF. La primera captura la relación lineal entre las variables de la serie separadas por un número determinado de regazos, mientras que la segunda realiza la misma función que la primera eliminando el efecto acumulativo correlativo del dato actual con todos sus anteriores. Así, una tendencia lineal decreciente y suave en los correlogramas indica la no estacionariedad de los datos, mientras que un corte abrupto en ellos es signo de lo contrario. Estas funciones permiten detectar la estacionariedad, el tipo de modelo y los retardos que son significativamente diferentes a 0. Se omite aquí aplicar alguna transformación sobre la variable objetivo (de tipo logaritmo, por ejemplo) para no añadir más complejidad al modelo en si. Así, se implementa con el siguiente código:

```
fig, ax = plt.subplots(ncols = 1, nrows = 2, figsize = (10,6))
plot_acf(diff.dropna(), lags = 36, ax = ax[0])
ax[0].grid()
ax[0].set_title('ACF')
plot_pacf(diff.dropna(), lags = 36, ax = ax[1])
ax[1].grid()
ax[1].set_title('PACF')
ax[1].set_xlabel('Regazo')
plt.tight_layout()
plt.show()
```

Se ha escogido mostrar las funciones ACF y PACF para los siguientes 36 regazos respecto al regazo 0. El resultado es el siguiente:

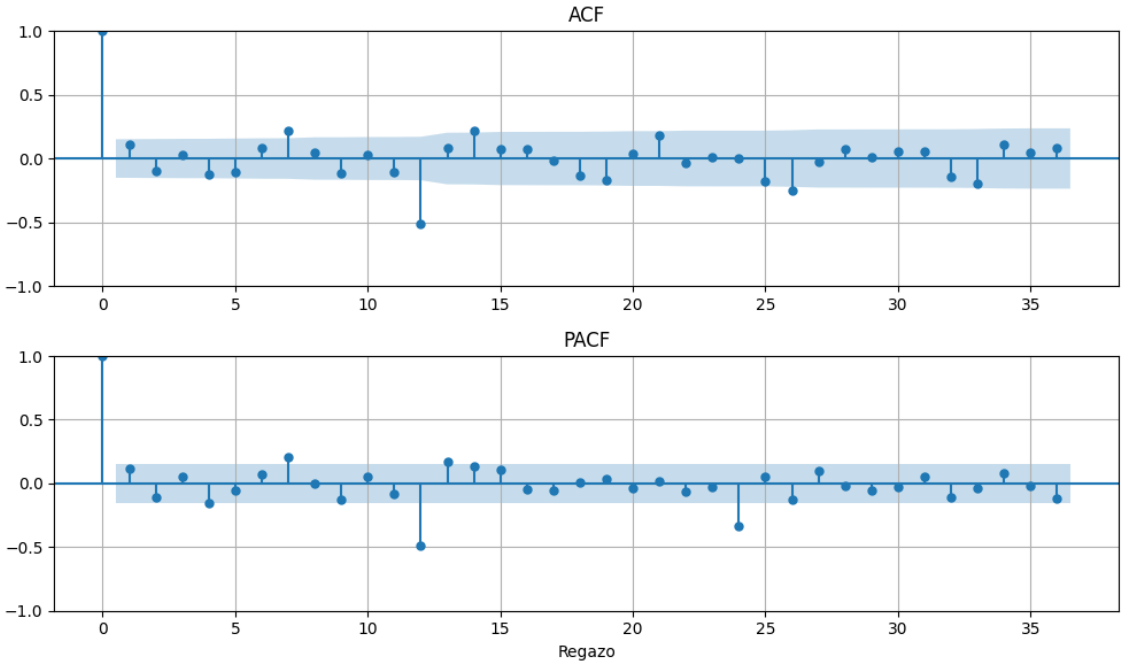

Figura 7: ACF y PACF para el Modelo ARIMA manual.

Aquellos que caigan dentro de la zona azul (bandas de confianza) dejan de ser relevantes para el análisis (se consideran igual a 0). El primer valor de ambos gráficos es 1 ya que esta es la correlación del primer regazo consigo mismo. Con esto, se observa un corte abrupto en la tendencia de las autocorrelaciones en ambos gráficos, lo que indica la estacionariedad del modelo. Se identifica además un comportamiento cíclico de las autcorrelaciones cada 12 meses, tal que las más fuertes suceden en los retardos múltiplos de 12, confirmando el

comportamiento estacional de los datos. Por ello, se deben asociar valores a 6 parámetros: p,d,q y P,D,Q:

- **p=0**: Representa el número de términos autorregresivos en el modelo. Indica el número de regazos anteriores a considerar para hallar el nuevo valor. En el PACF, se identifica un cambio abrupto en las autocorrelaciones entre el regazo 0 y 1, por lo que se toma 0 como el número de regazos a tener en cuenta en la componente autorregresiva.

- **d=0**: Representa el número de veces que se diferencian los datos para obtener estacionariedad. Se toma $d = 0$ ya que la diferenciación se había logrado con una única diferenciación estacional (de periodo 12 regazos).

- **q=0**: Representa el número de términos en la parte de medias móviles, utilizando los errores de las predicciones pasadas para obtener la actual. Se identifica un cambio abrupto en las autocorrelaciones en el ACF entre los regazos 0 y 1, sugiriendo $q = 0$.

- **P=2**: Mismo significado que p, pero aplicado para aquellos regazos múltiplos del periodo (12 meses). En el PACF las autocorrelaciones caen dentro de la franja de confianza a partir del regazo 36, entonces se escoge tomar P como 2.

- **D = 1**: Mismo significado que d pero para aquellos regazos múltiplos del periodo. Anteriormente se aplicó una única diferenciación de periodo 12 para obtener estacaionariedad en los datos, por lo que se toma $D = 1$.

- **Q = 1**: Mismo significado que q pero para aquellos regazos múltiplos del periodo. En el ACF, a partir del regazo 12 las autocorrelaciones dejan de ser relevantes, entonces se toma $Q = 1$.

Elegido el valor de los parámetros, se entrena el modelo ARIMA con la siguiente rutina:

```
manual_model = sm.tsa.ARIMA(x_train, order = (0,0,0), seasonal_order = (2,1,1,12))
results_manual = manual_model.fit()
```

Con results_manual.summary() se obtienen los parámetros del modelo ajustado junto a otros coeficientes que indican la eficiencia del método:

Coeficiente	**Valor**	**Desv. Est.**	**z**	**P>|z|**	**[0.025]**	**[0.975]**
ar.S.L12	-0.5920	0.183	-3.228	0.001	-0.951	-0.233
ar.S.L24	-0.3510	0.132	-2.667	0.008	-0.609	-0.093
ma.S.L12	-0.3864	0.186	-2.081	0.037	-0.750	-0.022
sigma2	1.8477	0.294	6.285	0.000	1.271	2.424

Cuadro 5: Resultados de los parámetros del modelo ARIMA Manual. El último coeficiente (sigma2) indica la varianza de los residuos.

Cabe mencionar que en los coeficientes anteriores se cumple que $P < 0{,}05$, por lo que son coeficientes significativos. Así, la expresión analítica que da cuenta de las predicciones a partir de los valores de sus regazos toma la siguiente forma:

$$
\begin{aligned}
&(1-\Phi_1 B^{12}-\Phi_2 B^{24})(1-B^{12})^1 X_t && (1)\\
&\quad = (1+\Theta_1 B^{12}) Z_t && (2)\\
\Rightarrow &(1+0{,}59B^{12}+0{,}35B^{24})(1-B^{12})X_t && (3)\\
&\quad = (1-0{,}39B^{12}) Z_t && (4)\\
\Rightarrow &(1+0{,}59B^{12}+0{,}35B^{24})(X_t - X_{t-12}) && (5)\\
&\quad = Z_t - 0{,}39 Z_{t-12} && (6)\\
\Rightarrow &X_t - 0{,}41 X_{t-12} - 0{,}24 X_{t-24} - 0{,}35 X_{t-36} && (7)\\
&\quad = Z_t - 0{,}39 Z_{t-12} && (8)\\
\Rightarrow &X_t = +0{,}41 X_{t-12} + 0{,}24 X_{t-24} + 0{,}35 X_{t-36} + Z_t - 0{,}39 Z_{t-12} && (9)\\
& && (10)
\end{aligned}
$$

Por otra parte, es necesario comprobar que el ruido producido por el modelo es de tipo *blanco gaussiano*, tal que los residuos estimados tengan media 0, varianza constante y ausencia de correlación para cualquier retardo (que no haya ninguna dependencia entre ellos no explicada por el modelo). Se comprueba dicha proposición con la siguiente rutina:

```
results_manual.plot_diagnostics(figsize = (10,6))
plt.tight_layout()
plt.show()
```

El resultado es el siguiente:

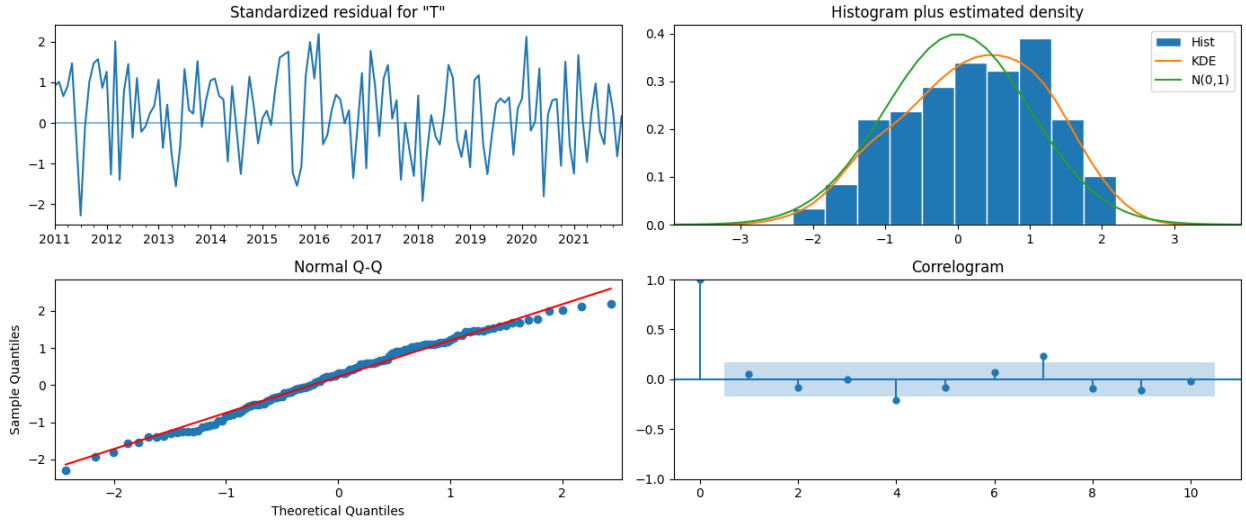

Figura 8: Análisis de los residuos del modelo ARIMA Manual.

Por una parte, se observa en el primer gráfico la distribución aleatoria de los residuos alrededor de la media 0 con varianza constante. Paralelamente, se representa en un histograma la dsitribución de dichos residuos junto a una distribución normal $N(0,1)$ (media 0 y desviación estándar 1), mostrando un comportamiento gaussiano y verificando su aleatoriedad (falta de correlación entre ellos). Por otra parte, el Q-Q plot muestra la comparación entre los cuantiles de los residuos estandarizados y los de una distribución normal teórica. Los puntos se ajustan con éxito a la linea, entonces los residuos siguen una distribución normal. Por último, en el cuarto gráfico se muestra el correlograma que da cuenta de la función ACF. La mayoría de regazos caen dentro de las bandas de confianza, por lo que no hay autocorrelación significativa. Se apoya la anterior proposición con el estadístico de Ljung-Box, que suma las autocorrelaciones de los residuos y comprueba que estén verdaderamente incorrelacionados. Se obtiene con results_manual.summary(), donde para el presente caso toma el valor de 0.37, superior a 0.05 (el indicado por un intervalo de confianza del 95 %), por lo que se confirma que los residuos se hallan incorrelacionados.

Habiendo asegurado la proposición anterior, se procede a predecir las observaciones de la subdivisión *Test* con el siguiente código (con su respectivo intervalo de confianza):

```
predictions1 = results_manual.get_forecast(steps = 36)
conf_limits1 = predictions1.conf_int()
```

Si se representan las predicciones junto a los datos *Test* y los límites de confianza, se obtiene lo siguiente:

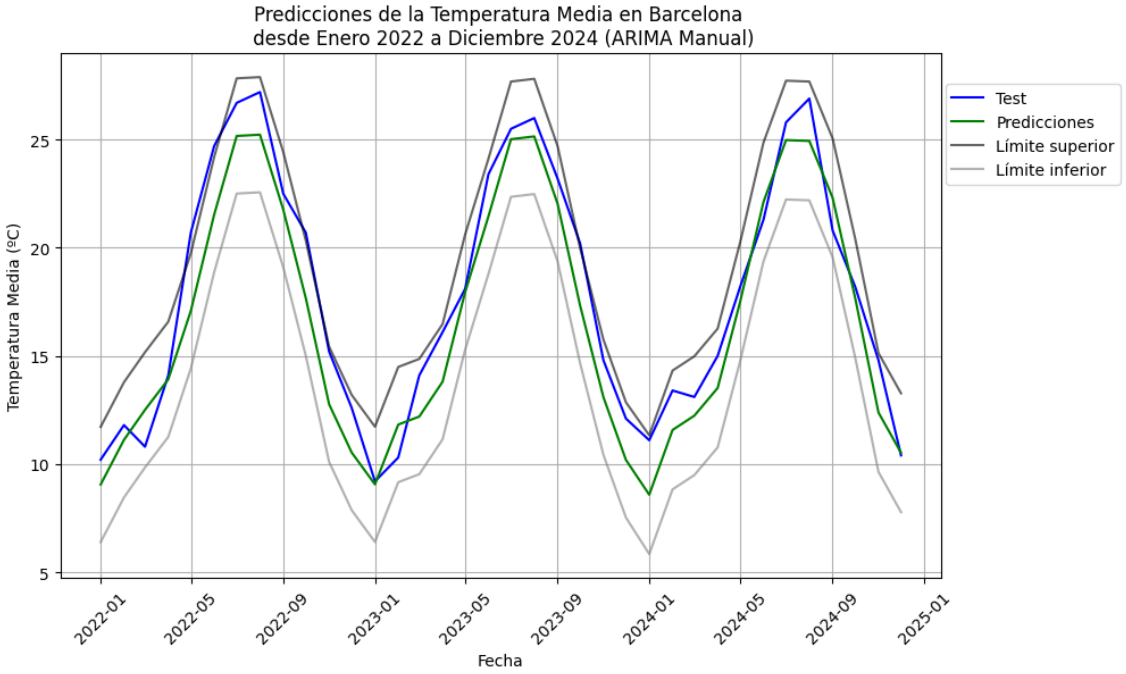

Figura 9: Predicciones del modelo ARIMA Manual para la subdivisión *Test* de los datos.

Se opta a continuación por analizar los estadísticos de mayor relevancia, entre ellos el AIC y el BIC que equilibran la bondad del modelo ajustando su complejidad, penalizando los modelos más complejos para evitar el sobreajuste (penaliza menos el BIC que el AIC al querer minimizar la pérdida de información). Se incluyen también el MSE, MAE y R^2 obtenidos a partir de los datos de testeo. Además, se muestra el valor de la Verosimilitud, con la que se construye el AIC y BIC:

Log Likelihood	**AIC**	**BIC**	**MSE Test**	**MAE Test**	R^2 **Test**
-234.62	477.24	488.77	3.14	1.52	0.90

Cuadro 6: Estadísticos de precisión del modelo ARIMA Manual.

3.3.2. Automático

Alternativamente, uno puede implementar una rutina que dados unos valores mínimos y máximos para los parámetros p,d,q y P,D,Q se busque aquel modelo que minimice el AIC, un coeficiente estadístico que da cuenta de la bondad del ajuste, penalizando el modelo cuantos más parámetros incluya y favoreciéndolo cuanto mejor sea el ajuste. Este método permite implementar gran parte de los pasos de la Estrategia Box-Jenkins para hallar el mejor modelo. Se implementa como sigue:

```
arima_model = pm.auto_arima(x_train, start_p = 0, start_q = 0,
                            max_p = 2, max_q = 2, m = 24,
                            start_P = 0, seasonal = True, d = 0, D = 1,
                            trace = True, error_action = 'ignore',
                            suppress_warnings = True, stepwise = True)
```

El resultado es el siguiente:

```
Performing stepwise search to minimize aic
 ARIMA(0,0,0)(0,1,1)[12] intercept   : AIC=inf, Time=0.76 sec
 ARIMA(0,0,0)(0,1,0)[12] intercept   : AIC=540.228, Time=0.04 sec
 ARIMA(1,0,0)(1,1,0)[12] intercept   : AIC=498.257, Time=0.26 sec
 ARIMA(0,0,1)(0,1,1)[12] intercept   : AIC=inf, Time=1.07 sec
 ARIMA(0,0,0)(0,1,0)[12] intercept   : AIC=538.726, Time=0.04 sec
 ARIMA(1,0,0)(0,1,0)[12] intercept   : AIC=541.148, Time=0.07 sec
 ARIMA(1,0,0)(2,1,0)[12] intercept   : AIC=466.933, Time=0.59 sec
 ARIMA(1,0,0)(2,1,1)[12] intercept   : AIC=463.732, Time=1.36 sec
 ARIMA(1,0,0)(1,1,1)[12] intercept   : AIC=465.099, Time=0.48 sec
 ARIMA(1,0,0)(2,1,2)[12] intercept   : AIC=464.203, Time=1.97 sec
 ARIMA(1,0,0)(1,1,2)[12] intercept   : AIC=inf, Time=5.51 sec
 ARIMA(0,0,0)(2,1,1)[12] intercept   : AIC=462.567, Time=0.78 sec
 ARIMA(0,0,0)(1,1,1)[12] intercept   : AIC=464.007, Time=0.36 sec
 ARIMA(0,0,0)(2,1,0)[12] intercept   : AIC=465.174, Time=0.44 sec
 ARIMA(0,0,0)(2,1,2)[12] intercept   : AIC=462.794, Time=1.34 sec
 ARIMA(0,0,0)(1,1,0)[12] intercept   : AIC=497.305, Time=0.18 sec
 ARIMA(0,0,0)(1,1,2)[12] intercept   : AIC=inf, Time=2.90 sec
 ARIMA(0,0,1)(2,1,1)[12] intercept   : AIC=463.577, Time=0.90 sec
 ARIMA(1,0,1)(2,1,1)[12] intercept   : AIC=461.886, Time=3.93 sec
 ARIMA(1,0,1)(1,1,1)[12] intercept   : AIC=463.881, Time=1.49 sec
 ARIMA(1,0,1)(2,1,0)[12] intercept   : AIC=463.269, Time=1.78 sec
 ARIMA(1,0,1)(2,1,2)[12] intercept   : AIC=461.967, Time=2.49 sec
 ARIMA(1,0,1)(1,1,0)[12] intercept   : AIC=inf, Time=1.38 sec
 ARIMA(1,0,1)(1,1,2)[12] intercept   : AIC=inf, Time=5.20 sec
 ARIMA(2,0,1)(2,1,1)[12] intercept   : AIC=465.354, Time=2.23 sec
 ARIMA(1,0,2)(2,1,1)[12] intercept   : AIC=465.470, Time=2.48 sec
 ARIMA(0,0,2)(2,1,1)[12] intercept   : AIC=464.370, Time=1.00 sec
 ARIMA(2,0,0)(2,1,1)[12] intercept   : AIC=464.883, Time=1.20 sec
 ARIMA(2,0,2)(2,1,1)[12] intercept   : AIC=464.957, Time=6.68 sec
 ARIMA(1,0,1)(2,1,1)[12] intercept   : AIC=464.274, Time=1.47 sec

Best model:  ARIMA(1,0,1)(2,1,1)[12] intercept
Total fit time: 50.409 seconds
```

Entonces, el modelo que mejor AIC ofrece es aquel donde se toma $p = 1$, $d = 0$, $q = 1$ y $P = 2$, $D = 1$, $Q = 1$. Se entrena el modelo con los nuevos parámetros escogidos:

```
auto_model = sm.tsa.ARIMA(x_train, order = (1,0,1), seasonal_order = (2,1,1,12))
results_auto = auto_model.fit()
```

Con results_auto.summary() se obtienen los coeficientes del modelo:

Coeficiente	**Valor**	**Desv. Est.**	**z**	**P>|z|**	**[0.025]**	**[0.975]**
ar.L1	-0.7820	0.109	-7.155	0	-0.996	-0.568
ma.L1	0.9226	0.090	10.295	0	0.747	1.098
ar.S.L12	-0.5771	0.182	-3.165	0.002	-0.934	-0.220
ar.S.L24	-0.3644	0.135	-2.698	0.007	-0.629	-0.100
ma.S.L12	-0.3784	0.201	-1.881	0.060	-0.773	0.016
sigma2	1.7918	0.296	6.053	0	1.212	2.372

Cuadro 7: Resultados de los parámetros del modelo ARIMA Automático.

Comprobando $P > |z|$, se halla de nuevo que los coeficientes son significativos, por lo que la expresión analítica del modelo ARIMA Automático queda de la siguiente forma:

$$(1-\Phi_1 B^{12}-\Phi_2 B^{24})(1-\phi_1 B)(1-B^{12})^1(1-B)^0 X_t \tag{11}$$
$$=(1+\Theta_1 B^{12})(1+\theta_1 B^1)Z_t \tag{12}$$
$$\Rightarrow(1+0{,}58B^{12}+0{,}36B^{24})(1+0{,}78B)(1-B^{12})X_t \tag{13}$$
$$=(1-0{,}38B^{12})(1+0{,}92B^1)Z_t \tag{14}$$
$$\Rightarrow(1+0{,}58B^{12}+0{,}36B^{24})(1+0{,}78B)(X_t-X_{t-12}) \tag{15}$$
$$=(1-0{,}38B^{12})(Z_t+0{,}92Z_{t-1}) \tag{16}$$
$$\Rightarrow(1+0{,}58B^{12}+0{,}36B^{24})(X_t-X_{t-12}+0{,}78X_{t-1}-0{,}78X_{t-13}) \tag{17}$$
$$=Z_t+0{,}92Z_{t-1}-0{,}38Z_{t-12}-0{,}35Z_{t-13} \tag{18}$$
$$\Rightarrow X_t+0{,}78X_{t-1}-0{,}42X_{t-12}-0{,}34X_{t-13} \tag{19}$$
$$-0{,}22X_{t-24}-0{,}17X_{t-25}-0{,}36X_{t-36}-0{,}28X_{t-37} \tag{20}$$
$$=Z_t+0{,}92Z_{t-1}-0{,}38Z_{t-12}-0{,}35Z_{t-13} \tag{21}$$
$$\Rightarrow X_t=-0{,}78X_{t-1}+0{,}42X_{t-12}+0{,}34X_{t-13}+0{,}22X_{t-24} \tag{22}$$
$$+0{,}17X_{t-25}+0{,}36X_{t-36}+0{,}28X_{t-37} \tag{23}$$
$$+Z_t+0{,}92Z_{t-1}-0{,}38Z_{t-12}-0{,}35Z_{t-13} \tag{24}$$

El análisis de sus residuos se muestra a continuación:

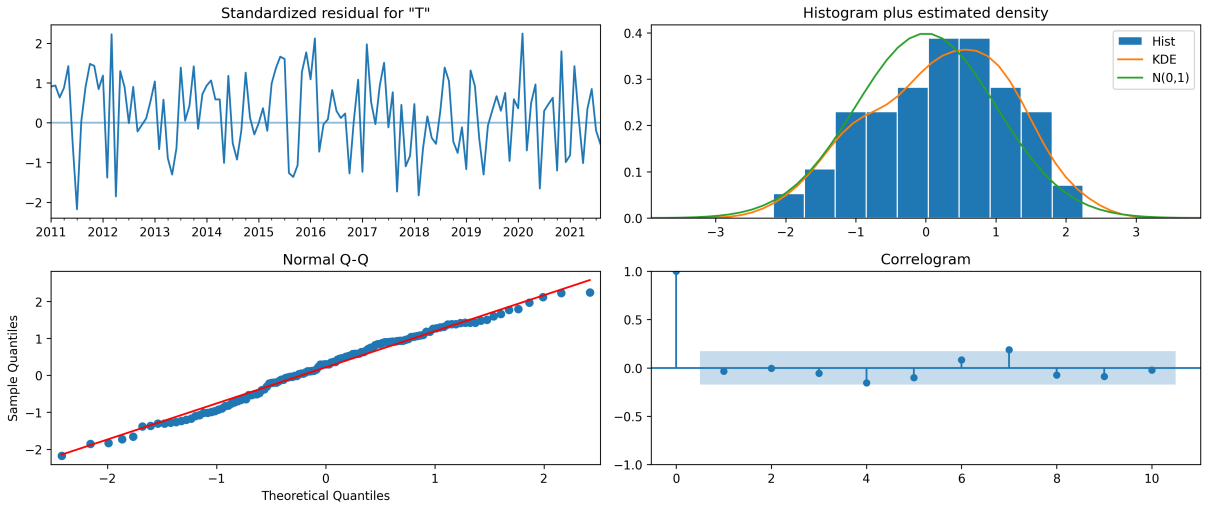

Figura 10: Análisis de los residuos del modelo ARIMA Automático.

Como se observa en los gráficos anteriores, los residuales siguen una distribución normal (N(0,1)), respondiendo a una distrbución aleatoria de media 0. Además, no se observa ninguna correlación entre ellos, que es lo que uno espera para poder confiar en el modelo. Dicha proposición se sustenta sobre el estadístico de Ljung-Box, que toma el valor de 0.54, entonces los residuos siguen una distribución normal, típica de valores incorrelacionados. En la siguiente tabla se resumen los estadísticos de precisión principales de dicho modelo:

Log Likelihood	**AIC**	**BIC**	**MSE Test**	**MAE Test**	R^2 **Test**
-231.79	475.58	492.87	3.18	1.53	0.90

Cuadro 8: Estadísticos de precisión del modelo ARIMA Automático.

A continuación, se grafican las predicciones para los datos de prueba del modelo ARIMA Automático:

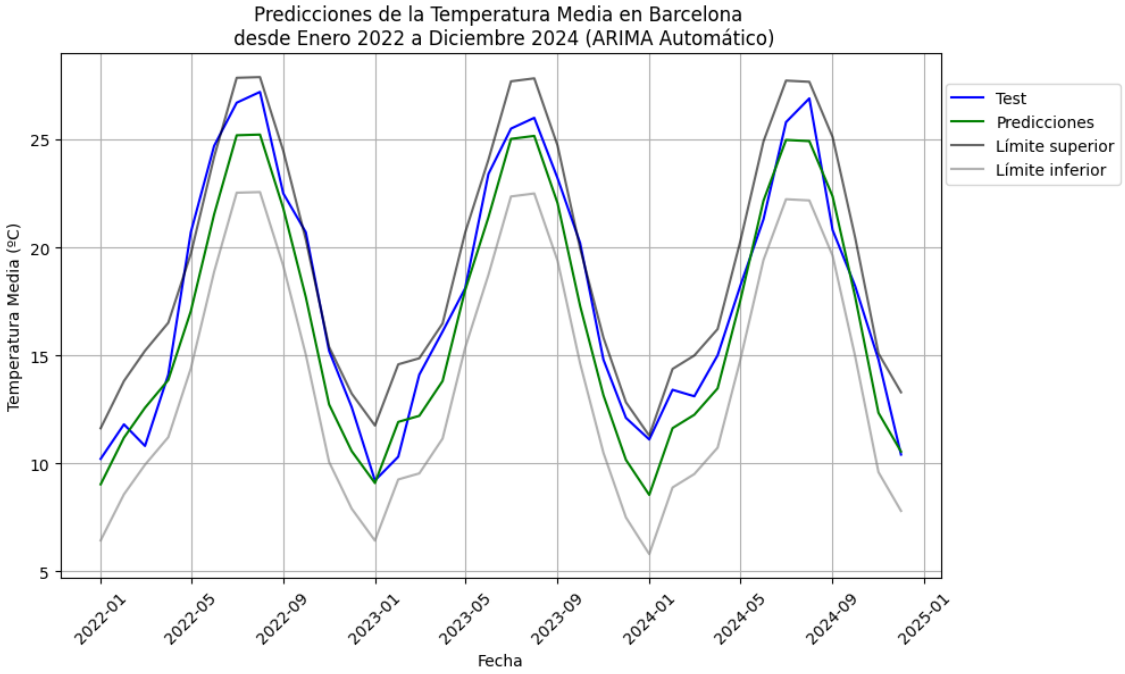

Figura 11: Predicciones del modelo ARIMA Automático para la subdivisión *Test* de los datos.

Se comparan en la siguiente tabla los estadísticos de ambos modelos (ARIMA Manual y Automático):

Modelo	**Log Likelihood**	**AIC**	**BIC**	**MSE Test**	**MAE Test**	**R^2 Test**
Manual	-234.62	477.24	488.77	3.14	1.52	0.92
Automático	-226.14	475.58	492.87	3.18	1.53	0.90

Cuadro 9: Estadísticos de precisión de los modelos ARIMA Manual y Automático.

Cabe aquí argumentar que un mejor AIC no conduce a mejores resultados en MSE y MAE con los datos de prueba, hecho que se verifica en el presente ejemplo donde un modelo Manual supera a un modelo Automático (quedaría utilizar validación cruzada para apoyar con mayor solidez dicha proposición). Así, si bien el rendimiento de ambos es muy parecido, se prefiere escoger el Modelo Manual dada su mejor precisión en los estadísticos MAE, MSE y R^2 y su mayor simplicidad (con menos parámetros en su expresión analítica).

Con esto, se calculan las predicciones del modelo ganador para la subdivisión de los datos *Test* y para los siguientes 12 meses (calculada con results_auto.get_forecast(steps = 48).predicted_mean)(se muestran únicamente las últimas 20 predicciones):

Fecha	Predicciones	Test
2024-05	17.5	18.2
2024-06	22.2	21.3
2024-07	25.0	25.8
2024-08	24.9	26.9
2024-09	22.4	20.8
2024-10	17.7	18.2
2024-11	12.3	14.8
2024-12	10.5	10.4
2025-01	8.8	-
2025-02	11.5	-
2025-03	12.4	-
2025-04	13.7	-
2025-05	17.4	-
2025-06	21.7	-
2025-07	25.1	-
2025-08	25.1	-
2025-09	22.1	-
2025-10	17.6	-
2025-11	12.7	-
2025-12	10.5	-

Cuadro 10: Tabla de Predicciones y Valores Teóricos para el Modelo ARIMA Manual.

Se muestran gráficamente a continuación:

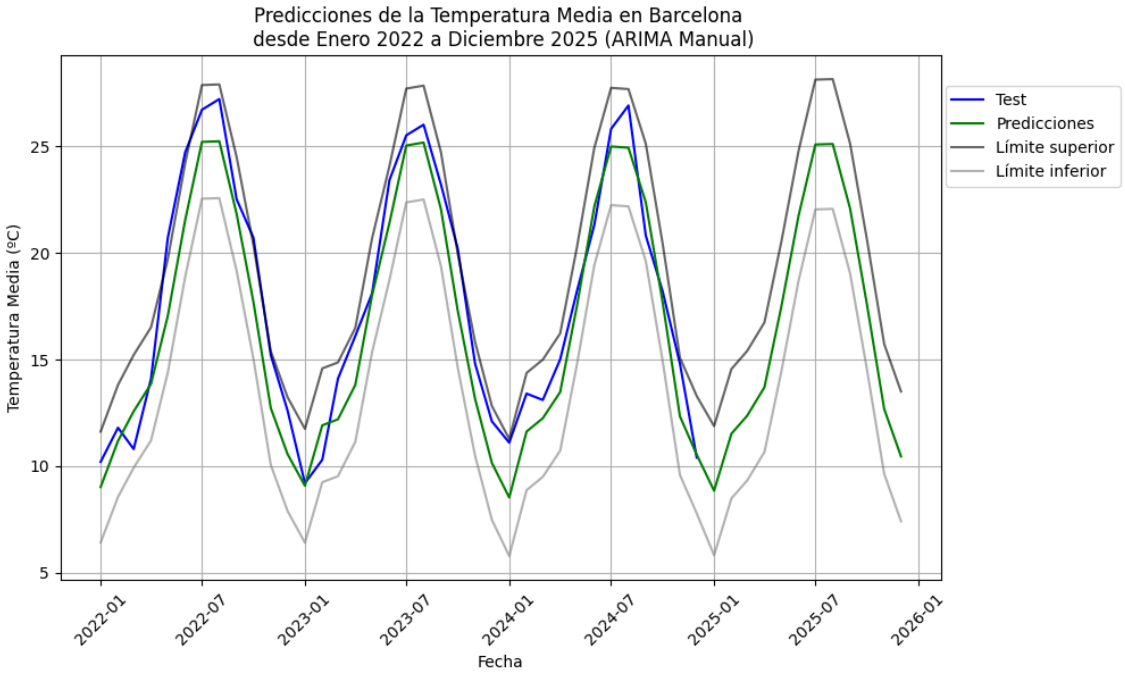

Figura 12: Predicciones de las Temperaturas Medias por el modelo ARIMA Manual.

Se observa como las predicciones se ajustan con éxito a la subdivisión de los datos *Test*. Por último, se comparan las 12 últimas predicciones para la subdivisión *Test* realizadas por los modelos ARIMA Manual y Suavizado:

Fecha	Pred. ARIMA	Pred. Suav.	Test
2024-01	8.5	9.3	11.1
2024-02	11.6	10.1	13.4
2024-03	12.2	12.3	13.1
2024-04	13.5	14.7	15.0
2024-05	17.5	17.8	18.2
2024-06	22.2	22.4	21.3
2024-07	25.0	25.5	25.8
2024-08	24.9	25.6	26.9
2024-09	22.4	22.5	20.8
2024-10	17.7	18.6	18.2
2024-11	12.3	13.3	14.8
2024-12	10.5	10.7	10.4

Cuadro 11: Predicciones realizadas por los modelos ARIMA Manual y Suavizado con la subdivisión *Test*.

Gráficamente, los datos anteriores se muestran como sigue:

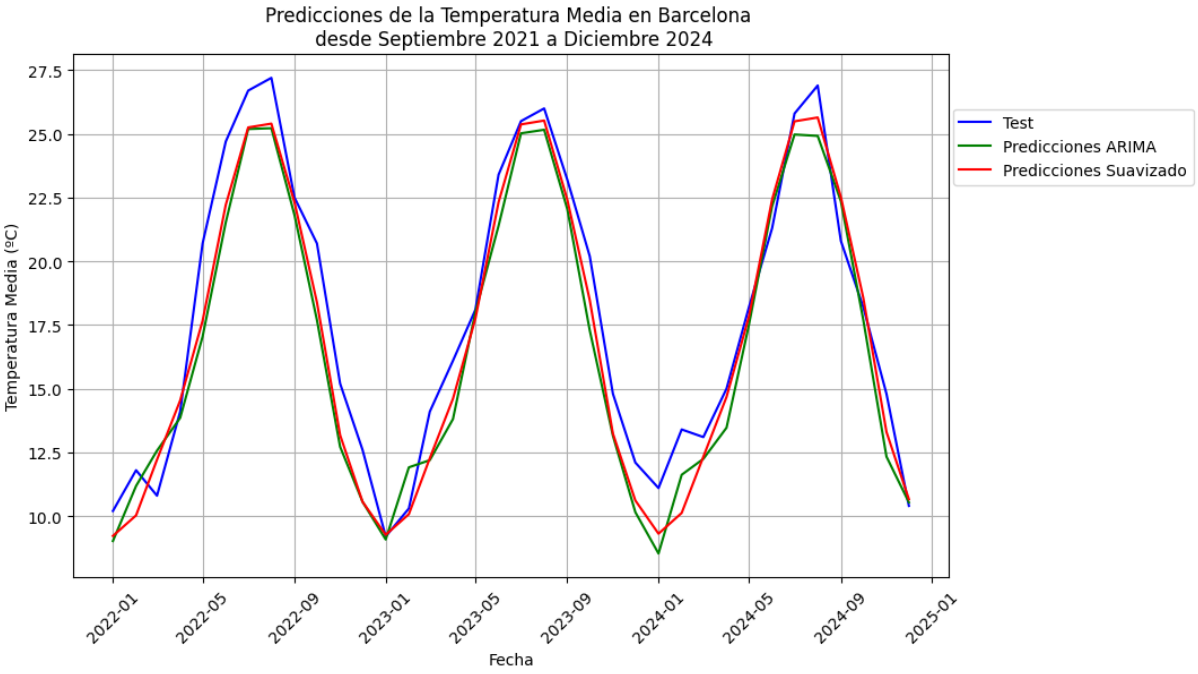

Figura 13: Predicciones de las Temperaturas Medias por el modelo ARIMA Manual y Suavizado.

Se escoge entre un modelo u otro en base a los siguientes estadísticos:

Modelo	MSE	MAE	R^2
ARIMA Manual	3.14	1.52	0.90
Suavizado	2.20	1.23	0.93

Cuadro 12: Comparativa del rendimiento entre los modelos ARIMA Manual y Suavizado.

Se observa que en todos los estadísticos analizados el método de Suavizado rinde por encima del método ARIMA. Además, la complejidad de los modelos ARIMA es bastante superior a la de los modelos de Suavizado al estar diseñados los primeros para capturar patrones más complejos o tendencias a largo plazo. Estos argumentos consolidan la conclusión de que los métodos de Suavizado ofrecen (para el dataset actual) resultados más precisos que los ofrecidos por ARIMA con una menor coste computacional.

4. Conclusiones

Los hallazgos y observaciones realizados hasta ahora se resumen en las siguientes conclusiones:

1. Existe una fuerte componente estacional de periodo 12 meses en las Temperaturas Medias Mensuales entre Febrero 2010 y Diciembre 2024 en el territorio de Barcelona.
2. Existe una tendencia creciente consistente en las Temperaturas Medias a lo largo de la ventana temporal analizada.
3. Las funciones ACF y PACF permiten indicar con éxito los parámetros a introducir en los modelos ARIMA.
4. Un modelo ARIMA Automático no siempre asegura mejores resultados que un modelo ARIMA Manual: deben de tenerse en cuenta más estadísticos de precisión para considerar un modelo u otro.
5. El modelo de Suavizado ofrece mejores resultados que el modelo ARIMA Manual al combinar buena precisión con simplicidad en el ajuste.

5. Bibliografía

[1] *Monthly average air temperatures of the city of Barcelona since 1780* .(2019). datos.gob.es.Link.

[2] Alonso Revenga, Juana María. *Tema 1: Análisis Descriptivo de una Serie Temporal.*(2025). Universidad Complutense de Madrid.

[3] Alonso Revenga, Juana María. *Tema 2: Modelos ARIMA.*(2025). Universidad Complutense de Madrid.