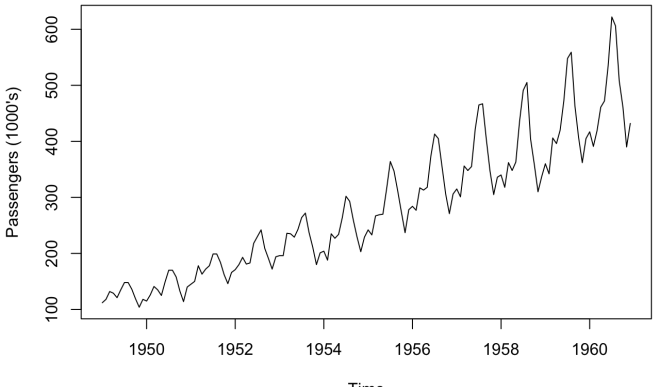
Introducción a Series Temporales

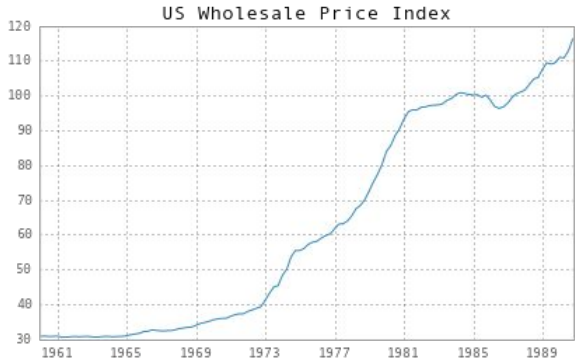
¿Qué es una serie temporal?

Wikipedia:

> *A time series is a **series of data points indexed** (or listed or graphed) **in time order**. Most **commonly**, a time series is a sequence taken at **successive equally spaced points in time**.*

Box & Jenkins AirPassenger
Data (in thousands)





Random variation

Toda colección de datos tomada en el tiempo tiene **ruido**

Métodos de reducción de ruido: “smoothing” (alisamiento)

- Alisamiento por medias
- Alisamiento exponencial

Random variation

Toda colección de datos tomada en el tiempo tiene **ruido**

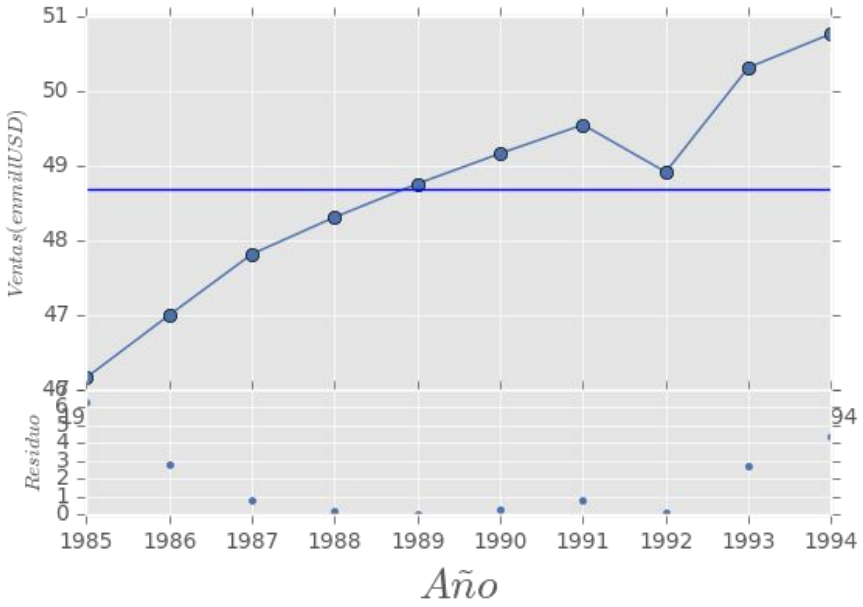
Métodos de reducción de ruido: “smoothing” (alisamiento)

- Alisamiento por medias
- Alisamiento exponencial

¿Qué pasa si queremos predecir usando la media de valores pasados?

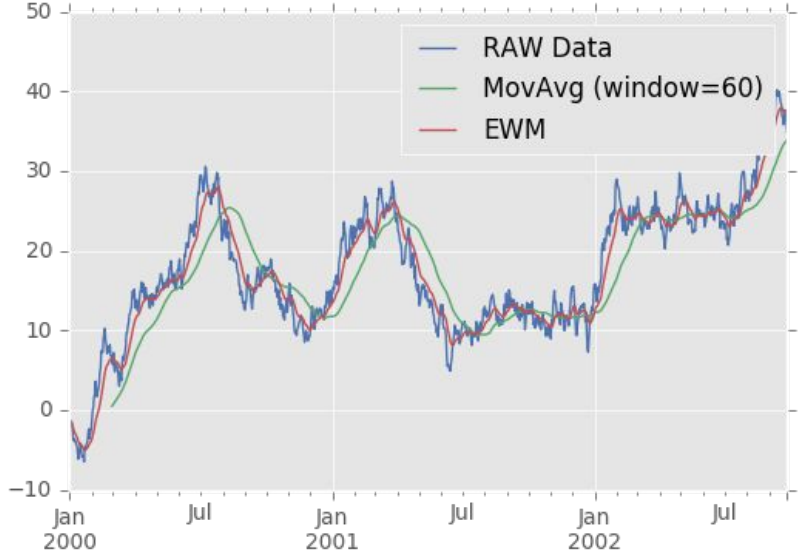
Predicción con la media

La predicción con media sólo es útil cuando no hay “trend”



Moving average

Calcular la media en una ventana de N valores.



- Simple Moving Average(SMA):

$$\bar{p}_{\mathrm{SM}} = \frac{p_M + p_{M-1} + \cdots + p_{M-(n-1)}}{n} = \frac{1}{n}\sum_{i=0}^{n-1} p_{M-i}$$

- Weighted Moving Average (WMA):

$$\mathrm{WMA}_M = \frac{np_M + (n-1)p_{M-1} + \cdots + 2p_{(M-n+2)} + p_{(M-n+1)}}{n + (n-1) + \cdots + 2 + 1}$$

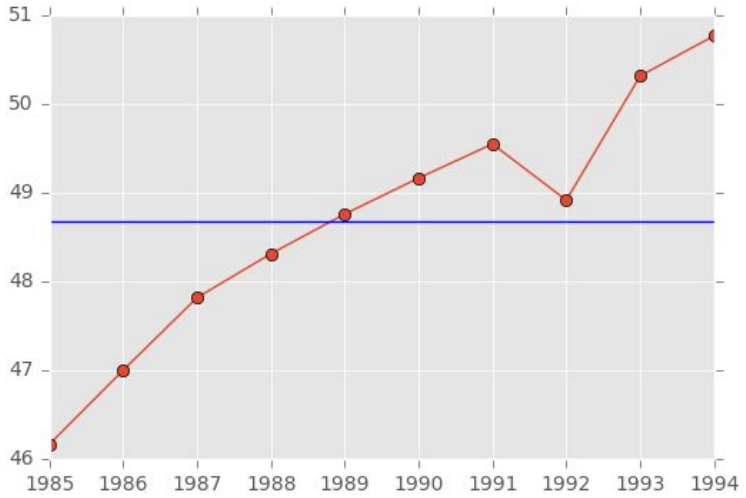
- Exponentially Weighted Moving Average (EWMA)

$$S_t = \begin{cases} Y_1, & t = 1 \\ \alpha \cdot Y_t + (1-\alpha) \cdot S_{t-1}, & t > 1 \end{cases}$$

Mean squared error (Error cuadrático medio)

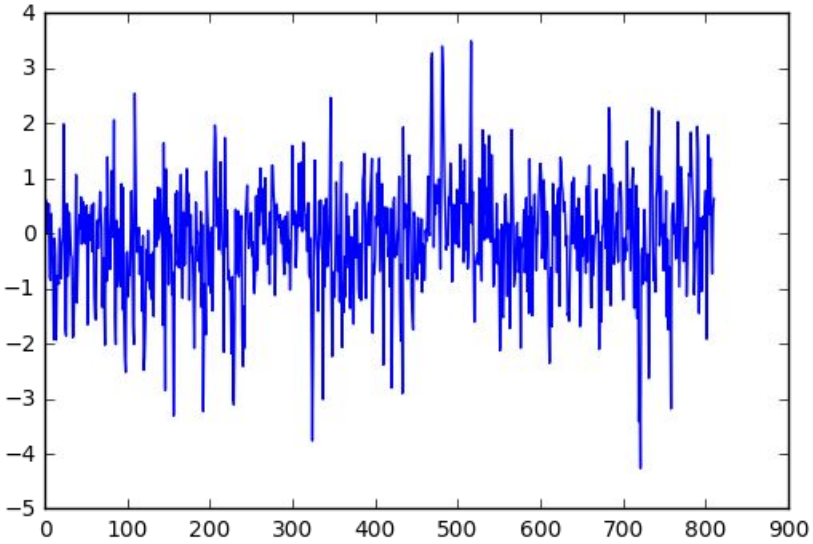
Ejemplo: ventas de PC de un proveedor

- Calculamos la media de ventas
    a. "error" = resta entre estimación (media) y valor real.
    b. "error squared": error al cuadrado.
    c. "SSE": suma de errores al cuadrado.
    d. "MSE": media de errores cuadráticos

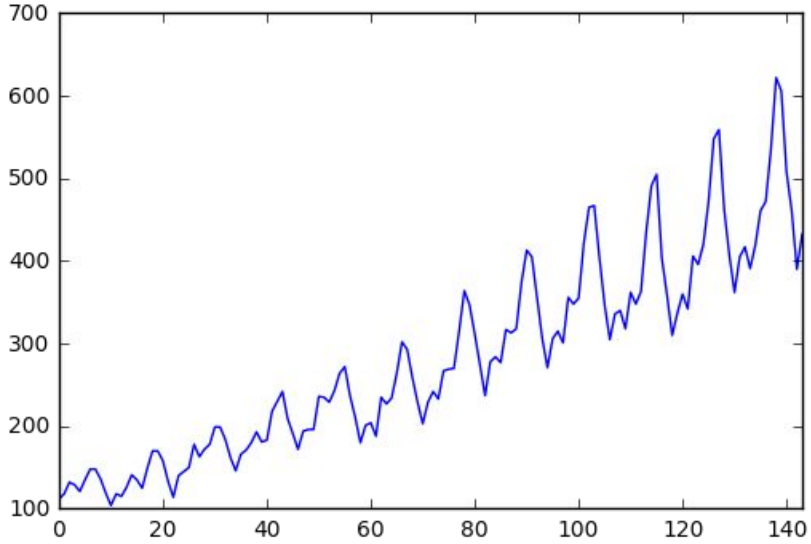


Series estacionarias vs Series no estacionarias

Presión del nivel del mar

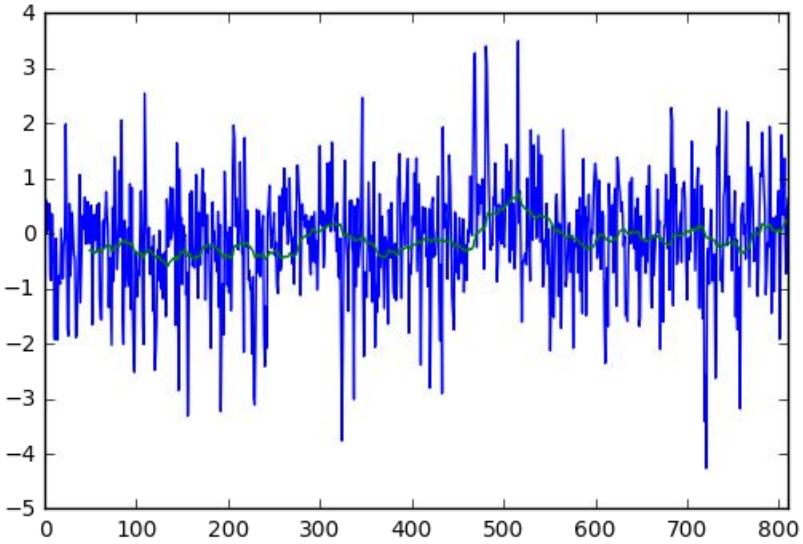


Cantidad de pasajeros aéreos

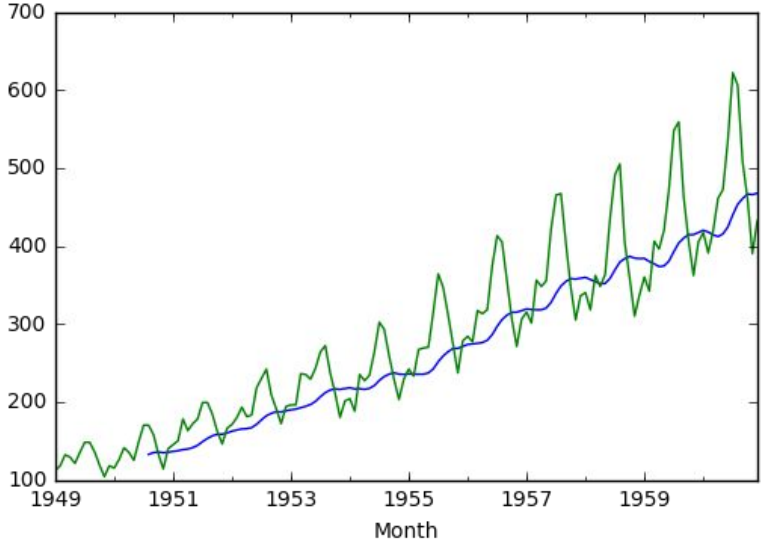


Series estacionarias vs Series no estacionarias

Presión del nivel del mar



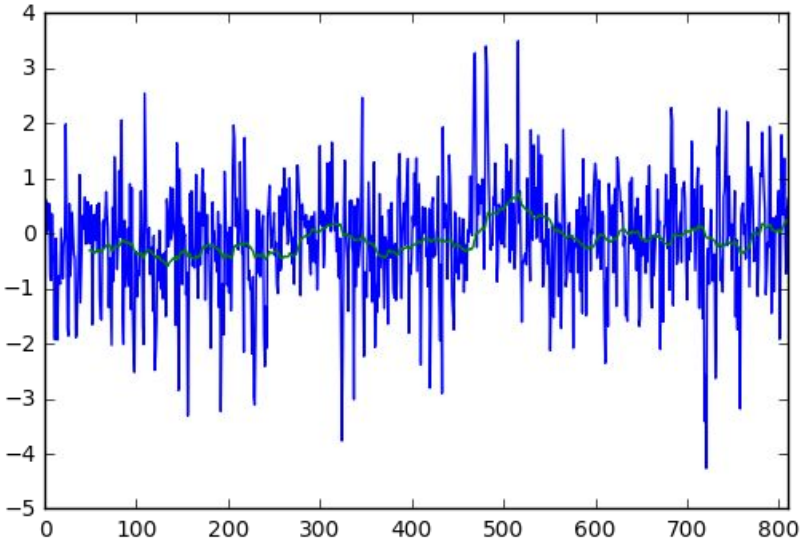
Cantidad de pasajeros aéreos



```
from statsmodels.tsa.stattools import adfuller
print adfuller(df['#Passengers'],)
```

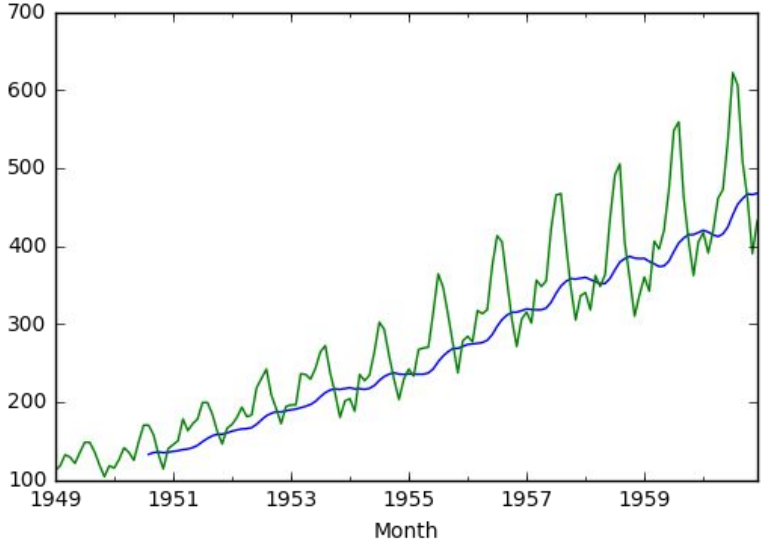
Test de Dickley-Fuller

Presión del nivel del mar



Estadístico = -20.85582597179523
P-value = 0.0

Cantidad de pasajeros aéreos



Estadístico = 0.81536887920604695
p-value = 0.99188024343764103

Goodness of fit

En datos de entrenamiento $$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

Suponer que tenemos f entrenado en {(x1, y1),(x2, y2),...,(xn, yn)}, con un MSE pequeño. Nos interesa:

f(x0) que sea parecido y0, para (x0, y0) una nueva muestra.

¡Queremos elegir el mejor modelo/método que me de el **menor MSE en los datos nuevos**!,

Tendencia lineal

$$Y \approx \beta_0 + \beta_1 X.$$

- Queremos los β que **mejor** ajusten

Tendencia lineal

$$Y \approx \beta_0 + \beta_1 X.$$

- Queremos los β que **mejor** ajusten

- Criterio de cuadrados mínimos

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x},$$

Tendencia lineal

$$Y \approx \beta_0 + \beta_1 X.$$

- Queremos los β que **mejor** ajusten

- Criterio de cuadrados mínimos

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2},$$
$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x},$$

- Errores

$$\mathrm{SE}(\hat{\beta}_0)^2 = \sigma^2 \left[\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}\right], \quad \mathrm{SE}(\hat{\beta}_1)^2 = \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \quad (3.8)$$

- Intervalos de confianza:
  - Si los errores son *normales*, existe aproximadamente un 95% de probabilidades de encontrar a β en el intervalo

$$\left[\hat{\beta}_1 - 2 \cdot \mathrm{SE}(\hat{\beta}_1),\ \hat{\beta}_1 + 2 \cdot \mathrm{SE}(\hat{\beta}_1)\right]$$

Escenario sencillo

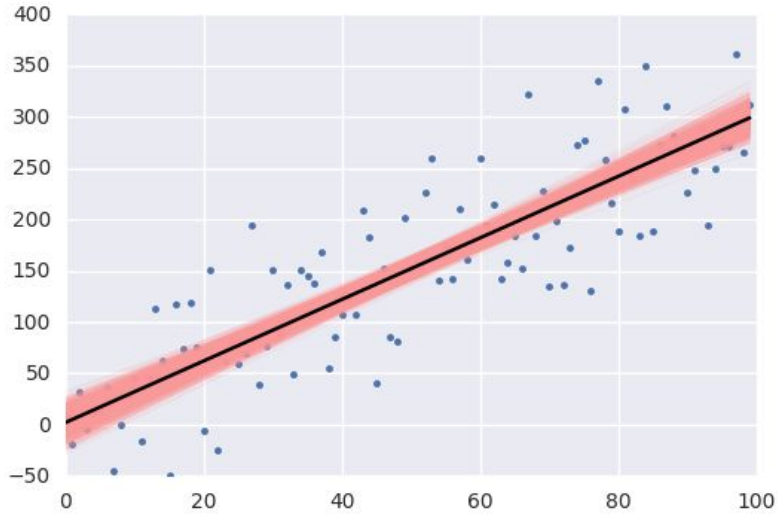
Sabemos la función lineal:

$$f(x) = 2 + 3x$$

Suponemos que medimos con ruido

$$f(x) = 2 + 3x + \varepsilon$$

Calculamos los β para 1000 iteraciones



¿Coeficientes confiables?

Hipótesis nula: no hay relación entre X e Y

Formalmente, queremos ver si β≠0

Calculamos el estadístico

$$t = \frac{\hat{\beta}_1 - 0}{\mathrm{SE}(\hat{\beta}_1)},$$

Si no hay relación entre X e Y, entonces tendremos una distribución t.

Tendencia lineal

```
import statsmodels as sm
print sm.regression.linear_model.OLS(pas,dates).fit().summary()
```

$$Y \approx \beta_0 + \beta_1 X.$$

- Queremos los β que **mejor** ajusten
- Criterio de cuadrados mínimos

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x},$$

OLS Regression Results

Dep. Variable:	#Passengers	R-squared:	0.955
Model:	OLS	Adj. R-squared:	0.955
Method:	Least Squares	F-statistic:	3055.
Date:	Wed, 16 Aug 2017	Prob (F-statistic):	2.19e-98
Time:	23:31:25	Log-Likelihood:	-804.19
No. Observations:	144	AIC:	1610.
Df Residuals:	143	BIC:	1613.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf.	Int.]
x1	3.6012	0.065	55.273	0.000	3.472	3.730

Omnibus:	13.493	Durbin-Watson:	0.271
Prob(Omnibus):	0.001	Jarque-Bera (JB):	6.026
Skew:	-0.264	Prob(JB):	0.0492
Kurtosis:	2.148	Cond. No.	1.00

Tendencia lineal

```
import statsmodels as sm
print sm.regression.linear_model.OLS(pas,dates).fit().summary()
```

$$Y \approx \beta_0 + \beta_1 X.$$

- Queremos los β que **mejor** ajusten
- Criterio de cuadrados mínimos

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x},$$

OLS Regression Results

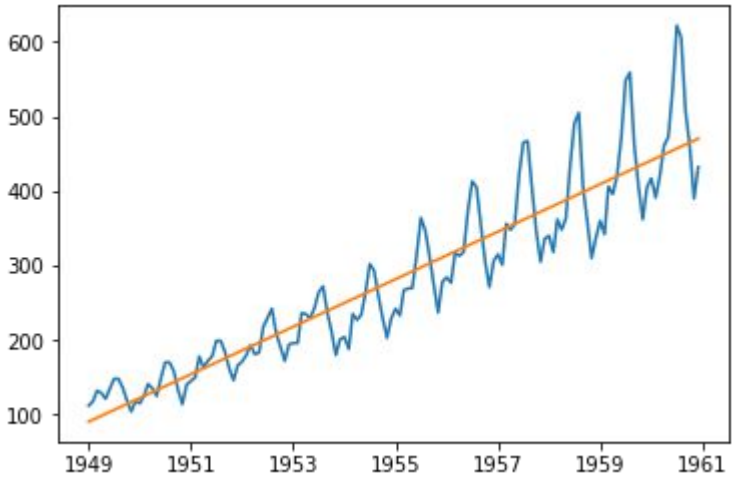
Dep. Variable:	#Passengers	R-squared:	0.955
Model:	OLS	Adj. R-squared:	0.955
Method:	Least Squares	F-statistic:	3055.
Date:	Wed, 16 Aug 2017	Prob (F-statistic):	2.19e-98
Time:	23:31:25	Log-Likelihood:	-804.19
No. Observations:	144	AIC:	1610.
Df Residuals:	143	BIC:	1613.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf.	Int.]
x1	3.6012	0.065	55.273	0.000	3.472	3.730

Omnibus:	13.493	Durbin-Watson:	0.271
Prob(Omnibus):	0.001	Jarque-Bera (JB):	6.026
Skew:	-0.264	Prob(JB):	0.0492
Kurtosis:	2.148	Cond. No.	1.00

Regresión lineal

```
dates = df.index.to_julian_date()
dates = sm.add_constant(dates)
fit = sm.regression.linear_model.OLS(pas,dates).fit()
```



OLS Regression Results

Dep. Variable:	#Passengers	**R-squared:**	0.854
Model:	OLS	**Adj. R-squared:**	0.852
Method:	Least Squares	**F-statistic:**	827.3
Date:	Sun, 03 Sep 2017	**Prob (F-statistic):**	4.29e-61
Time:	22:37:30	**Log-Likelihood:**	-754.89
No. Observations:	144	**AIC:**	1514.
Df Residuals:	142	**BIC:**	1520.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2.123e+05	7390.250	-28.725	0.000	-2.27e+05	-1.98e+05
x1	0.0873	0.003	28.763	0.000	0.081	0.093

Omnibus:	24.600	**Durbin-Watson:**	0.538
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	33.832
Skew:	0.939	**Prob(JB):**	4.50e-08
Kurtosis:	4.453	**Cond. No.**	4.69e+09

```
fit = sm.regression.linear_model.OLS(pas,dates).fit()
```

Evaluando la regresión lineal

Residual Standard Error: `fit.mse_resid`

$$\mathrm{RSE} = \sqrt{\frac{1}{n-2}\mathrm{RSS}} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}.$$

OLS Regression Results

Dep. Variable:	#Passengers	**R-squared:**	0.854
Model:	OLS	**Adj. R-squared:**	0.852
Method:	Least Squares	**F-statistic:**	827.3
Date:	Sun, 03 Sep 2017	**Prob (F-statistic):**	4.29e-61
Time:	22:37:30	**Log-Likelihood:**	-754.89
No. Observations:	144	**AIC:**	1514.
Df Residuals:	142	**BIC:**	1520.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2.123e+05	7390.250	-28.725	0.000	-2.27e+05	-1.98e+05
x1	0.0873	0.003	28.763	0.000	0.081	0.093

Omnibus:	24.600	**Durbin-Watson:**	0.538
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	33.832
Skew:	0.939	**Prob(JB):**	4.50e-08
Kurtosis:	4.453	**Cond. No.**	4.69e+09

```
fit = sm.regression.linear_model.OLS(pas,dates).fit()
```

Evaluando la regresión lineal

Residual Standard Error: `fit.mse_resid`

$$\mathrm{RSE} = \sqrt{\frac{1}{n-2}\mathrm{RSS}} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}.$$

R-squared: $R^2 = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}} = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$

$$\mathrm{TSS} = \sum (y_i - \bar{y})^2$$

Notar que $R^2 = r^2$ con $r = \mathrm{Cor}(X, Y)$

OLS Regression Results

Dep. Variable:	#Passengers	**R-squared:**	0.854
Model:	OLS	**Adj. R-squared:**	0.852
Method:	Least Squares	**F-statistic:**	827.3
Date:	Sun, 03 Sep 2017	**Prob (F-statistic):**	4.29e-61
Time:	22:37:30	**Log-Likelihood:**	-754.89
No. Observations:	144	**AIC:**	1514.
Df Residuals:	142	**BIC:**	1520.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2.123e+05	7390.250	-28.725	0.000	-2.27e+05	-1.98e+05
x1	0.0873	0.003	28.763	0.000	0.081	0.093

Omnibus:	24.600	**Durbin-Watson:**	0.538
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	33.832
Skew:	0.939	**Prob(JB):**	4.50e-08
Kurtosis:	4.453	**Cond. No.**	4.69e+09

Regresión lineal múltiple

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon,$$

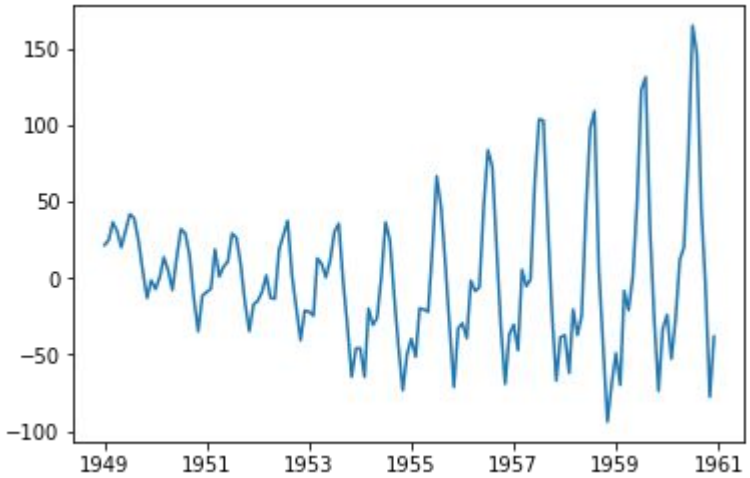
Queremos saber si alguno de los β es distinto de 0

$$H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$$

Se calcula el estadístico $F = \dfrac{(\text{TSS} - \text{RSS})/p}{\text{RSS}/(n-p-1)},$

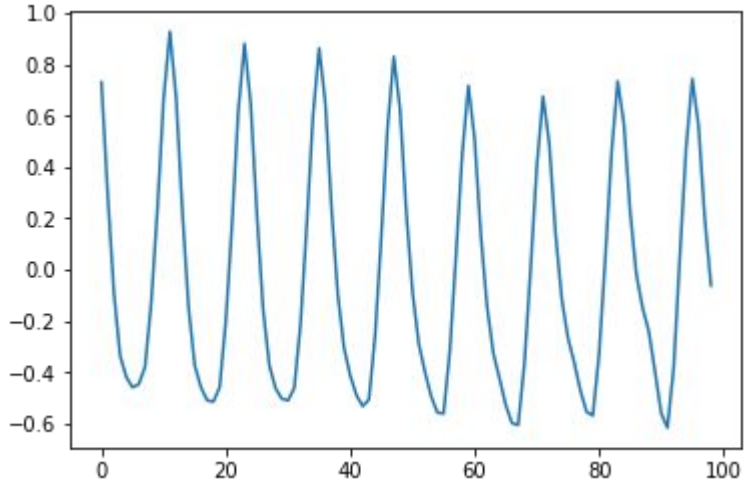
Mirando señal periódica

```
lineal = df.index.to_julian_date()*fit.params[1]+fit.params[0]

plt.plot(df.index,df['#Passengers']-lineal)
```



Mirando señal periódica: autocorrelación

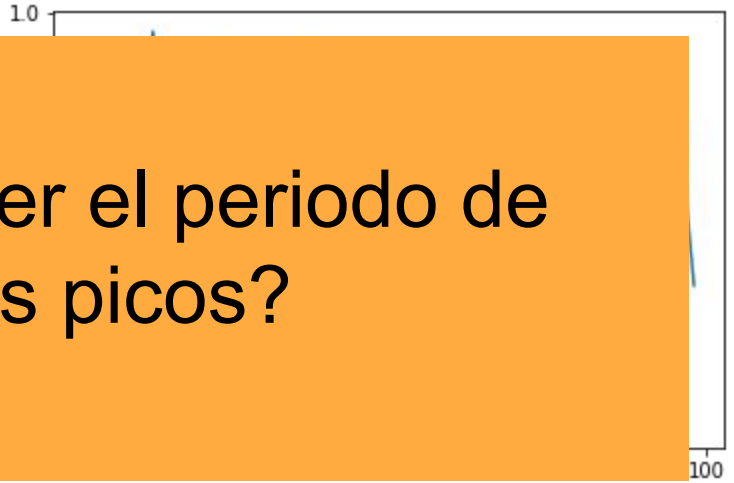
Definición: la autocorrelación es la correlación de Pearson entre los valores de una serie temporal y la misma serie corrida por un lag




```
plt.plot([(df['#Passengers']-lineal).autocorr(lag=x) for x in range(1,100)])
```

Mirando señal periódica: autocorrelación

Defi
corr
una
por



¿Cómo hago para saber el periodo de repetición de los picos?

```
plt.plot([(df['#Passengers']-lineal).autocorr(lag=x) for x in range(1,100)])
```

Transformada de Fourier

Espectro de frecuencias: toda señal s(t) puede ser descompuesta como

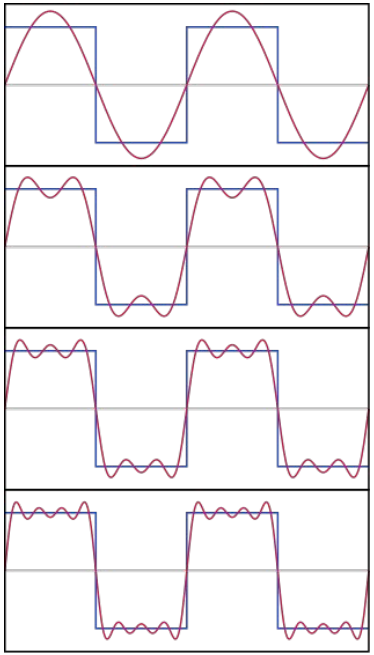
$$s(t) = \int_{\mathbb{R}} A(\nu) e^{-2\pi i \nu t} d\omega$$

O en caso de tener una señal ***f(t)*** integrable en el intervalo $[t_0 - T/2, t_0 + T/2]$

$$f(t) \sim \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \cos\left(\frac{2n\pi}{T} t\right) + b_n \sin\left(\frac{2n\pi}{T} t\right) \right]$$

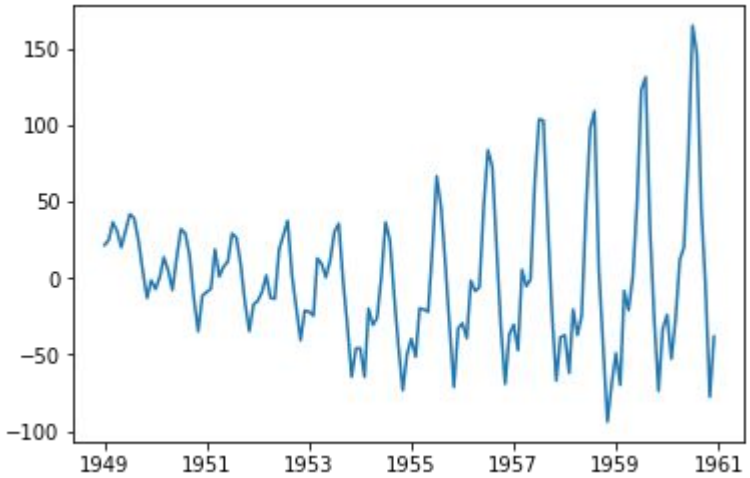
EN CRIOLLO: una serie de datos puede ser escrita como sumas de sin y cos. Mirando los coeficientes de la serie de fourier podemos identificar patrones periódicos salientes.

Ejemplo: reconstrucción de una onda cuadrada



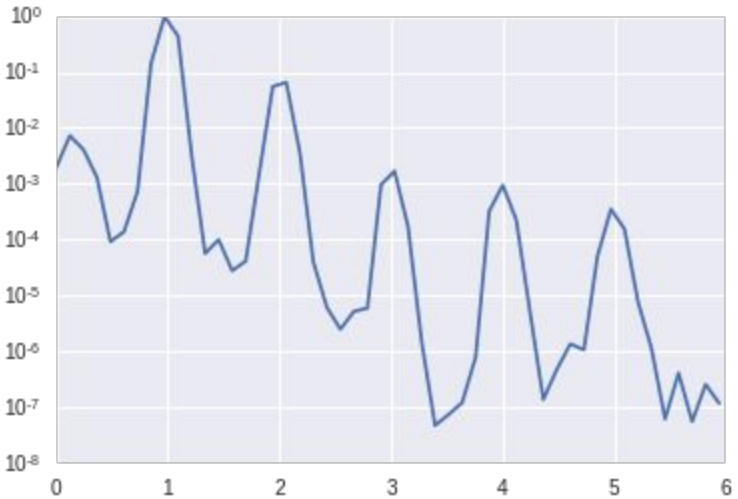
Mirando señal periódica

```
lineal = df.index.to_julian_date()*fit.params[1]+fit.params[0]

plt.plot(df.index,df['#Passengers']-lineal)
```

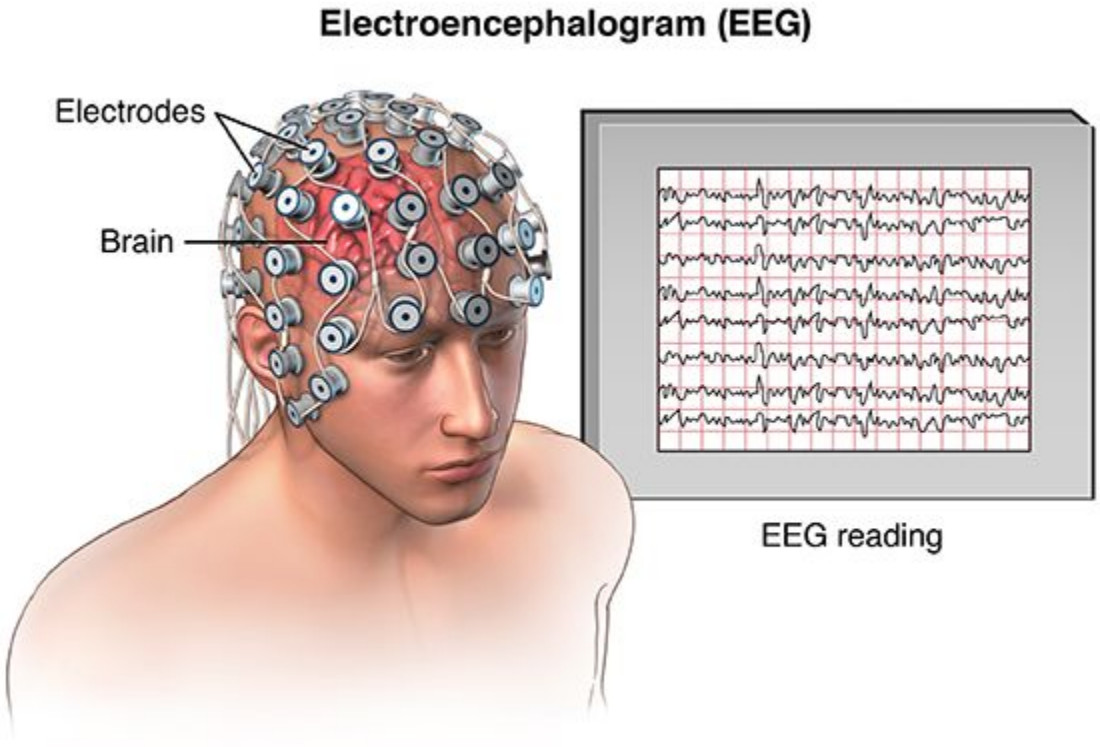


Fast Fourier Transform (welch)

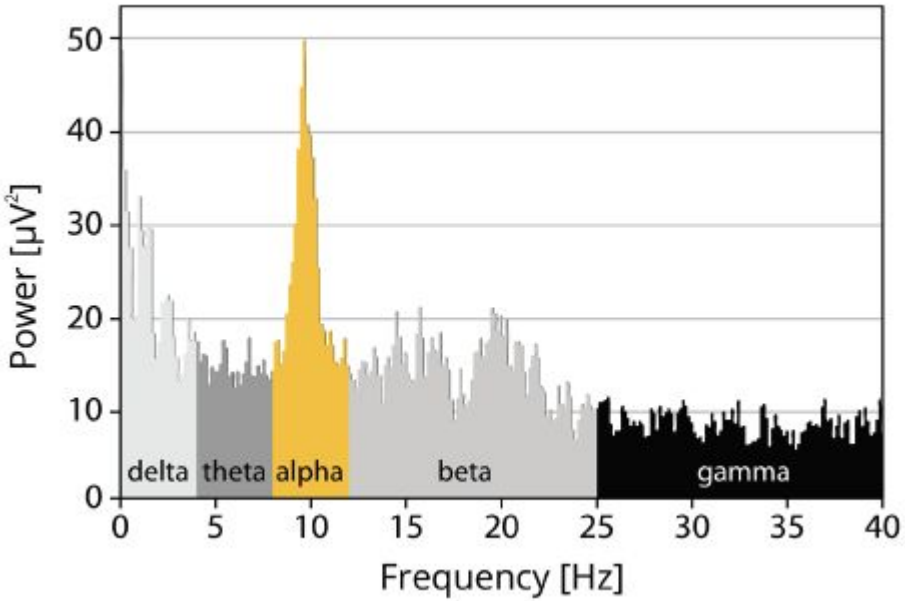
```
import scipy

ac = [(df['#Passengers']-lineal).autocorr(lag=x) for x in range(1,100)]
f, Pxx_den = scipy.signal.welch(ac,fs=12)
plt.semilogy(f, Pxx_den)
```



TP de EEG



Frecuencias características en EEG



Resumen

- ¿Qué es una serie temporal?
- Suavizado por media, ventana móvil, etc
- Series estacionarias vs no estacionarias (test de Dickley-Fuller)
- Regresión lineal
  http://www-bcf.usc.edu/~gareth/ISL/ISLR%20Seventh%20Printing.pdf
- Evaluación de la regresión lineal
- Analizando señal periódica
  - Autocorrelación
  - Transformada de fourier

Resumen de paquetes y funciones

Para manejo de series: pandas

```
import pandas as pd
```

Para regresión lineal y test adfuller: statsmodels

```
import statsmodels.api as sm
```

Sm.regression.OLS y sm.tsa.adfuller

Para autocorrelación: autocorr en pandas

Para FFT: método welch en scipy.signal

```
from scipy.signal import welch
```