



Time Series Management

Michele Linardi Ph.D.

michele.linardi@orange.fr



Syllabus

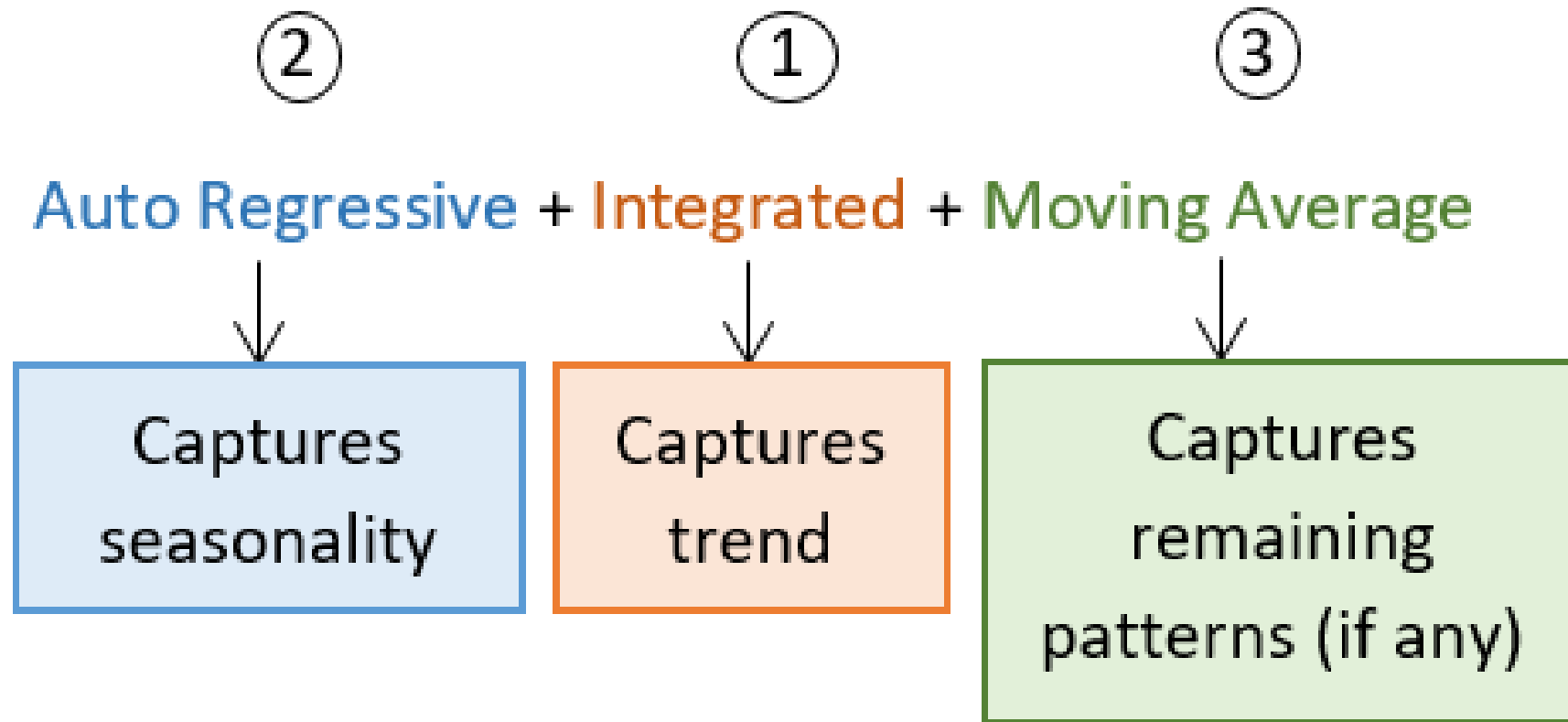
- ARIMA model
- Backshift Notation
- Model selection and forecasting
- MLE (Maximum likelihood estimation)
- AIC Akaike's Information Criterion

Time series data... quick recap

- A **univariate time series** is a sequence of measurements of the same variable collected over time. Most often, the measurements are made at regular time intervals.



AR.I.MA



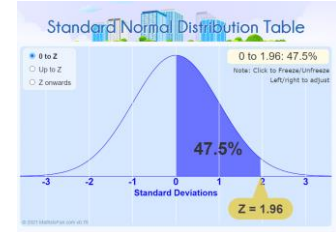
ACF and PACF

$$\text{ACF (lag } h) = \rho_h = \frac{\text{Covariance}(X_{[t]}, X_{[t-h]})}{\text{Std.Dev.}(X_{[t]})\text{Std.Dev.}(X_{[t-h]})}$$

Limites de signification ACF et PACF

$$S^+ = \frac{+Z}{N-h} \quad S^- = \frac{-Z}{N-h}$$

$N := \text{time series values}$ $h := \text{lag}$



Pick Z such that the area of $N(0,1)$ between $(-Z)$ and (Z) is equal to the desired amount of confidence (in general for 95% c.i. , $Z = 1.96$)

<https://www.mathsisfun.com/data/standard-normal-distribution-table.html>

Direct (partial) effect of $X_{[t-h]}$ on $X_{[t]}$, removing $X_{[t-h+1]} , \dots , X_{[t-1]}$

$$\text{PACF (lag 3)} = \rho_{3,3} = \frac{\text{Covariance}(X_{[t]}, X_{[t-3]} | X_{[t-2]}, X_{[t-1]})}{\sqrt{\text{Variance}(X_{[t]} | X_{[t-1]}, X_{[t-2]}) \text{Variance}(X_{[t-3]} | X_{[t-1]}, X_{[t-2]})}}$$

...

Levinson and Durbin recursive algorithm (for PACF computation)

$$\text{PACF (lag } h) = \rho_{h,h} = \frac{\rho_h - \sum_{j=1}^{h-1} \rho_{h-1,j} \rho_{h-j}}{1 - \sum_{j=1}^{h-1} \rho_{h-1,j} \rho_j}$$

where :

$$\rho_{h,j} = \rho_{h,j} - \rho_{h,h} \rho_{h-1,h-j}$$

for :

$$j = 1, 2, \dots, h-1$$

Autoregression (AR) models

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \dots + w_t$$

Where c is a constant, ϕ_i are parameters to estimate and w_t the error term (iid $\sim N(0, \sigma_{\omega_t})$)

Constraints:

- For $p = 1$, $-1 < \phi_1 < 1$.
- For $p = 2$, $-1 < \phi_2 < 1$, $\phi_2 + \phi_1 < 1$ and $\phi_2 - \phi_1 < 1$.
- For $p = 3$, more complicated conditions

Integrated model (d)

It is important to remove trends, or non-stationarity, from time series data prior to model building, since such autocorrelations dominate the ACF. One way of removing non-stationarity is through the method of differencing. :

$$Y'_t = Y_t - Y_{t-1}$$

Occasionally, such taking of first differences is insufficient to remove non-stationarity. In that case, second-order differences usually produce the desired effect:

$$Y''_t = Y'_t - Y'_{t-1}$$

d := order of the integration

Integrated model (seasonal)

A seasonal difference is the difference between an observation and the corresponding observation from the previous year, quarter or month as appropriate, where s is the number of time periods back. For example, with monthly data, $s = 12$ and the seasonal difference is obtained as:

$$Y'_t = Y_t - Y'_{t-12}$$

Moving Average (MA) models

$$Y_t = c + w_t - \theta_1 w_{t-1} - \theta_2 w_{t-2} + \dots$$

Where c is a constant, θ_i are parameters to estimate and w_i the error terms (iid $\sim N(0,1)$)

Constraints:

- For $q = 1$, $-1 < \theta_1 < 1$.
- For $q = 2$, $-1 < \theta_2 < 1$, $\theta_2 + \theta_1 < 1$.
- For $q = 3$, more complicated conditions

Backshift notation

The backshift notation is commonly used to represent ARIMA models. It uses the operator B , which shifts data back one period:

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

Two applications of B shift the data back two periods:

$$B(BY_t) = B^2Y_t = Y_{t-2}$$

In general, B^s represents “shift back s time periods”. Note that a first difference is represented by $1 - B$:

$$Y'_t = Y_t - Y_{t-1} = Y_t - BY_t = (1-B) Y_t$$

$$\text{ARIMA}(p,d,q) = \text{AR}(p) , I(d) , \text{MA}(q)$$

$$\text{ARIMA}(1,1,1) \Rightarrow Y_t = c + \phi_1 Y_{t-1} + \theta_1 w_{t-1} + w_t$$

$$\Rightarrow (1 - \phi_1 B)(1 - B)Y_t = c + (1 + \theta_1 B) w_t$$

The general expression of the ARIMA(p, d, q) in backshift notation :

$$\text{ARIMA}(p,d,q) \Rightarrow (1 - \phi_1 B - \dots - \phi_p B^p)(1 - B)^d Y_t = c + (1 + \theta_1 B + \dots + \theta_q B^q) w_t$$

Model selection and forecasting

Phase 1 (Identification):

Preliminary analysis and data preparation:

- Difference data to obtain a stationary series.

Model selection:

- Analyse time plots, ACF, PACF to identify potential models.

Model selection and forecasting

Phase 2 (Estimation and testing (1/2)):

1. Estimate parameters in potential models
2. Select best model using suitable criterion...
3. If satisfying model is not found go back to Phase 1
4. Otherwise...

Model selection and forecasting

Phase 2 (Estimation and testing (1/2)):

1. **Estimate parameters in potential models**
2. Select best model using suitable criterion...
3. If satisfying model is not found go back to Phase 1
4. Otherwise...

Likelihood

**Likelihood function := $L(\text{Model with parameters } \theta \mid \text{observed data})$
:= $\text{Prob}(\text{observed data} \mid \text{Model with parameters } \theta)$**

The likelihood of a fully-specified model with a set of parameters θ , given some observed data, is equal to the probability of observing these data, given the defined model with those specific parameter values.

Likelihood is a quantitative measure of model fit. Higher likelihoods correspond to a higher probability of the model producing the observed data (the data “fit” the model well).

Maximum Likelihood estimation (MLE)

$$\hat{\Theta}_{MLE} = \underset{\Theta}{\operatorname{arg\,max}} L(\Theta)$$

The goal of MLE is to find the parameter Θ that maximizes the likelihood function (or equivalently, the log-likelihood).

Likelihood of the Normal Distribution

Consider you have a sample of data x_1, x_2, \dots, x_n drawn from a normal distribution with unknown mean μ and variance σ^2 .

$$f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

The log-likelihood function is:

$$\log L(\mu, \sigma^2) = -\frac{n}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \mu)^2$$

Maximizing this with respect to μ and σ^2 gives the MLE estimates:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^n x_i$$

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{\mu})^2$$

Sum of squares error (SSE):

$$SSE(\boldsymbol{\theta}) = \sum_{i=1}^T (Y_i - f(Y_{i-1}, \dots; \boldsymbol{\theta}))^2 = \sum_{i=1}^T r_i^2(\boldsymbol{\theta})$$

Parameters



Model prediction



Find the values of the parameters which maximise the probability of obtaining the data that we have observed (minimize **$SSE(\boldsymbol{\theta})$**)

AIC (Akaike's Information Criterion)

$$\text{AIC} = -2 \ln(L) + 2m$$

L := likelihood

$m = p + q$.

By varying the choices of p , q , we want to pick the lowest AIC.

AIC penalizes overfitting

BIC (Bayesian Information Criterion)

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

L:= likelihood

K := number of parameters to estimate

By varying the choices of p, q, we want to pick the lowest AIC.

The BIC introduces a penalty term for the number of parameters in the model. The penalty term is larger in BIC than in AIC.

Model quality - R^2

- The R^2 quantifies the degree of any linear correlation between Y_{obs} and Y_{pred} .

$$S(\boldsymbol{\theta}) = \sum_{i=1}^T (Y_i - f(Y_{i-1}, \dots; \boldsymbol{\theta}))^2 = \sum_{i=1}^T r_i^2(\boldsymbol{\theta})$$

$$S_{\text{tot}}(\boldsymbol{\theta}) = \sum_{i=1}^T (Y_i - \bar{Y})^2$$

$$R^2 = 1 - \frac{S(\boldsymbol{\theta})}{S_{\text{tot}}(\boldsymbol{\theta})}$$


Error

$$RMSE = \sqrt{\frac{\sum_{i=1}^T (Y_i - \widehat{Y}_i)^2}{N}}$$

Prediction

The RMSE (or RMSD) of a sample is the quadratic mean of the differences between the observed values and predicted ones. These deviations are called **residuals** when the calculations are performed over the data sample that was used for estimation and are called **errors** (or prediction errors) when computed out-of-sample.

Error

$$MAPE = \sum_{i=1}^N \sqrt{\frac{(Y_i - \widehat{Y}_i)^2}{Y_i}}$$


Prediction

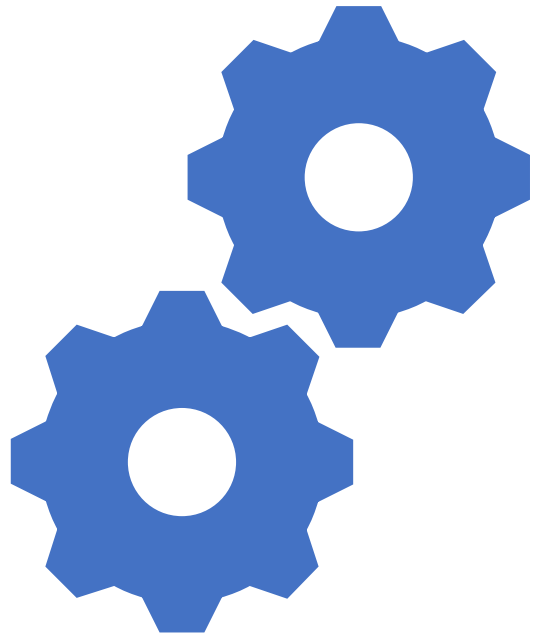
The **mean absolute percentage error** (MAPE), also known as mean absolute percentage deviation (MAPD), is a measure of prediction accuracy of a forecasting method in statistics.

Model selection and forecasting

Phase 2 (Estimation and testing (2/2)):

Testing:

1. Check ACF/PACF of residuals (errors)
2. Are residuals white noise ?



Model selection and forecasting

Phase 3 (Application):

Use the model to forecast.

TS Analysis in Python

- SciPy
- NumPy;
- Matplotlib;
- Pandas;

 statsmodels

<https://www.statsmodels.org/stable/index.html>
<https://github.com/statsmodels/statsmodels/>



ACF and PACF statsmodels

Import the data in a Pandas DataFrame

- `from pandas import read_excel`
- `series = read_excel([filename], sheet_name='MAdata',)`

https://pandas.pydata.org/docs/reference/api/pandas.read_excel.html

ACF and PACF



Import functions to plot ACF and PACF

- `from statsmodels.graphics.tsaplots import plot_acf, plot_pacf`
- `plot_acf([series], title='', lags=xx)`
- `plot_pacf([series], title='', lags=xx)`

https://www.statsmodels.org/stable/generated/statsmodels.graphics.tsaplots.plot_acf.html#statsmodels.graphics.tsaplots.plot_acf

ARIMA MODEL statsmodels

Compute ARIMA MODEL

```
from statsmodels.tsa.arima.model.ARIMA import ARIMA.  
model = ARIMA([series], order=(p, d, q)),
```

where p , d , and q represent the parameters of the model

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

<https://www.statsmodels.org/stable/dev/generated/statsmodels.base.model.GenericLikelihoodModelResults.aic.html#statsmodels.base.model.GenericLikelihoodModelResults.aic>

TD - Consignes

- Given the file **priceData.xlsx**
- Write a Python program to **compute the ACF and PACF** (test several lags) of the time series contained in **priceData.xlsx**
- Write a Python program to **estimate and compute the best ARIMA** model of the time series in **priceData.xlsx**, based on **AIC criterion**.
- **Which kind of model is chosen ?**

References

- Andrew V. Metcalfe, Paul S.P. Cowpertwait, **Introductory Time Series with R** (2009).
- Aileen Nielsen, **Practical Time Series Analysis: Prediction with Statistics and Machine Learning** (2019).
- Changquan Huang , Alla Petukhina, **Applied Time Series Analysis and Forecasting with Python** (2022)