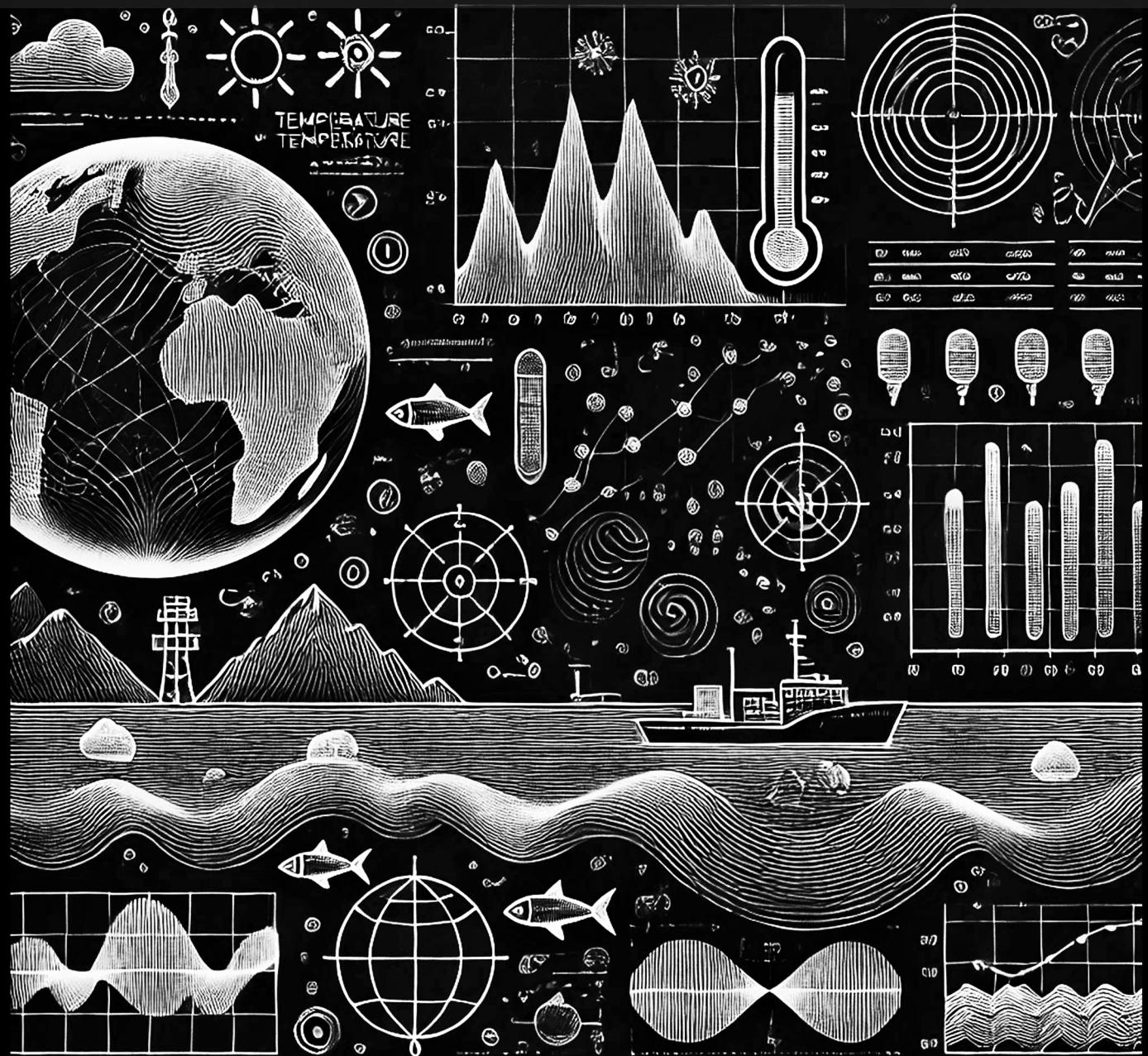


Series Temporales de Temp. Oceánicas

Práctica 2 de el módulo de Minería de Datos y Modelización Predictiva

Laura Rodríguez Ropero



Series Temporales de Temp. Oceánicas

Práctica 2 de el módulo de Minería de Datos y
Modelización Predictiva

por

Laura Rodríguez Ropero

20/02/2025

Índice

| | |
|---|----|
| 1. Introducción y Objetivos | 1 |
| 2. Análisis Gráfico y Descomposición de la Serie | 2 |
| 3. Reserva de Datos para Validación de Modelos | 5 |
| 4. Ajuste de un Modelo de Suavizado Exponencial | 6 |
| 5. Análisis de Autocorrelación y Selección del Modelo ARIMA | 9 |
| 6. Expresión Algebraica del Modelo Ajustado | 13 |
| 7. Predicciones e Intervalos de Confianza | 14 |
| 8. Comparación de Predicciones y Conclusiones | 16 |

Profesor: Juan Antonio Guevara
Facultad: Facultad de Estudios Estadísticos
Universidad: Universidad Complutense de Madrid
Máster: Big Data, Data Science e Inteligencia Artificial



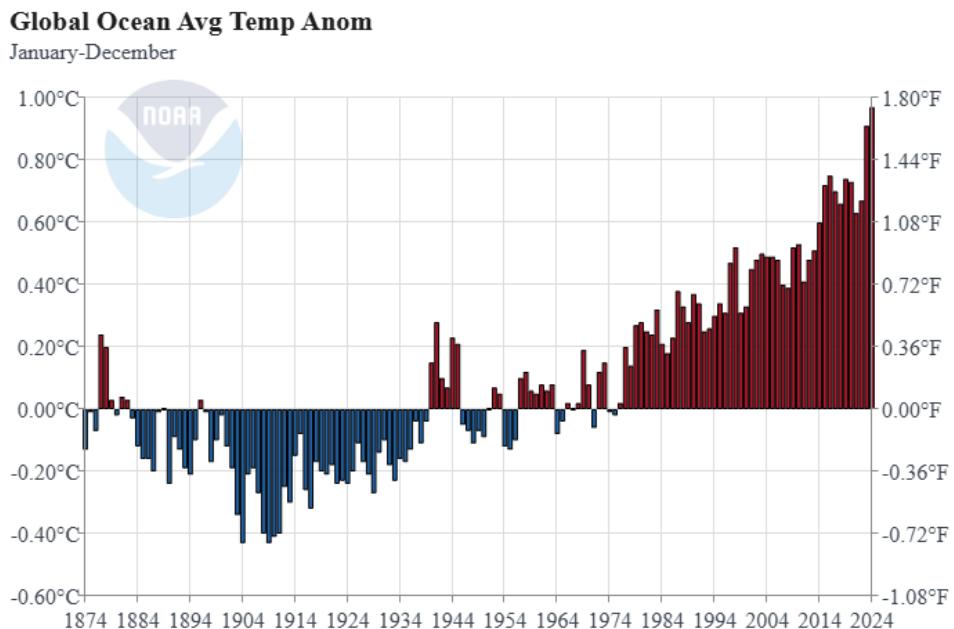
1

Introducción y Objetivos

En este trabajo se lleva a cabo un análisis de las temperaturas globales superficiales en los océanos, utilizando el conjunto de datos proporcionado por el National Centers for Environmental Information (NCEI) de la National Oceanic and Atmospheric Administration (NOAA). Este conjunto de datos [1] incluye las temperaturas promedio mensuales de la superficie oceánica a nivel global desde 1874 hasta la fecha, representadas como una serie temporal que permite observar tendencias, patrones estacionales y anomalías térmicas a lo largo del tiempo.

El objetivo principal de este análisis es estudiar la evolución de las temperaturas oceánicas globales, centrándose en las variaciones de largo plazo asociadas con fenómenos climáticos naturales y antropogénicos, como el cambio climático.

Desde una perspectiva metodológica, este análisis se enmarca en el contexto de la minería de datos y la modelización predictiva, cumpliendo con los requisitos de trabajar con una serie temporal con estacionalidad y aproximadamente 150 observaciones seleccionadas de este conjunto de datos. Se implementan técnicas estadísticas y de visualización de datos para descomponer la serie temporal, identificar tendencias y patrones estacionales, y modelar su comportamiento futuro.

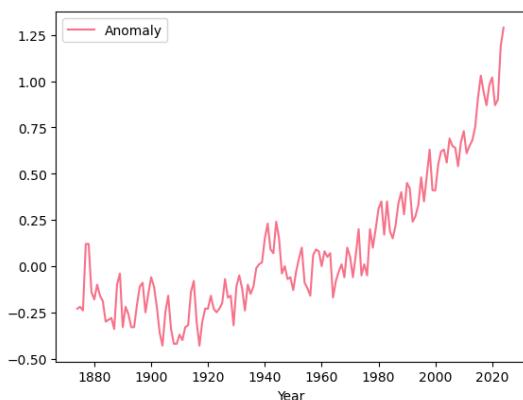


2

Análisis Gráfico y Descomposición de la Serie

El conjunto de datos presenta las anomalías de temperatura global de la superficie terrestre y oceánica, expresadas en grados Celsius, con respecto al periodo base de 1901-2000. Esto significa que las desviaciones de temperatura se calculan tomando como referencia el promedio de temperaturas registrado durante ese siglo. El uso de un periodo base es crucial para evaluar tendencias climáticas, ya que proporciona un marco de comparación estable. En este caso, se observa que las anomalías de temperatura eran predominantemente negativas antes de la segunda mitad del siglo XX, indicando temperaturas más frías que el promedio del periodo base. Sin embargo, a partir de las últimas décadas del siglo XX y especialmente en el siglo XXI, las anomalías se vuelven consistentemente positivas, reflejando un claro aumento de las temperaturas globales, lo que está alineado con el calentamiento global impulsado por el cambio climático.

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import numpy as np
4 import seaborn as sns
5 from statsmodels.tsa.seasonal import seasonal_decompose
6 from statsmodels.tsa.holtwinters import ExponentialSmoothing
7 import statsmodels.api as sm
8
9 data = pd.read_csv("C:/Users/lrodr/OneDrive/Documentos/master_ucm/trabajos/7/data.csv")
10 data["Year"] = pd.to_datetime(data["Year"], format="%Y")
11 data.index = data['Year']
12 del data['Year']
13
14 sns.lineplot(data)
15 plt.show()
```



Se observa que la serie no es estacionaria, ya que la media muestra una tendencia creciente y la varianza no parece constante, lo cuál significa que probablemente se trate de una serie heterocedástica. Tampoco se observa una estacionalidad demasiado evidente, pero no lo podemos descartar.

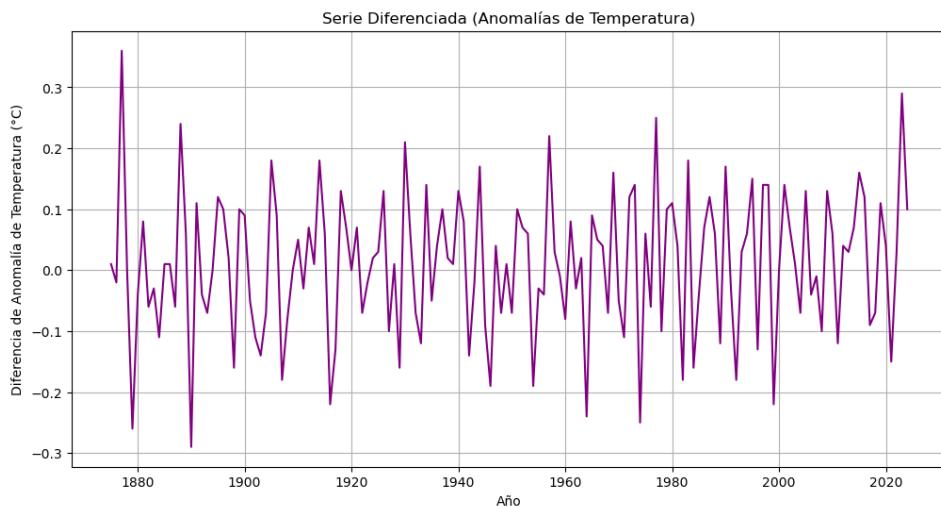
Para estabilizar la serie, el tratamiento adecuado sería diferenciarla para lograr estacionariedad y, si es necesario, aplicar una transformación logarítmica para corregir la heterocedasticidad. Además, parece haber un comportamiento estacional, posiblemente asociado a ciclos climáticos, lo que sugiere que un modelo SARIMA o una descomposición estacional podrían ser apropiados.

Como la serie contiene valores negativos y positivos, la aplicación directa del logaritmo no es posible, ya que la función logarítmica está definida únicamente para valores positivos, por lo tanto la hemos descartado como opción.

Vamos a probar a aplicar una diferenciación de primer orden para eliminar la tendencia.

```

1 data_diff = data["Anomaly"].diff().dropna()
2 data_diff_df = data_diff.reset_index()
3
4 plt.figure(figsize=(12, 6))
5 sns.lineplot(data=data_diff_df, x="Year", y="Anomaly", color="purple")
6 plt.title("Serie Diferenciada (Anomalías de Temperatura)")
7 plt.xlabel("Año")
8 plt.ylabel("Diferencia de Anomalía de Temperatura (°C)")
9 plt.grid(True)
10 plt.show()
```

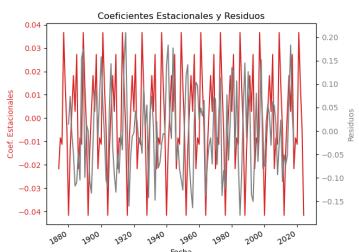


Ahora la serie diferenciada parece más estable, sin una tendencia clara. Se observa un comportamiento más errático, característico de una serie estacionaria. La diferenciación ha eliminado la tendencia.

Ahora procedemos a obtener la descomposición de la serie. Dado que las anomalías de temperatura incluyen valores negativos, la mejor opción es aplicar un modelo aditivo.

```

1 data_de = seasonal_decompose(data, model='additive', period=12)
2
3 fig, ax1 = plt.subplots()
4 color = 'tab:red'
5 ax1.set_xlabel('Fecha')
6 ax1.set_ylabel('Coef. Estacionales', color=color)
7 ax1.plot(data_de.seasonal, color=color)
8 ax1.tick_params(axis='y', labelcolor=color)
9 plt.xticks(rotation=30, ha='right')
10 ax2 = ax1.twinx() # Compartir el eje x
11 color = 'tab:grey'
12 ax2.set_ylabel('Residuos', color=color)
13 ax2.plot(data_de.resid, color=color)
14 ax2.tick_params(axis='y', labelcolor=color)
15 plt.title('Coeficientes Estacionales y Residuos')
16 plt.show()
```

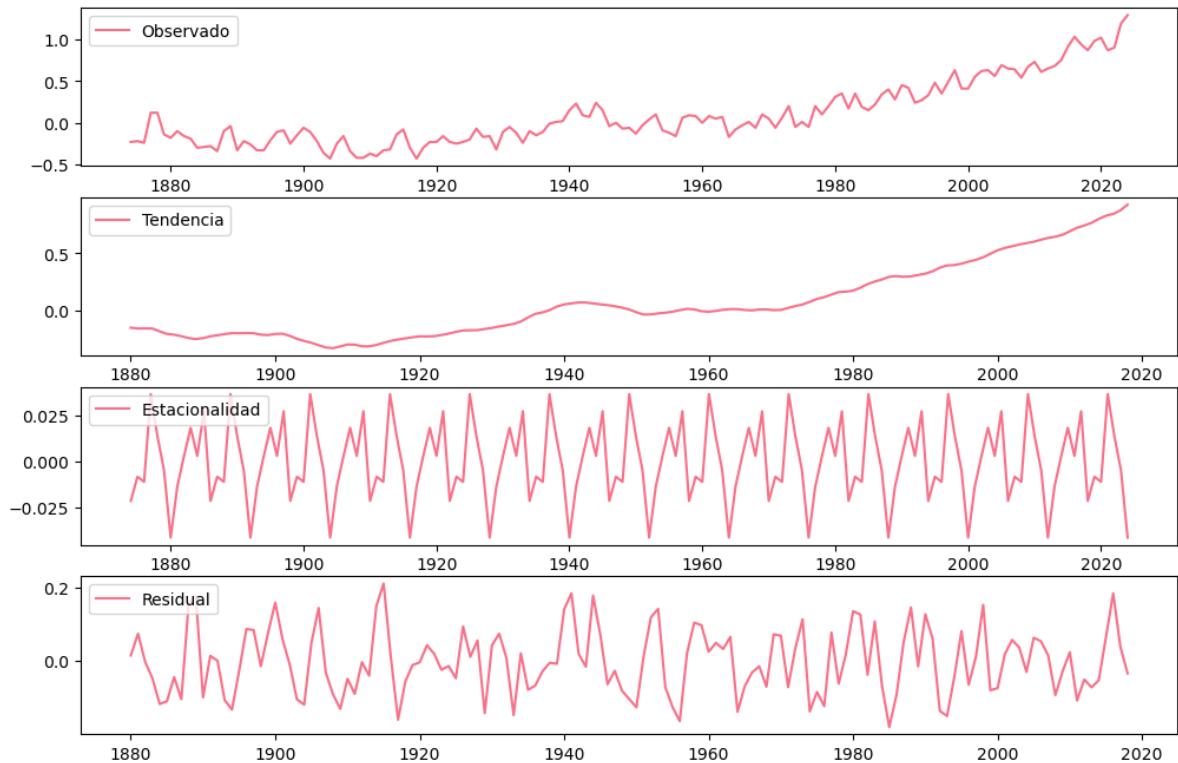


Al representar la gráfica de esta manera, no se observan las fechas en todos los gráficos, por lo que vamos a visualizar los componentes de la siguiente forma:

```

1 plt.figure(figsize=(12, 8))
2 plt.subplot(4, 1, 1)
3 plt.plot(data_de.observed, label='Observado')
4 plt.legend(loc='upper left')
5 plt.subplot(4, 1, 2)
6 plt.plot(data_de.trend, label='Tendencia')
7 plt.legend(loc='upper left')
8 plt.subplot(4, 1, 3)
9 plt.plot(data_de.seasonal, label='Estacionalidad')
10 plt.legend(loc='upper left')
11 plt.subplot(4, 1, 4)
12 plt.plot(data_de.resid, label='Residual')
13 plt.legend(loc='upper left')
14 plt.show()

```



Aquí está la descomposición aditiva de la serie de anomalías de temperatura. Se pueden observar los siguientes componentes:

- Serie Original: Muestra la tendencia general de aumento de la temperatura global con fluctuaciones anuales.
- Tendencia: Un aumento progresivo en las anomalías de temperatura, especialmente desde mediados del siglo XX.
- Estacionalidad: Un patrón recurrente de variación que parece repetirse a lo largo de los años.
- Residuo: La parte aleatoria de la serie, que representa las fluctuaciones no explicadas por la tendencia o la estacionalidad.

3

Reserva de Datos para Validación de Modelos

Para evaluar la eficacia de los modelos de predicción, es fundamental contar con un mecanismo que permita medir su capacidad de generalización. Para ello, se ha llevado a cabo una separación de los datos en dos conjuntos:

- **Conjunto de entrenamiento:** Incluye la mayoría de las observaciones históricas y se utiliza para ajustar los modelos.
- **Conjunto de prueba:** Compuesto por las últimas 10 observaciones, reservado exclusivamente para la validación del modelo.

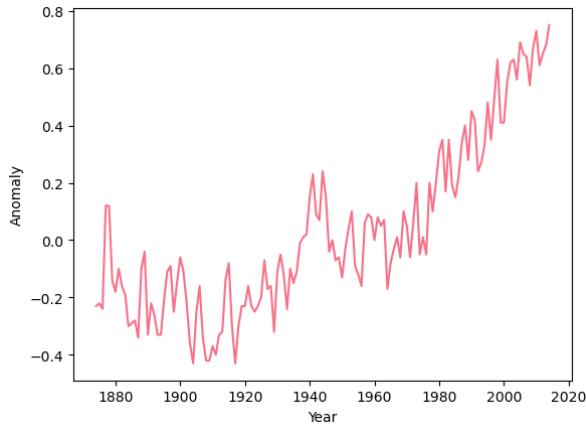
```
1 data_TR = data.iloc[:-10]
2 data_TST = data.iloc[-10:]
```

La selección de este subconjunto de prueba responde a la necesidad de evaluar cómo el modelo se desempeña en datos no utilizados durante el entrenamiento. Estas 10 observaciones representan los valores reales con los que se compararán las predicciones generadas, permitiendo así calcular métricas de error y determinar la precisión del modelo.

Este procedimiento es crucial para validar la capacidad del modelo de capturar patrones temporales y realizar proyecciones fiables. Además, al reservar un segmento final de la serie, se simula un escenario real en el que el modelo debe prever valores futuros sin disponer de información adicional.

4

Ajuste de un Modelo de Suavizado Exponencial



Observo que la serie presenta una tendencia ascendente clara, pero no detecto un patrón estacional recurrente, por lo que la mejor opción es aplicar un alisado exponencial doble de Holt. Este método me permitirá modelar la tendencia de la serie y proyectarla en el futuro sin necesidad de un componente estacional, ya que no hay variaciones cíclicas evidentes en períodos fijos. Dado que el modelo de Holt captura la tendencia de manera efectiva, lo utilizaré para realizar predicciones sobre la evolución de las anomalías de temperatura en los próximos años.

La ecuación general del modelo es:

$$x_t = L_t + b_t t + z_t$$

donde L_t representa el nivel suavizado y b_t la tendencia.

De la serie original, se obtiene la serie suavizada mediante las siguientes ecuaciones:

Estimación de la tendencia:

$$L_t = \alpha x_t + (1 - \alpha)(L_{t-1} + b_{t-1}), \quad L_1 = x_1$$

Estimación de la pendiente:

$$b_t = \beta(L_t - L_{t-1}) + (1 - \beta)b_{t-1}, \quad b_1 = x_2 - x_1$$

Ecuación de predicción: Para estimar valores futuros, suponiendo que hemos observado hasta el tiempo n , la predicción se define como:

$$\hat{X}_{n+m} = L_n + b_n m, \quad m \geq 1$$

donde m representa el número de períodos hacia el futuro.

```

1 modelo_holt = ExponentialSmoothing(data_TR["Anomaly"], trend="add", seasonal=None).fit()
2 predicciones_holt = modelo_holt.forecast(steps=10)
3 modelo_holt.summary()

```

| ExponentialSmoothing Model Results | | | |
|------------------------------------|----------------------|-------------------|------------------|
| Dep. Variable: | Anomaly | No. Observations: | 141 |
| Model: | ExponentialSmoothing | SSE | 1.646 |
| Optimized: | True | AIC | -619.531 |
| Trend: | Additive | BIC | -607.736 |
| Seasonal: | None | AICC | -618.904 |
| Seasonal Periods: | None | Date: | Wed, 12 Feb 2025 |
| Box-Cox: | False | Time: | 20:11:18 |
| Box-Cox Coeff.: | None | | |
| | coeff | code | optimized |
| smoothing_level | 0.4363502 | alpha | True |
| smoothing_trend | 0.0000000 | beta | True |
| initial_level | -0.1965962 | l0 | True |
| initial_trend | 0.0064356 | b0 | True |

El suavizado de nivel (smoothing_level = 0.436) indica que el modelo otorga un peso moderado a las observaciones recientes, lo que significa que está capturando cambios en los datos sin reaccionar demasiado rápido. El suavizado de tendencia (smoothing_trend = 0.0000) sugiere que el modelo no está incorporando una tendencia significativa, lo que puede ser una limitación pues la serie tiene una tendencia real. Los criterios de selección de modelos (AIC = -619.531 y BIC = -607.736) indican que el modelo tiene un buen ajuste relativo. El SSE es 1.646, lo que indica un error relativamente alto y, de hecho, indica que el modelo tiene un ajuste poco preciso en relación con la variabilidad de los datos.

Somos conscientes de las limitaciones del modelo de Holt aplicado a la serie original. Por esta razón, procedemos a aplicar el modelo sobre la serie diferenciada, con el objetivo de mejorar la estacionariedad y permitir que el método capture de manera más efectiva los patrones subyacentes de la serie temporal.

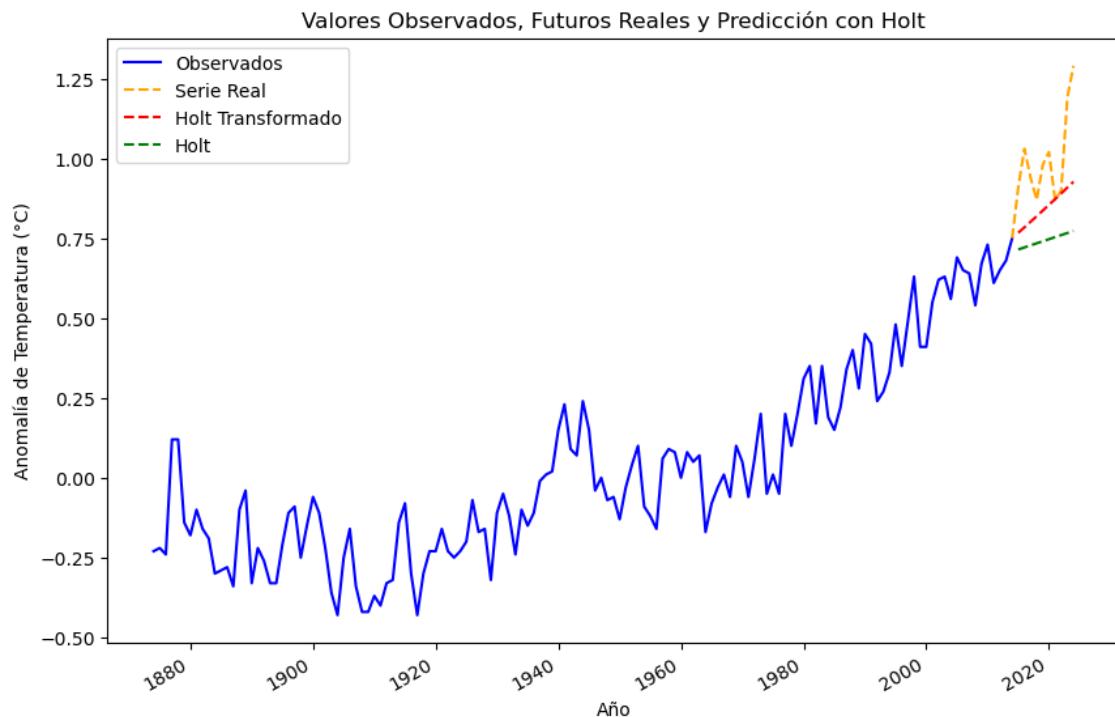
La diferenciación (aplicada en el apartado anterior) funciona de la siguiente manera. Asumimos que la tendencia en el instante t es muy próxima a la tendencia en el instante $t - 1$, y construimos una nueva serie definida como:

$$z_t = x_t - x_{t-1}$$

```

1 data_diff = data["Anomaly"].diff().dropna()
2 data_diff_TR = data_diff.iloc[:-10]
3
4 modelo_holt_diff = ExponentialSmoothing(data_diff_TR, trend="add", seasonal=None).fit()
5 pred_diff = modelo_holt_diff.forecast(steps=10)
6
7 # Reconstruir la serie original sumando las diferencias acumuladas al último valor conocido
8 ultimo_valor_original = data_TR["Anomaly"].iloc[-1] # Último valor antes de la predicción
9 pred_final = ultimo_valor_original + pred_diff.cumsum() # Revertir la diferenciación
10
11 # Asegurar que las fechas coincidan con el conjunto de prueba
12 pred_final.index = data_TST.index
13
14 plt.figure(figsize=(10, 6))
15 plt.plot(data_TR.index, data_TR["Anomaly"], label='Observados', linestyle='--', color='blue')
16 plt.plot(data.iloc[-11:-1].index, data.iloc[-11:-1]["Anomaly"], label='Serie Real', linestyle='--',
17           color='orange')
18 plt.plot(pred_final.index, pred_final, label='Holt Transformado', linestyle='--', color='red')
19 plt.plot(predicciones_holt.index, predicciones_holt, label='Holt', linestyle='--', color='green')
20
21 plt.xlabel("Año")
22 plt.ylabel("Anomalía de Temperatura (°C)")
23 plt.title("Valores Observados, Futuros Reales y Predicción con Holt")
24 plt.legend()
25 plt.xticks(rotation=30, ha='right')
26 plt.show()

```



Análisis del Modelo Holt: La predicción mantiene la tendencia creciente observada en los datos. Se observa que la predicción no captura completamente la variabilidad de la serie real, lo que indica que el modelo podría mejorarse ajustando los parámetros o probando otro método como ARIMA.

Análisis del Modelo Holt Transformado: La predicción del modelo Holt aplicado sobre la serie diferenciada sigue la tendencia general de la serie, pero presenta una mayor variabilidad en comparación con la versión estándar del modelo. Esto sugiere que la diferenciación ha permitido capturar mejor los cambios en la serie, aunque aún se observan desviaciones respecto a los valores reales.

Si bien el modelo transformado reduce parte de la suavización excesiva del modelo original, sigue sin reflejar completamente la estructura de los datos observados, especialmente en las fluctuaciones a corto plazo. La diferencia entre la predicción y la serie real podría indicar la necesidad de ajustar aún más los parámetros de suavizado o explorar métodos más avanzados, como ARIMA, que podrían modelar mejor la dinámica de los datos diferenciados.

5

Análisis de Autocorrelación y Selección del Modelo ARIMA

Vamos a comenzar haciendo una diferenciación de la serie, como antes.

```
1 data_diff_1 = data.diff(1).dropna()
2 print(f'ADF_Statistic:{adfuller(data)[0]},p-value:{adfuller(data)[1]}')
3 print(f'KPSS_Statistic:{kpss(data)[0]},p-value:{kpss(data)[1]}')
4 print(f'ADF_Statistic:{adfuller(data_diff_1)[0]},p-value:{adfuller(data_diff_1)[1]}')
5 print(f'KPSS_Statistic:{kpss(data_diff_1)[0]},p-value:{kpss(data_diff_1)[1]}')
```

Output[]: Test estacionariedad serie original

ADF Statistic: 1.9020335186272215, p-value: 0.9985312835501976

KPSS Statistic: 1.484156626569767, p-value: 0.01

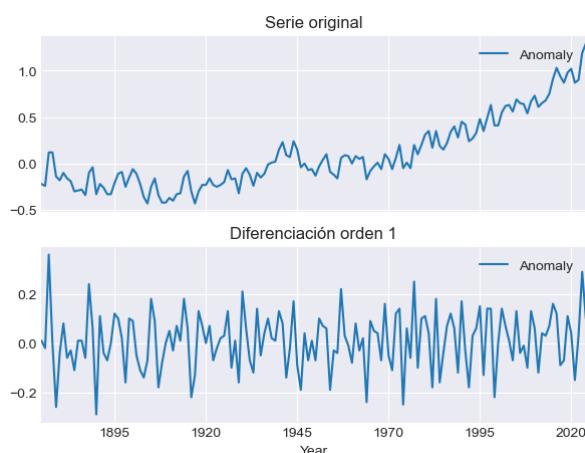
Test estacionariedad serie diferenciada de orden 1

ADF Statistic: -7.466760512058273, p-value: 5.1800024309746794e-11

KPSS Statistic: 0.4038286007299803, p-value: 0.07550491347845678

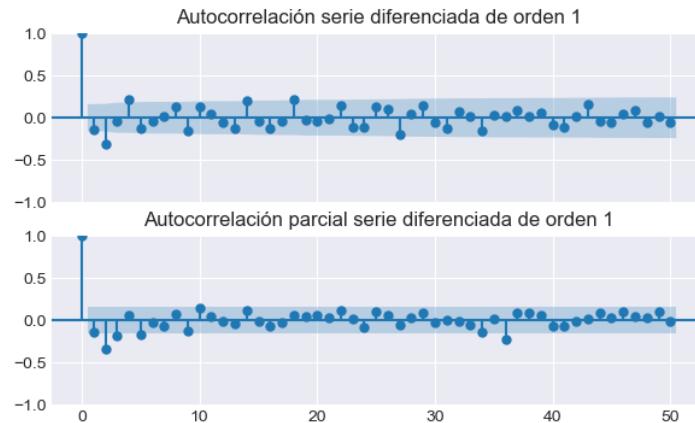
La serie original no es estacionaria, como lo indican los resultados de las pruebas ADF y KPSS: el test ADF arroja un valor p de 0.9985, muy superior al umbral de 0.05, lo que impide rechazar la hipótesis nula de que la serie tiene una raíz unitaria, mientras que el test KPSS, con un valor p de 0.01, sugiere que la serie es no estacionaria al rechazar su hipótesis nula de estacionariedad. Sin embargo, al aplicar una diferencia de primer orden, la serie se vuelve estacionaria, ya que el test ADF muestra un estadístico de -7.47 y un valor p prácticamente cero (5.18e-11), lo que permite rechazar la hipótesis nula y confirmar estacionariedad, mientras que el test KPSS, con un valor p de 0.0755, respalda aún más esta conclusión al no rechazar la hipótesis nula de estacionariedad.

```
1 fig, axs = plt.subplots(nrows=2, ncols=1,
2                         figsize=(7, 5), sharex=True)
3 data.plot(ax=axs[0], title='Serie_original')
3 data_diff_1.plot(ax=axs[1], title='Diferenciación_orden_1')
```



Calculamos el gráfico de autocorrelación de la serie original y la serie diferenciada:

```
1 fig, axs = plt.subplots(nrows=2, ncols=1, figsize=(7, 4), sharex=True)
2 plot_acf(data_diff_1, ax=axs[0], lags=50, alpha=0.05)
3 axs[0].set_title('Autocorrelación serie diferenciada de orden 1')
4 plot_pacf(data_diff_1, ax=axs[1], lags=50, alpha=0.05)
5 axs[1].set_title('Autocorrelación parcial serie diferenciada de orden 1')
```



El gráfico de autocorrelación (ACF) de la serie diferenciada de orden 1 muestra que solo el primer rezago tiene una autocorrelación significativa, mientras que los rezagos posteriores oscilan alrededor de cero dentro de las bandas de confianza, lo que indica la ausencia de una estructura de dependencia a largo plazo y sugiere que la serie diferenciada es estacionaria. Por otro lado, el gráfico de autocorrelación parcial (PACF) muestra un pico significativo en el primer rezago y una caída rápida en los siguientes, lo que es característico de un proceso autorregresivo de primer orden (AR(1)). Dado que en la PACF solo el primer rezago es significativo antes de disminuir rápidamente, podemos concluir que el componente autorregresivo (AR) tiene un orden $p=1$. En la ACF, la rápida disminución sugiere que el componente de media móvil (MA) es de orden $q=0$, ya que no hay un patrón claro de decaimiento lento o picos alternos. Por lo tanto, el modelo ARMA que mejor describe esta serie diferenciada es ARIMA(1,1,0), indicando que la serie original era integrada de orden 1 y que un modelo autorregresivo de primer orden es adecuado tras la diferenciación.

```
1 warnings.filterwarnings("ignore", category=UserWarning, message='Non-invertible|Non-
stationary')
2 modelo = ARIMA(order = (1, 1, 0))
3 modelo_res = modelo.fit(data_TR)
4 warnings.filterwarnings("default")
5 modelo_res.summary()
```

| SARIMAX Results | | | | | | |
|-------------------------|------------------|-------------------|--|-------|--------|--------|
| Dep. Variable: | y | No. Observations: | 141 <th data-cs="3" data-kind="parent"></th> <th data-kind="ghost"></th> <th data-kind="ghost"></th> | | | |
| Model: | SARIMAX(1, 1, 0) | Log Likelihood | 106.459 | | | |
| Date: | Mon, 17 Feb 2025 | AIC | -206.917 | | | |
| Time: | 08:42:42 | BIC | -198.092 | | | |
| Sample: | 01-01-1874 | HQIC | -203.331 | | | |
| | - 01-01-2014 | | | | | |
| Covariance Type: | opg | | | | | |
| | coef | std err | z | P> z | [0.025 | 0.975] |
| intercept | 0.0082 | 0.010 | 0.852 | 0.394 | -0.011 | 0.027 |
| ar.L1 | -0.1828 | 0.101 | -1.818 | 0.069 | -0.380 | 0.014 |
| sigma2 | 0.0128 | 0.002 | 7.959 | 0.000 | 0.010 | 0.016 |
| Ljung-Box (L1) (Q): | 0.60 | Jarque-Bera (JB): | | 0.53 | | |
| Prob(Q): | 0.44 | Prob(JB): | | 0.77 | | |
| Heteroskedasticity (H): | 0.80 | Skew: | | -0.14 | | |
| Prob(H) (two-sided): | 0.46 | Kurtosis: | | 2.91 | | |

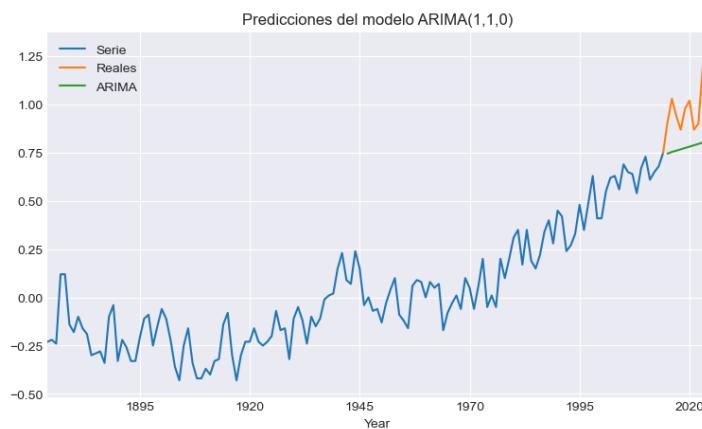
La salida muestra el resumen del modelo SARIMAX(1,1,0), que es equivalente a ARIMA(1,1,0), ajustado a la serie temporal con 141 observaciones. El criterio de información de Akaike (AIC = -206.917) y el criterio bayesiano de Schwarz (BIC = -198.092) indican la calidad del modelo, siendo un AIC más bajo mejor para la selección del modelo. Los coeficientes estimados incluyen un intercepto de 0.0082, que no es significativo ($p=0.394$), y un coeficiente AR(1) de -0.1828, que se acerca a ser significativo ($p=0.069$). El parámetro de varianza de los residuos ($\sigma^2=0.0128$) es altamente significativo ($p<0.001$), lo que indica que los residuos tienen una varianza bien definida.

Las pruebas de diagnóstico en la parte final del informe evalúan la idoneidad del modelo. La prueba de Ljung-Box para el primer rezago (L1) tiene un estadístico Q=0.60 con un valor p=0.44, lo que indica que no hay autocorrelación significativa en los residuos, sugiriendo que el modelo ha capturado bien la estructura de la serie. La prueba de Jarque-Bera (JB=0.53, p=0.77) no rechaza la hipótesis de normalidad de los residuos, lo que sugiere que están normalmente distribuidos. Además, la prueba de heterocedasticidad (H=0.80, p=0.46) no encuentra evidencia de varianza no constante en los residuos. En general, estos resultados indican que el modelo ARIMA(1,1,0) se ajusta bien a los datos, sus residuos parecen no estar autocorrelacionados ni sesgados, y no hay signos claros de problemas en la varianza.

```

1 # Predicción
2 predicciones_pmdarima = modelo_res.predict(n_periods=10)
3 fig, ax = plt.subplots(figsize=(9, 5))
4 data_TR.plot(ax=ax, label='Serie')
5 data.iloc[-11:].plot(ax=ax, label='Reales')
6 predicciones_pmdarima.plot(ax=ax, label='ARIMA')
7 ax.set_title('Predicciones del modelo ARIMA(1,1,0)')
8 ax.legend(['Serie', 'Reales', 'ARIMA'])

```



```

1 # Mean Squared Error (MSE)
2 mse = mean_squared_error(data_TST, predicciones_pmdarima)
3 # Root Mean Squared Error (RMSE)
4 rmse = np.sqrt(mse)
5 # Mean Absolute Error (MAE)
6 mae = mean_absolute_error(data_TST, predicciones_pmdarima)

```

Output[]:

Mean Squared Error (MSE): 0.06428515601975546
Root Mean Squared Error (RMSE): 0.2535451755008473
Mean Absolute Error (MAE): 0.2218251288498833

Ahora vamos a emplear Auto-ARIMA, un método automatizado que selecciona los parámetros óptimos(p,d,q) del modelo ARIMA, minimizando criterios de información como AIC y BIC. Veamos si podemos obtener un mejor ajuste a la serie temporal que con el modelo ARIMA anterior.

```

1 # Auto arima: selección basada en AIC
2 modelo = auto_arima(y = data_TR, start_p = 0, start_q = 0, max_p = 3, max_q = 3, seasonal =
   False, test = 'adf', d = None, trace = True, error_action = 'ignore', suppress_warnings =
   True, stepwise = True )

```

Output[]: Best model: ARIMA(1,1,3)(0,0,0)[0]

```

1 # Modelo ARIMA con statsmodels.arima
2 warnings.filterwarnings("ignore", category=UserWarning, message='Non-invertible|Non-
   stationary')
3 modelo_final = ARIMA(order = (1,1,3))
4 modelo_final_res = modelo_final.fit(data_TR)
5 warnings.filterwarnings("default")
6 modelo_final_res.summary()

```

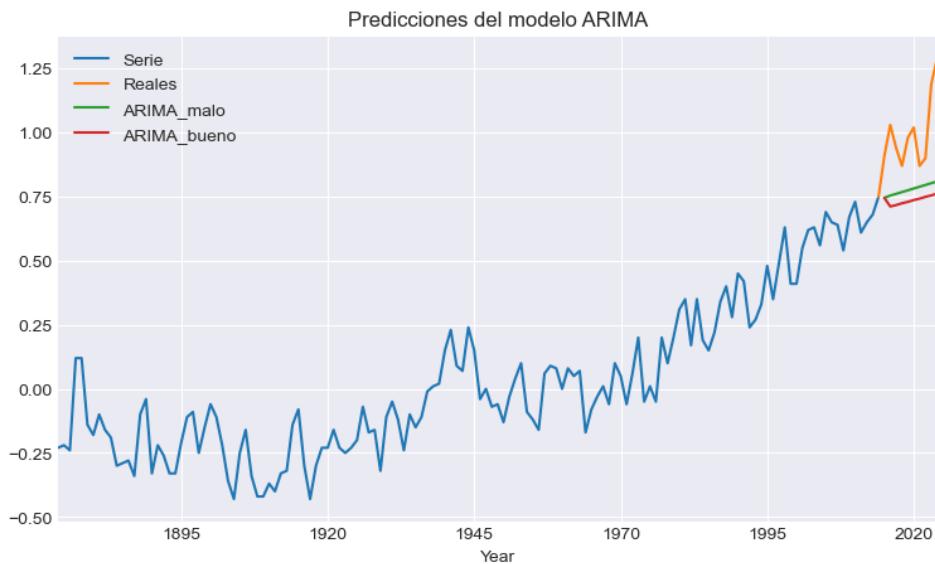
| SARIMAX Results | | | | | | | |
|-------------------------|-------------------------|-------------------|----------|-------|--------|--------|--|
| Dep. Variable: | y | No. Observations: | 141 | | | | |
| Model: | SARIMAX(1, 1, 3) | Log Likelihood: | 122.818 | | | | |
| Date: | Mon, 17 Feb 2025 | AIC: | -233.637 | | | | |
| Time: | 23:05:44 | BIC: | -215.987 | | | | |
| Sample: | 01-01-1874 - 01-01-2014 | HQIC: | -226.464 | | | | |
| Covariance Type: | opg | | | | | | |
| | coef | std err | z | P> z | [0.025 | 0.975] | |
| intercept | 0.0122 | 0.006 | 2.077 | 0.038 | 0.001 | 0.024 | |
| ar.L1 | -0.9496 | 0.052 | -18.292 | 0.000 | -1.051 | -0.848 | |
| ma.L1 | 0.6745 | 0.093 | 7.278 | 0.000 | 0.493 | 0.856 | |
| ma.L2 | -0.6147 | 0.089 | -6.910 | 0.000 | -0.789 | -0.440 | |
| ma.L3 | -0.4115 | 0.081 | -5.076 | 0.000 | -0.570 | -0.253 | |
| sigma2 | 0.0101 | 0.001 | 7.822 | 0.000 | 0.008 | 0.013 | |
| Ljung-Box (L1) (Q): | 0.11 | Jarque-Bera (JB): | 0.96 | | | | |
| Prob(Q): | 0.74 | Prob(JB): | 0.62 | | | | |
| Heteroskedasticity (H): | 0.88 | Skew: | 0.18 | | | | |
| Prob(H) (two-sided): | 0.66 | Kurtosis: | 2.82 | | | | |

El modelo ARIMA(1,1,3) parece bien ajustado, con coeficientes significativos, ausencia de autocorrelación en los residuos y sin problemas de heterocedasticidad. La validación con Ljung-Box y Jarque-Bera respalda que los residuos son esencialmente ruido blanco. Dado el bajo AIC, este modelo es una buena opción para realizar predicciones confiables.

```

1 # Predicción
2 predicciones_pmdarima = predicciones_pmdarima_fin = modelo_final_res.predict(n_periods=10)
3 fig, ax = plt.subplots(figsize=(9, 5))
4 data_TR.plot(ax=ax, label='Serie')
5 data.iloc[-11:].plot(ax=ax, label='Reales')
6 predicciones_pmdarima.plot(ax=ax, label='ARIMA_malo')
7 predicciones_pmdarima_fin.plot(ax=ax, label='ARIMA_bueno')
8 ax.set_title('Predicciones del modelo ARIMA')
9 ax.legend(['Serie', 'Reales', 'ARIMA_malo', 'ARIMA_bueno'])

```



```

1 # Mean Squared Error (MSE)
2 mse = mean_squared_error(data_TST, predicciones_pmdarima_fin)
3 # Root Mean Squared Error (RMSE)
4 rmse = np.sqrt(mse)
5 # Mean Absolute Error (MAE)
6 mae = mean_absolute_error(data_TST, predicciones_pmdarima_fin)

```

Output[]:

Mean Squared Error (MSE): 0.08499075650353012
Root Mean Squared Error (RMSE): 0.29153174184560093
Mean Absolute Error (MAE): 0.26277130040734215

6

Expresión Algebraica del Modelo Ajustado

El modelo ajustado en este análisis es un **ARIMA(1,1,3)**, cuya expresión algebraica se define como:

$$(1 - \phi_1 B)(1 - B)X_t = (1 + \theta_1 B + \theta_2 B^2 + \theta_3 B^3)Z_t \quad (6.1)$$

Donde:

- X_t representa la serie temporal de anomalías de temperatura oceánica.
- B es el operador de rezago, es decir, $BX_t = X_{t-1}$.
- ϕ_1 es el coeficiente del término autorregresivo (AR) de primer orden.
- $\theta_1, \theta_2, \theta_3$ son los coeficientes del componente de medias móviles (MA) de órdenes 1, 2 y 3, respectivamente.
- Z_t representa un término de error blanco, con media cero y varianza constante.

Sustituyendo los coeficientes estimados en la ecuación, obtenemos:

$$(1 + 0.9496B)(1 - B)X_t = (1 + 0.6745B - 0.6147B^2 - 0.4115B^3)Z_t \quad (6.2)$$

Componentes del Modelo:

- **Parte autorregresiva (AR):** $(1 - \phi_1 B)$, donde ϕ_1 es el coeficiente autorregresivo que mide la influencia de los valores pasados en la serie.
- **Parte de diferenciación:** $(1 - B)$, que se encarga de eliminar la tendencia para hacer la serie estacionaria.
- **Parte de medias móviles (MA):** $(1 - \theta_1 B - \theta_2 B^2 - \theta_3 B^3)$, donde $\theta_1, \theta_2, \theta_3$ representan los coeficientes de los choques pasados en la serie.

Por tanto, este modelo describe la evolución de las anomalías de temperatura considerando tanto el impacto de valores pasados (componente AR) como la influencia de choques aleatorios previos (componente MA).

7

Predicciones e Intervalos de Confianza

A partir del modelo ajustado **ARIMA(1,1,3)** con `pmdarima`, se generan predicciones para los siguientes 10 períodos futuros. Para cuantificar la incertidumbre de las predicciones, se calculan los intervalos de confianza al 95%.

Para obtener las predicciones y sus intervalos con `pmdarima`, utilizamos:

```

1 pred, conf_int = modelo_final_res.predict(n_periods=10, return_conf_int=True)
2 pred_index = pd.date_range(start=data_TR.index[-1] + pd.DateOffset(years=1), periods=10, freq
   ='A') # Ajusta la frecuencia según la serie
3 pred_series = pd.Series(pred, index=pred_index)
4 conf_lower = conf_int[:, 0] # Límite inferior
5 conf_upper = conf_int[:, 1] # Límite superior

```

Donde:

- \hat{X}_{t+h} representa la predicción del modelo ARIMA.
- `conf_int` contiene los límites inferior y superior del intervalo de confianza al 95%.

El intervalo de confianza se define como:

$$\hat{X}_{t+h} \pm z_{\alpha/2} \cdot \sigma_{\hat{X}_{t+h}} \quad (7.1)$$

Donde:

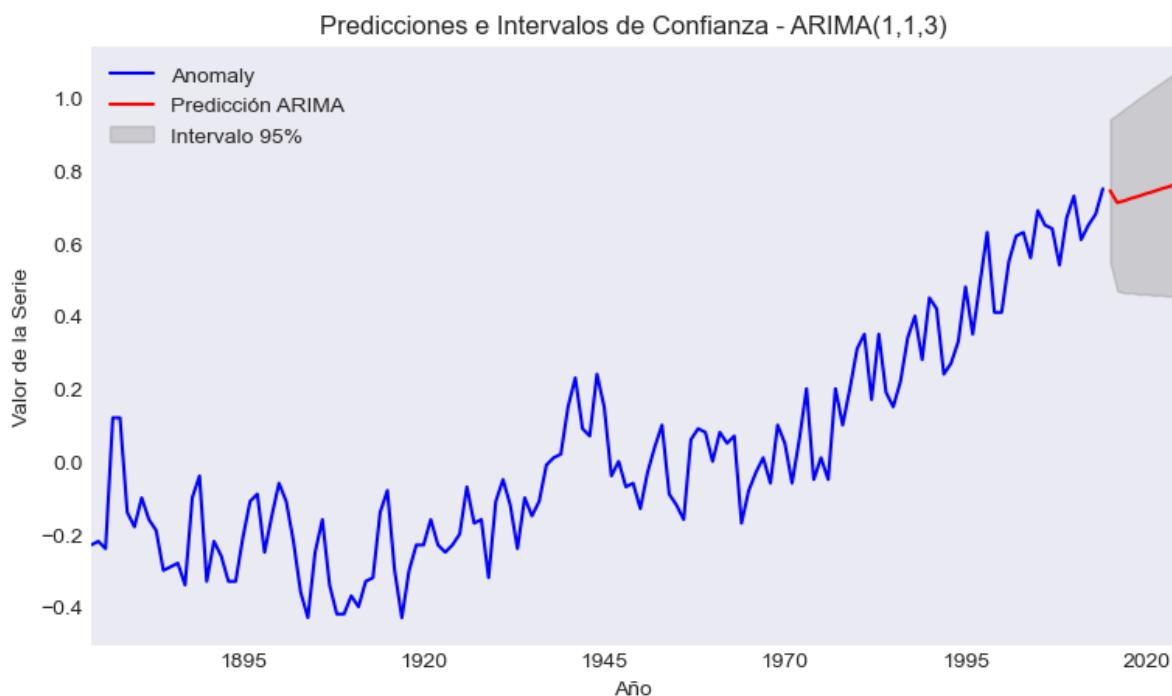
- $z_{\alpha/2}$ es el valor crítico de la distribución normal estándar ($z_{0.025} \approx 1.96$ para un 95% de confianza).
- $\sigma_{\hat{X}_{t+h}}$ es la desviación estándar de la predicción en el horizonte h .

El siguiente gráfico muestra las predicciones junto con los intervalos de confianza generados con `pmdarima`:

```

1 fig, ax = plt.subplots(figsize=(9, 5))
2 data_TR.plot(ax=ax, label='Serie_Histórica', color='blue')
3 pred.plot(ax=ax, label='Predicción_ARIMA', color='red')
4 ax.fill_between(pred_index, conf_lower, conf_upper, color='gray', alpha=0.3, label='Intervalo
   _95%')
5 ax.set_title('Predicciones_Intervalos_Confianza_ARIMA(1,1,3)')
6 ax.legend()
7 plt.xlabel('Año')
8 plt.ylabel('Valor_de_la_Serie')
9 plt.grid()
10 plt.show()

```



Los resultados muestran que la predicción se ajusta bien en el corto plazo, pero la incertidumbre crece conforme se incrementa el horizonte de predicción, lo que es esperado en modelos ARIMA.

8

Comparación de Predicciones y Conclusiones

Para evaluar la calidad de los modelos predictivos utilizados, se han comparado las predicciones generadas por el modelo de suavizado exponencial de Holt y el modelo ARIMA. Ambos modelos fueron entrenados con la serie temporal de anomalías de temperatura oceánica y validados utilizando los últimos 10 valores de la serie.

El modelo de Holt es un método de suavizado exponencial diseñado para capturar tendencias a lo largo del tiempo. En este caso, se ha utilizado una versión con tendencia aditiva, dado que no se observa una estacionalidad evidente en la serie. Sin embargo, a pesar de ser un modelo simple y eficiente, el suavizado exponencial tiende a generar predicciones lineales, lo que puede limitar su capacidad para capturar patrones complejos o cambios abruptos en la serie.

Por otro lado, el modelo **ARIMA(1,1,3)** ha demostrado ser una opción más robusta para la predicción de la serie temporal, ya que incorpora tanto la tendencia como la autocorrelación entre los valores pasados. En la comparación gráfica de las predicciones, se observa que ARIMA proporciona una aproximación más precisa a los valores reales, con una menor desviación en comparación con Holt.

Limitaciones del Modelo Aditivo Las predicciones generadas por ARIMA presentan una forma relativamente recta debido a la naturaleza del modelo utilizado. Al aplicar un modelo aditivo, los componentes de la serie (tendencia y error) se combinan de manera lineal, lo que puede resultar en una predicción con menor variabilidad. Para mejorar este comportamiento, se intentó aplicar una transformación logarítmica con desplazamiento (Shift + Log) para permitir la aplicación de un modelo multiplicativo. Sin embargo, esta transformación resultó inefectiva, ya que tras la diferenciación de primer orden, volvieron a aparecer valores negativos en la serie, lo que impide la aplicación del modelo multiplicativo.

Posibles Mejoras: Incorporación de Estacionalidad Aunque en el análisis actual no se ha encontrado una estacionalidad clara, es posible que al extender la serie temporal con datos de muchos más años, se pueda detectar algún patrón estacional a largo plazo. En ese caso, la implementación de un modelo estacional, como un **SARIMA**, podría proporcionar una mejor captura de las dinámicas temporales presentes en la serie.

Además, otro enfoque para mejorar las predicciones sería la experimentación con modelos de Machine Learning o redes neuronales recurrentes (RNNs), que podrían capturar relaciones más complejas en la serie.

Conclusión En general, el modelo ARIMA ha mostrado un mejor desempeño que el modelo de Holt en términos de precisión de predicción, pero aún existen limitaciones que podrían ser abordadas en estudios futuros. La incorporación de datos adicionales y la exploración de modelos estacionales o no lineales podrían mejorar significativamente la capacidad predictiva del modelo.

Bibliografía

- [1] National Centers for Environmental Information (NCEI). *Climate at a Glance: Global Time Series*. <http://ncei.noaa.gov/access/monitoring/climate-at-a-glance/global/time-series/globe/tavg/ocean/ytd/12/1874-2024>. Accessed: 2025-01-23. National Oceanic and Atmospheric Administration (NOAA), n.d.