



Análisis de Series de Tiempo

Carrera de Especialización en Inteligencia Artificial

Cronograma

| | |
|----------------|---|
| Clase 1 | Introducción, nociones básicas de modelado, algunos ejemplos |
| Clase 2 | Tendencia determinística, modelos estacionarios AR, MA, ARMA, presentación de TP |
| Clase 3 | Estacionariedad, Estacionalidad, criterios de bondad, ARMA, ARIMA, SARIMA |
| Clase 4 | Caso de estudio, tendencia estocástica, estacionalidad, modelos SARIMA |
| Clase 5 | Predicciones, tratamiento de intervenciones y outliers |
| Clase 6 | Aplicaciones. Análisis espectral. Markov* |
| Clase 7 | Heteroscedasticidad. Frameworks. |
| Clase 8 | Presentaciones trabajo final |

Agenda

1. Estacionariedad
2. Estacionalidad
3. Modelos ARIMA y SARIMA
4. Criterios de bondad de modelos

ARMA(p,q)

Modelo ARMA

El modelo arma es una combinación de un proceso AR con un MA. Diremos que $\{Y_t\}$ sigue un modelo ARMA(p,q) si

$$Y_t = a_1 Y_{t-1} + \dots + a_p Y_{t-p} + e_t - b_1 e_{t-1} - \dots - b_q e_{t-q}$$

Si se satisfacen las condiciones de estacionariedad, el modelo ARMA(p,q) puede reescribirse como un proceso lineal general con coeficientes ψ_1, ψ_2, \dots dados por:

$$\begin{cases} \psi_0 = 1 \\ \psi_1 = -b_1 + a_1 \\ \psi_2 = -b_2 + a_2 + a_1 \psi_1 \\ \vdots \\ \psi_j = -b_j + a_p \psi_{j-p} + a_{p-1} \psi_{j-p+1} + \dots + a_1 \psi_{j-1} \end{cases} \quad \psi_j = 0, \text{ si } j < 0 \text{ y } b_j = 0 \text{ si } j > q$$

ARMA(p,q)

Se puede ver que la función de autocorrelación está dada por:

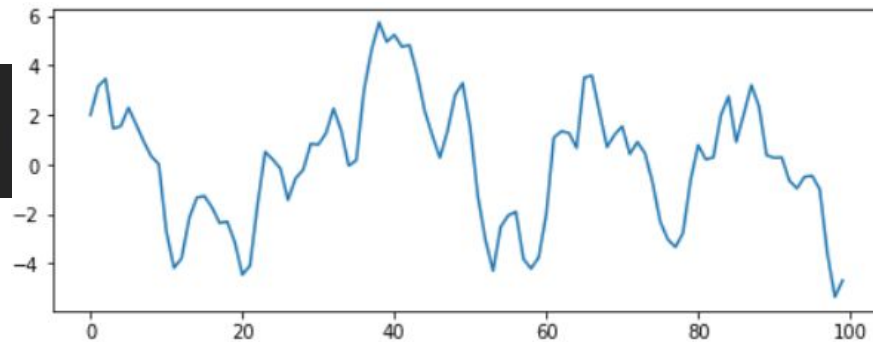
$$\begin{cases} C_0 = a_1 C_1 + a_2 C_2 + \dots - \sigma_e^2 (b_0 + b_1 \psi_1 + \dots + b_q \psi_q) \\ C_1 = a_1 C_0 + a_2 C_1 + \dots + a_p C_{p-1} - \sigma_e^2 (b_1 + b_2 \psi_1 + \dots + b_1 \psi_{q-1}) \\ \vdots \\ C_p = a_1 C_{p-1} + a_2 C_{p-2} + \dots + a_p C_0 - \sigma_e^2 (b_p + b_{p+1} \psi_1 + \dots + b_q \psi_{q-p}) \end{cases}$$

Si $k > q$ entonces la expresión puede simplificarse como:

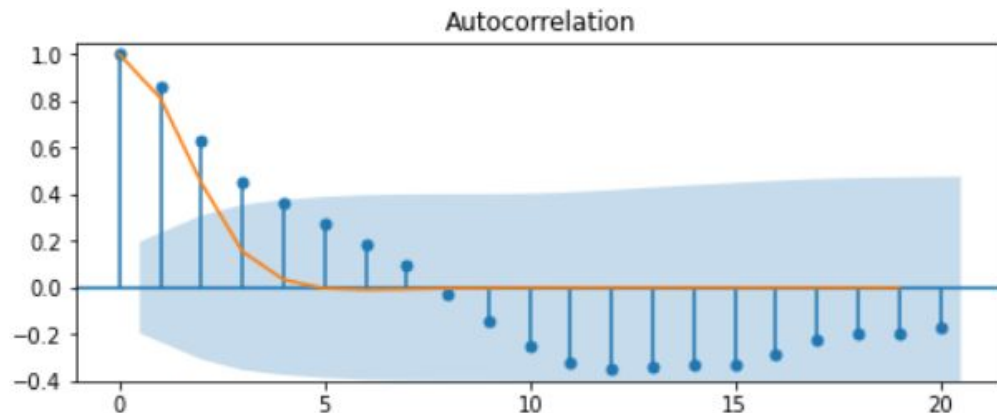
$$C_k = a_1 C_{k-1} + a_2 C_{k-2} + \dots + a_p C_{k-p}$$

Ejemplo!

```
arma_ts = arma_generate_sample(  
    ar=np.r_[1,-ar_coef], ma=np.r_[1,ma_coef],  
    nsample =100)
```



```
plot_acf(arma_ts, ax=ax)  
plt.plot(arma_acf(ar=np.r_[1, -ar_coef],  
    ma=np.r_[1,ma_coef], lags=20))
```

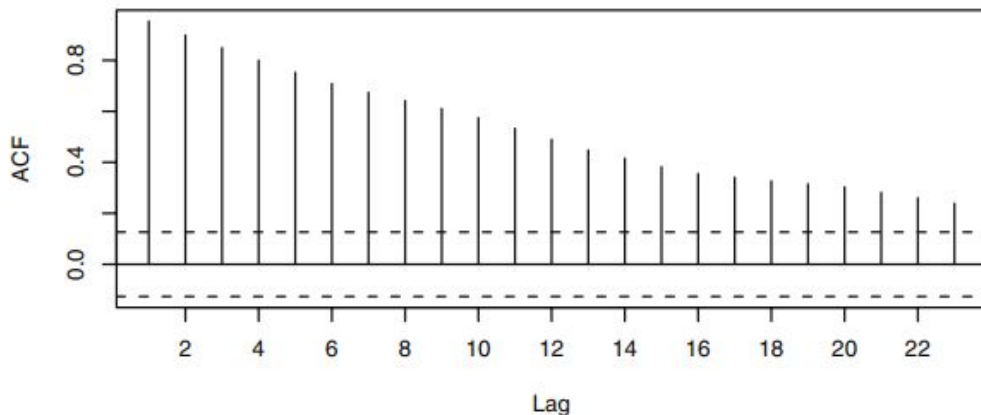


| Stationary Time Series | Non-Stationary Time Series |
|--|---|
| Statistical properties of a stationary time series are independent of the point in time where it is observed. | Statistical properties of a non-stationary time series is a function of time where it is observed. |
| Mean, variance and other statistics of a stationary time series remains constant. Hence, the conclusions from the analysis of stationary series is reliable. | Mean, variance and other statistics of a non-stationary time series changes with time. Hence, the conclusions from the analysis of a non-stationary series might be misleading. |
| A stationary time series always reverts to the long-term mean. | A non-stationary time series does not revert to the long term mean. |
| A stationary time series will not have trends, seasonality, etc. | Presence of trends, seasonality makes a series non-stationary. |

Estacionariedad

Autocorrelación

Un método relativamente fácil, aunque bastante a ojo, para verificar que una serie **no es estacionaria** es a través de su función de autocorrelación muestral.



Si la gráfica no alcanza valores nulos para lags grandes es porque posiblemente no sea estacionaria

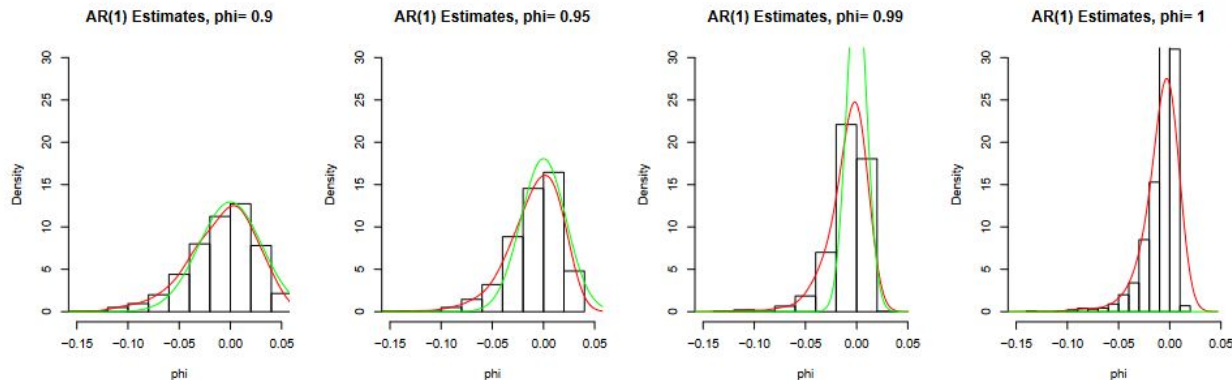
Test de Dickey-Fuller

Tengo un modelo AR(1) de la forma $Y_t = a_1 Y_{t-1} + e_t$ y quiero saber si $a_1 = 1$

Si desconocemos la estacionariedad del modelo, sólo podemos estimar el parámetro por OLS:

$$\hat{a}_1 = \frac{\sum_{t=1}^n Y_t Y_{t-1}}{\sum_{t=1}^n Y_{t-1}^2}$$

Se puede demostrar que $\sqrt{n}(\hat{a}_1 - a_1) \sim \mathcal{N}(0, 1 - a_1^2)$ si $|a_1| < 1$, mientras que si $a_1 = 1$ el estimador tiene una distribución asintótica de la forma



Test de Dickey-Fuller

Es un test para determinar si la serie es estacionaria (**a una diferenciación**)

Propone el modelo

$$Y_t = aY_{t-1} + \boxed{X_t} \leftarrow \begin{array}{l} \text{Proceso} \\ \text{estacionario} \end{array}$$

Si **a=1** → El proceso es **no estacionario** (hay caminante aleatorio)

Si miramos la serie diferenciada una vez, tenemos que

$$Y_t - Y_{t-1} = aY_{t-1} + W_t - Y_{t-1} = (a - 1)Y_{t-1} + W_t$$

Dickey y Fuller proponen entonces el test

$$H_0 : (a - 1) = 0 \quad vs. \quad H_1 : (a - 1) \neq 0$$

Su gran aporte fue hallar la distribución asintótica de $n(\widehat{a} - 1)$ bajo H_0 .

Test de dickey-Fuller Aumentado

Incorpora al modelo un término de ruido dependiente (pero estacionario)

$$Y_t = aY_{t-1} + X_t \quad X_t = \sum_{j=1}^p \rho_j X_{t-j} + \boxed{w_t} \leftarrow \text{Ruido Blanco}$$

Si tomamos la primera diferencia, y observamos que bajo H_0

$X_t = Y_t - Y_{t-1}$ tenemos que

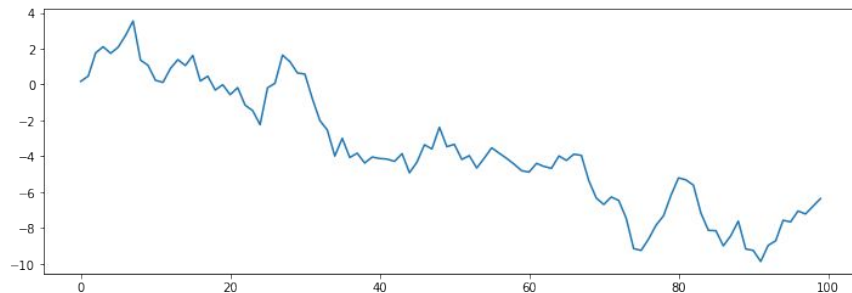
$$Y_t - Y_{t-1} = (a - 1)Y_{t-1} + \sum_{j=1}^p \rho_j (X_{t-j} - X_{t-j-1}) + W_t$$

Nuevamente se definen las hipótesis:

$$H_0 : (a - 1) = 0 \quad vs. \quad H_1 : (a - 1) \neq 0$$

p habría que estimarlo, pero podemos dejar que se encargue el software.

¿Cómo implementar en statsmodels?



```
from statsmodels.tsa.stattools import adfuller
adfuller(y)
```

✓ 0.9s

```
(-1.394757894883641, ← Estadístico
0.584786653359185, ← p-valor
4, ← # lags usados
95, ← # observaciones usadas
{'1%': -3.5011373281819504,
 '5%': -2.8924800524857854, ← umbrales
 '10%': -2.5832749307479226},
```

regression : {"c","ct","ctt","n"}

Constant and trend order to include in regression.

- "c" : constant only (default).
- "ct" : constant and trend.
- "ctt" : constant, and linear and quadratic trend.
- "n" : no constant, no trend.

Ejemplo con datos reales

```
inputfile = "../Datasets/TEC02.2000.2021.csv"

ts = pd.read_csv(inputfile, header=0, index_col=0, squeeze=True)
ts.fechaHora = pd.to_datetime(ts.fechaHora)
ts.fechaHora=pd.to_datetime(ts.fechaHora).dt.date
ts.fechaHora=pd.DatetimeIndex(ts.fechaHora)
ts=ts.sort_index(ascending=False)
```

Scripts/Dickey-Fuller_dataset_example.ipynb

```
adfuller(ts.ultimoPrecio)
(0.57890231841091,
 0.9870838971395222,
 32,
 4807,
 {'1%': -3.431711097447145,
  '5%': -2.862141452575749,
  '10%': -2.5670901548483767},
```

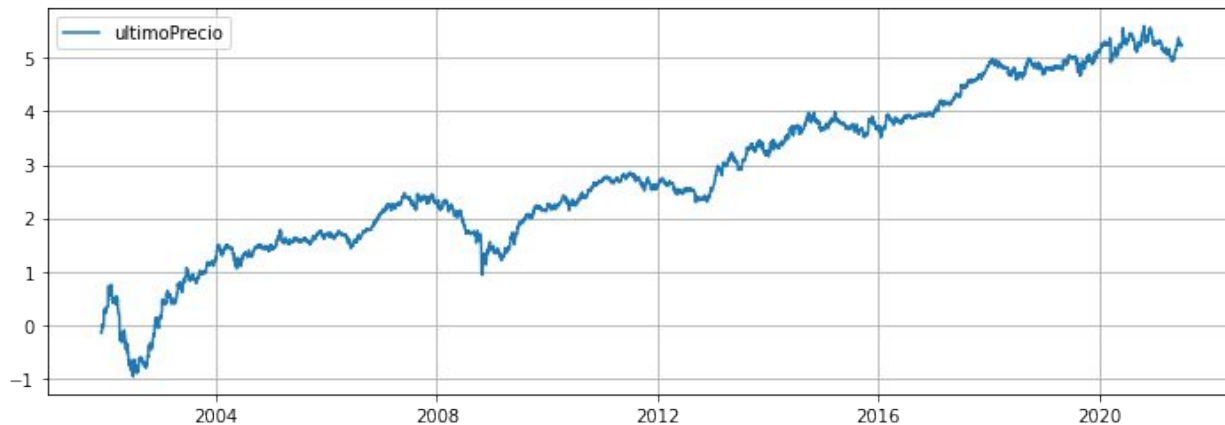


Ejemplo con datos reales

```
log_ultimoPrecio = np.log(ts.ultimoPrecio)
```

```
adfuller(log_ultimoPrecio, regression='ct')
```

```
(-3.037699968528972,  
 0.1218440464079108,  
 2,  
 4837,  
 {'1%': -3.9606428515465306,  
  '5%': -3.4113980566548285,  
  '10%': -3.127584714163724},
```

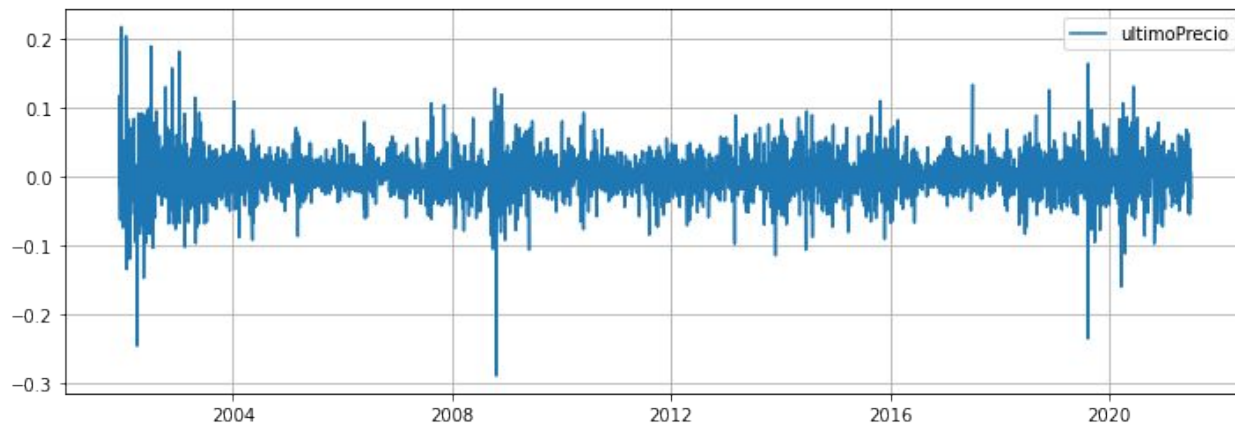


Ejemplo con datos reales

```
diff_log_ultimoPrecio = log_ultimoPrecio - log_ultimoPrecio.shift(1)
```

```
adfuller(diff_log_ultimoPrecio.dropna())
```

```
(-46.783753322231384,  
 0.0,  
 1,  
 4837,  
 {'1%': -3.4317026511738518,  
  '5%': -2.862137721117907,  
  '10%': -2.567088168437432},  
 -20357.65641243886)
```

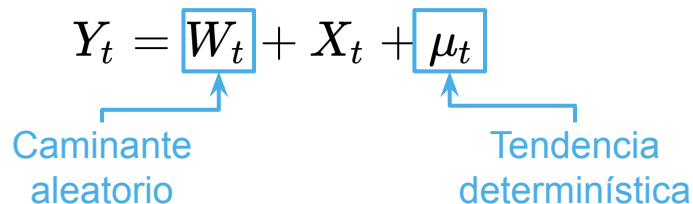


Test de Kwiatkowski–Phillips–Schmidt–Shin (KPSS)

Es un test para analizar si la serie es **estacionaria alrededor de una tendencia determinística**.

Plantea que la serie se puede descomponer como

$$Y_t = \boxed{W_t} + X_t + \boxed{\mu_t}$$



Caminante aleatorio Tendencia determinística

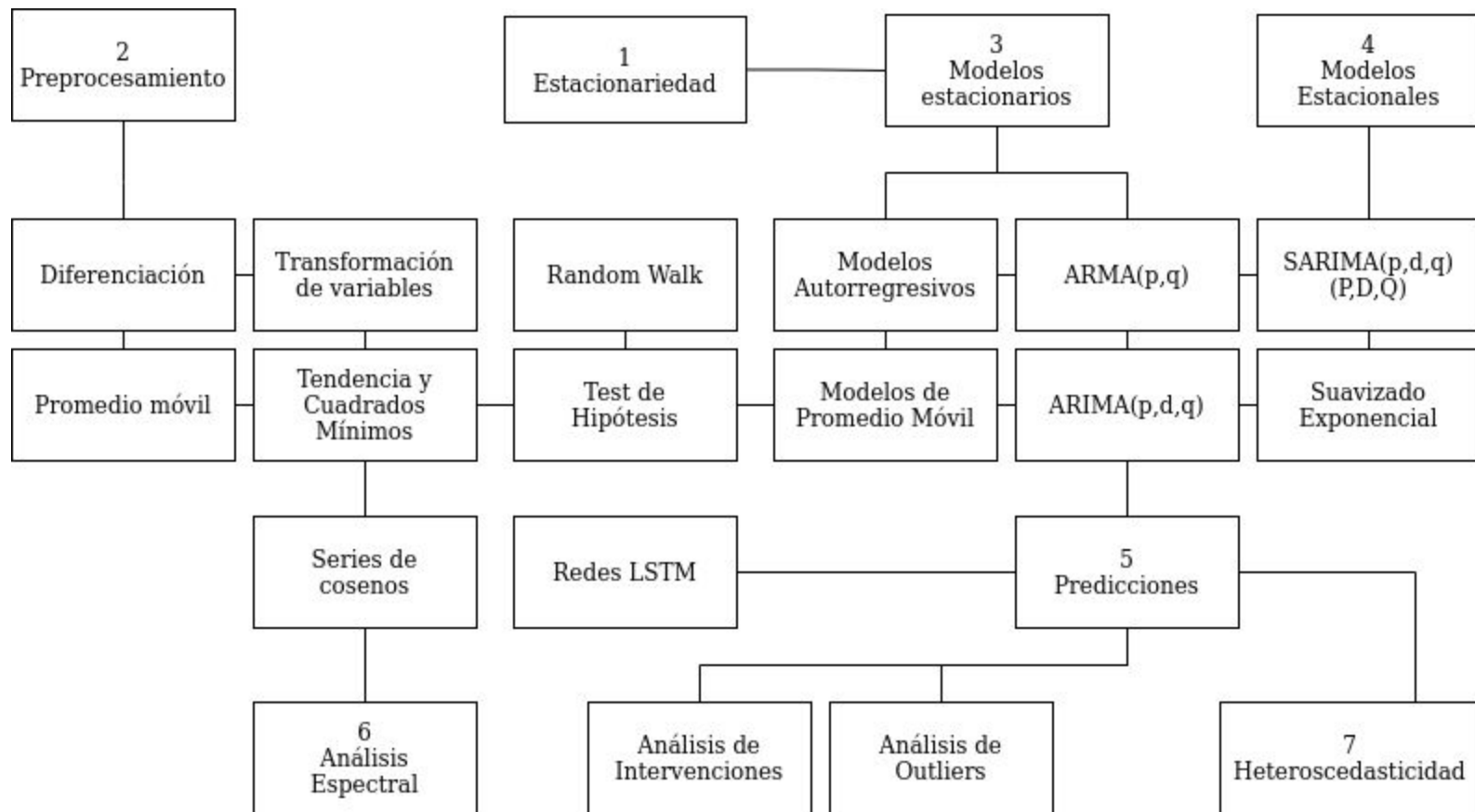
Para luego plantear las hipótesis: $H_0 : var(W_t) = 0$ $H_1 : var(W_t) > 0$

Observar que en este caso busco **no rechazar H_0** (p-valor>0.05)

Bibliografía extra para D-F y KPSS

- [Testing for unit roots](#)
- [Augmented Dickey-Fuller root tests](#)
- [Statsmodels: Stationarity and detrending](#)

ARIMA(p,d,q)



Operadores

- Operador de Backshift (B)
- Polinomio característico
- AR, MA compactos
- Abuso de notación para B
- Operador Nabla
- ARIMA compacto
- Ejemplo

AR, MA + Operador de Backshift (B)

MA(q):

$$\begin{aligned}Y_t &= e_t - b_1 e_{t-1} - \dots - b_q e_{t-q} \\&= e_t - b_1 B e_t - \dots b_q B^q e_t \\&= (1 - b_1 B - \dots - b_q B^q) e_t\end{aligned}$$

- **Proceso Lineal General:** es útil para encontrar propiedades generales, por ejemplo para condicionar los coeficientes del modelo

AR(p)

$$\begin{aligned}Y_t &= e_t + a_1 Y_{t-1} + \dots + a_p Y_{t-p} \\Y_t - a_1 Y_{t-1} - \dots - a_p Y_{t-p} &= e_t \\Y_t - a_1 B Y_t - \dots - a_p B^p Y_t &= e_t \\(1 - a_1 B - \dots - a_p B^p) Y_t &= e_t\end{aligned}$$

- **Polinomio característico:** es útil para expresar relaciones analíticas entre las raíces del polinomio y las expresiones de autocorrelación y autocovarianza del modelo.

$$BY_t = Y_{t-1}$$

$$B^2Y_t = Y_{t-2}$$

...

$$B^kY_t = Y_{t-k}$$

$$\nabla Y_t = Y_t - Y_{t-1}$$

$$\begin{aligned}\nabla^2 Y_t &= \nabla(\nabla Y_t) \\ &= \nabla(Y_t - Y_{t-1}) \\ &= Y_{t-1} - Y_{t-2}\end{aligned}$$

$$B^k Y_t = B_k Y_t$$

$$\begin{aligned}B_s^k Y_t &= B^{ks} Y_t \\ &= Y_{t-ks}\end{aligned}$$

$$B_{12}^k Y_t = Y_{t-k \cdot 12}$$

$$(1 - B)Y_t = Y_t - Y_{t-1}$$

$$\begin{aligned}(1 - B)^2 Y_t &= (1 - 2B + B^2)Y_t \\ &= Y_t - 2Y_{t-1} + Y_{t-2}\end{aligned}$$

$$\nabla_s^k Y_t = (1 - B_s)^k Y_t$$

AR(p)

$$e_t = Y_t - \sum_{k=1}^p a_k Y_{t-k}$$

$$e_t = (1 - \sum_{k=1}^p a_k B^k) Y_t$$

$$e_t = \phi(B) Y_t$$

MA(q)

$$Y_t = e_t - \sum_{k=1}^p b_k e_{t-k}$$

$$Y_t = (1 - \sum_{k=1}^p b_k B^k) e_t$$

$$Y_t = \theta(B) e_t$$

Modelo ARIMA

Diremos que $\{Y_t\}$ sigue un modelo ARIMA si $W_t = \nabla^d Y_t$ es un proceso ARMA(p,q) estacionario. Diremos en este caso que Y_t sigue un proceso ARIMA(p,d,q).

Operador de *backshift*: el operador de backshift (B) opera sobre el índice temporal de la serie de tiempo y la desplaza en una unidad de tiempo:

$$BY_t = Y_{t-1}$$

El operador B es lineal: $B(aY_t + bX_t + c) = aY_{t-1} + bX_{t-1} + c$

En general escribimos que $B^d Y_t = Y_{t-d}$

AR, MA + operador Backshift

MA(q):

$$\begin{aligned}Y_t &= e_t - b_1 e_{t-1} - \dots - b_q e_{t-q} \\&= e_t - b_1 B e_t - \dots b_q B^q e_t \Rightarrow Y_t = b(B) e_t \\&= (1 - b_1 B - \dots - b_q B^q) e_t\end{aligned}$$

ARMA(p,q)

$$a(B)Y_t = b(B)e_t$$

AR(p)

$$\begin{aligned}Y_t &= e_t + a_1 Y_{t-1} + \dots + a_p Y_{t-p} \\Y_t - a_1 Y_{t-1} - \dots - a_p Y_{t-p} &= e_t \\Y_t - a_1 B Y_t - \dots - a_p B^p Y_t &= e_t \Rightarrow a(B)Y_t = e_t \\(1 - a_1 B - \dots - a_p B^p)Y_t &= e_t\end{aligned}$$

ARIMA(p,d,q)

$$a(B)(1 - B)^d Y_t = b(B)e_t$$

ARMA, ARIMA + operator Backshift

ARMA(p,q)

$$Y_t = \theta(B)e_t$$

$$e_t = \phi(B)Y_t$$

$$\phi(B)Y_t = \theta(B)e_t$$

$$a(B)Y_t = b(B)e_t$$

ARIMA(p,d,q)

$$\nabla^d Y_t = W_t$$

$$W_t : ARMA(p, q)$$

$$a(B)(1 - B)^d Y_t = b(B)e_t$$

Ejemplos

https://github.com/charlieromano/TimeSeries/blob/main/Scripts/predicciones_AR_MA.py

Statsmodels

<https://www.statsmodels.org/devel/generated/statsmodels.tsa.arima.model.ARIMA.html>

Estacionalidad

Modelo SARMA multiplicativo

Cuando el proceso es estacionario podemos definir un modelo SARMA multiplicativo $\text{ARMA}(p,q)\times(P,Q)_s$, con período estacional s como un modelo AR con polinomio característico $a(x)\alpha(x)$ más un MA con polinomio característico $b(x)\beta(x)$. Con

$$a(x) = 1 - a_1x - a_2x^2 - \dots - a_px^p$$

$$\alpha(x) = 1 - \alpha_1x^s - \alpha_2x^{2s} - \dots - \alpha_Px^{Ps},$$

$$b(x) = 1 - b_1x - b_2x^2 - \dots - b_qx^q$$

$$\beta(x) = 1 - \beta_1x^s - \beta_2x^{2s} - \dots - \beta_Px^{Ps},$$

Modelo SARIMA multiplicativo

In general, then, we define a **multiplicative seasonal ARMA(p, q) \times (P, Q) $_s$ model with seasonal period s** as a model with AR characteristic polynomial $\phi(x)\Phi(x)$ and MA characteristic polynomial $\theta(x)\Theta(x)$, where

$$\left. \begin{aligned}\phi(x) &= 1 - \phi_1 x - \phi_2 x^2 - \dots - \phi_p x^p \\ \Phi(x) &= 1 - \Phi_1 x^s - \Phi_2 x^{2s} - \dots - \Phi_P x^{Ps}\end{aligned} \right\} \quad (10.2.6)$$

and

$$\left. \begin{aligned}\theta(x) &= 1 - \theta_1 x - \theta_2 x^2 - \dots - \theta_q x^q \\ \Theta(x) &= 1 - \Theta_1 x^s - \Theta_2 x^{2s} - \dots - \Theta_Q x^{Qs}\end{aligned} \right\} \quad (10.2.7)$$

The model may also contain a constant term θ_0 . Note once more that we have just a special ARMA model with AR order $p + Ps$ and MA order $q + Qs$, but the coefficients are not completely general, being determined by only $p + P + q + Q$ coefficients. If $s = 12$, $p + P + q + Q$ will be considerably smaller than $p + Ps + q + Qs$ and will allow a much more parsimonious model.

Modelo SARIMA multiplicativo

Una herramienta importante es el análisis de procesos estacionales **no estacionarios** es la diferenciación estacional de período s para la serie $\{Y_t\}$, denotada

$$\nabla_s Y_t = Y_t - Y_{t-s}$$

Se dice que una serie estacional no estacionaria sigue un modelo SARIMA(p,d,q)x(P,D,Q) $_s$ de período s si la serie diferenciada

$$W_t = \nabla^d \nabla_s^D Y_t$$

sigue un proceso SARMA(p,q)x(P,Q) $_s$.

$$\nabla_s^d Y_t = (1 - B_s)^d Y_t$$

Criterios de Bondad de modelos

Criterios de bondad de modelo

Uno de los objetivos cuando analizamos series de tiempo es poder modelarlas. Esto incluye tanto elegir la familia de modelos correcta como hallar todos los parámetros de la misma.

Es necesario entonces poder contar con algún criterio que nos permita saber cuán bueno es nuestro modelo, es decir cuán cerca se encuentra la distribución especificada por el modelo a la distribución verdadera de los datos.

Divergencia de Kullback-Leibler (DK-L)

Supongamos que los datos se encuentran generados por un proceso **desconocido** $g(y)$, y que consideramos un modelo $f(y)$ para aproximarnos a $g(y)$.

Si conociéramos $f(y)$, podríamos utilizar la divergencia de Kullback-Leibler

$$D_{\text{KL}}(g \parallel f) = \int_{-\infty}^{\infty} g(y) \log\left(\frac{g(y)}{f(y)}\right) dy = \mathbb{E}\left[\log\left(\frac{g(Y)}{f(Y)}\right)\right]$$

que nos da una medida de la pérdida de información que tenemos al aproximar $f(y)$ por $g(y)$. Cuanto más pequeño sea el valor de la divergencia K-L, más cerca estarán $g(y)$ y $f(y)$. El mejor modelo será entonces aquel minimice la DK-L.

Estimación de la divergencia K-L

Dado que en general no se conoce el valor verdadero de la distribución $g(y)$, debemos estimar el valor de $D_{\text{KL}}(g \parallel f)$, por lo cual debemos estimar su valor a partir de las muestras y_1, \dots, y_n . Para ello, se asume que las muestras son observadas de forma independiente de $g(y)$.

En primer lugar, vemos que podemos reescribir

$$D_{\text{KL}}(g \parallel f) = \mathbb{E} \left[\log \left(\frac{g(Y)}{f(Y)} \right) \right] = \mathbb{E}[\log(g(Y))] - \mathbb{E}[\log(f(Y))]$$

El primer término no puede ser calculado, pero si lo que nos interesa es comparar modelos, podemos descartarlo pues será una constante común a todos.

Estimación de la divergencia K-L

El término $\mathbb{E}[\log(f(Y))] = \int \log(f(y))g(y)dy$ y tampoco se puede calcular. Sin embargo la ley de los grandes números garantiza que

$$\frac{1}{n} \sum_{i=1}^n \log(f(y_i)) \rightarrow \mathbb{E}[\log(f(Y))]$$

Recordemos que $\ell = \sum_{i=1}^n \log(f(y_i)) = \log(L)$ es el logaritmo de la verosimilitud (*log-likelihood*), donde L es la función de verosimilitud.

Obs: Minimizar $D_{\text{KL}}(g \parallel f)$ equivale a maximizar la log-verosimilitud. Por lo tanto una forma natural de hallar los mejores parámetros para un modelo dado es utilizando el estimador de máxima verosimilitud.

Criterio de Información de Akaike (AIC)

El modelo de información de Akaike sirve para comparar la bondad de dos modelos propuestos, y se basa en los resultados analizados previamente.

$$\text{AIC} = 2k - 2\ell(\hat{\theta})$$

cantidad de parámetros

parámetros estimados por MV

Se elegirá entonces el modelo que alcance el menor AIC.

Observaciones:

- El resultado de AIC es asintótico pues se basa en la LGN
- AIC no nos dice cuán bueno es cada modelo sino cuál es el mejor de todos (podría ser que sean todos malos)

Criterio de información Bayesiano

Similar a la definición de AIC, el criterio de información Bayesiano también se basa en la log-verosimilitud, pero penaliza más una mayor cantidad de parámetros.

$$\text{BIC} = k \log(n) - 2\ell(\hat{\theta})$$

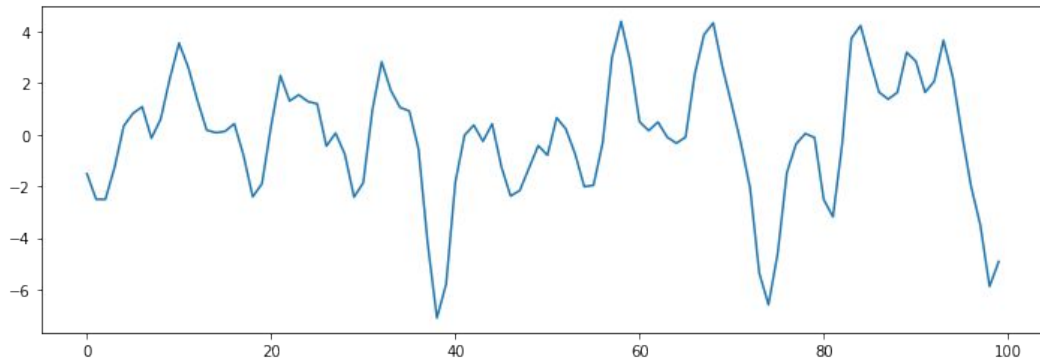
Nuevamente, se preferirán los modelos con BIC más bajo.

Si bien la expresión es similar a AIC, la derivación de BIC es totalmente distinta. AIC aparece como consecuencia de un enfoque Bayesiano, donde lo que se busca es el modelo f que maximice $\mathbb{P}(f|y_1, \dots, y_n)$.

Para una explicación más detallada ver “Elements of Statistical Learning”, Trevor Hastie, Robert Tibshirani, Jerome Friedman.

Ejemplo

```
ar_coef = np.array([0.8, -0.2])  
ma_coef = np.array([0.9, 0.5, -.3])  
arma_ts = arma_generate_sample(  
    ar=np.r_[1, -ar_coef], ma=np.r_[1, ma_coef], nsample=100)  
plt.figure(figsize=(8, 3))  
plt.plot(arma_ts)
```



Ejemplo

```
model_good = ARIMA(arma_ts, order=(2, 3, 0))
model_good_res = model_good.fit()
print(model_good_res.summary())
```

```
=====
Dep. Variable:          y      No. Observations:          100
Model:                ARIMA(2, 3, 0)  Log Likelihood        -206.926
Date:                Fri, 13 May 2022  AIC                419.851
Time:                23:31:29         BIC                427.575
Sample:              0             HQIC                422.974
                                - 100
Covariance Type:      opg
=====
              coef    std err          z      P>|z|      [0.025    0.975]
-----
ar.L1         -0.3483     0.117     -2.977     0.003     -0.578     -0.119
ar.L2         -0.1755     0.115     -1.531     0.126     -0.400     0.049
sigma2         4.1663     0.783     5.318     0.000     2.631     5.702
=====
Ljung-Box (L1) (Q):                1.11  Jarque-Bera (JB):                2.72
Prob(Q):                          0.29  Prob(JB):                      0.26
Heteroskedasticity (H):            1.00  Skew:                          0.14
Prob(H) (two-sided):              1.00  Kurtosis:                      2.23
=====

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step)
```

```
model_bad = ARIMA(arma_ts, order=(5, 5, 0))
model_bad_res = model_bad.fit()
print(model_bad_res.summary())
```

```
=====
SARIMAX Results
=====
Dep. Variable:          y      No. Observations:          100
Model:                ARIMA(5, 5, 0)  Log Likelihood        -234.720
Date:                Fri, 13 May 2022  AIC                481.439
Time:                23:31:32         BIC                496.762
Sample:              0             HQIC                487.631
                                - 100
Covariance Type:      opg
=====
              coef    std err          z      P>|z|      [0.025    0.975]
-----
ar.L1         -1.4774     0.105    -14.112     0.000     -1.683     -1.272
ar.L2         -1.4495     0.172     -8.426     0.000     -1.787     -1.112
ar.L3         -1.3962     0.182     -7.674     0.000     -1.753     -1.040
ar.L4         -0.8958     0.173     -5.189     0.000     -1.234     -0.557
ar.L5         -0.3412     0.104     -3.283     0.001     -0.545     -0.138
sigma2         7.8979     1.543     5.119     0.000     4.874    10.921
=====
Ljung-Box (L1) (Q):                3.41  Jarque-Bera (JB):                3.88
Prob(Q):                          0.06  Prob(JB):                      0.14
Heteroskedasticity (H):            0.99  Skew:                          0.27
Prob(H) (two-sided):              0.98  Kurtosis:                      2.17
=====
```

Cómo estimar el orden del modelo

¿Cómo estimar el orden del modelo?

Función de autocorrelación parcial: define el efecto que tiene $Y(t-k)$ sobre $Y(t)$

$$\alpha(1) = \text{corr}(Y_{t+1}, Y_t), \text{ for } k = 1$$

$$\alpha(k) = \text{corr}((Y_{t+k} - P_{t,k}(Y_{t+k}), Y_t - P_{t,k}(Y_t)), \text{ para } k \geq 2$$

donde $P_{t,k}(x)$ es el operador de proyección ortogonal de x sobre el espacio generado por $Y_{t+1}, \dots, Y_{t+k-1}$

Podemos estimar el orden del modelo usando la función de autocorrelación y la función de autocorrelación parcial.

Si $k > p$, la autocorrelación parcial es nula

Si $k > q$ la autocorrelación es nula

Sobre la estimación del orden de integración

Al estimar el orden d en el modelo ARIMA(p,d,q) se debe tener cuidado de no diferenciar de más.

Si bien la diferencia de cualquier serie estacionaria sigue siendo estacionaria, si diferenciamos más de lo necesario estamos introduciendo correlaciones innecesarias en los datos, complicando el modelado.

Además sobrediferenciar lleva a modelos no invertibles, los cuales traen problemas a la hora de estimar los parámetros del modelo

¿preguntas?