L2 Introduction to ARIMA

MSDS Spring 2025

Outline

I Important concepts to start with

- Stationary time series
- Auto-correlation (ACF)
- Apply in data: sample ACF and check for stationary

II ARIMA model approach

- Introduction
- ARMA for stationary process
- ARIMA for data with trend
- SARIMA for data with trend and seasonality
- SARIMAX to add exogenous predictors

I. Important Concepts related to ARIMA

Classical Statistical Models for Time Series Forecasting

The Classical Statistical Models, like ARIMA, Exponential Smoothing, and Prophet focus on predicting the components of the time series data:

- Trend
- Seasonality
- Cyclic variations caused by holidays or rare events
- Remaining noises (stationary)

And some methods allow to add exogenous variables (predictors other than the response time series itself)

- Identify and remove the components represents a strategic first step for building a time series forecasting model.
- The time series with the trend and/or seasonality removed often achieve an important status: stationary
- Stationary Time Series have consistent statistical characteristics that is easier to model.

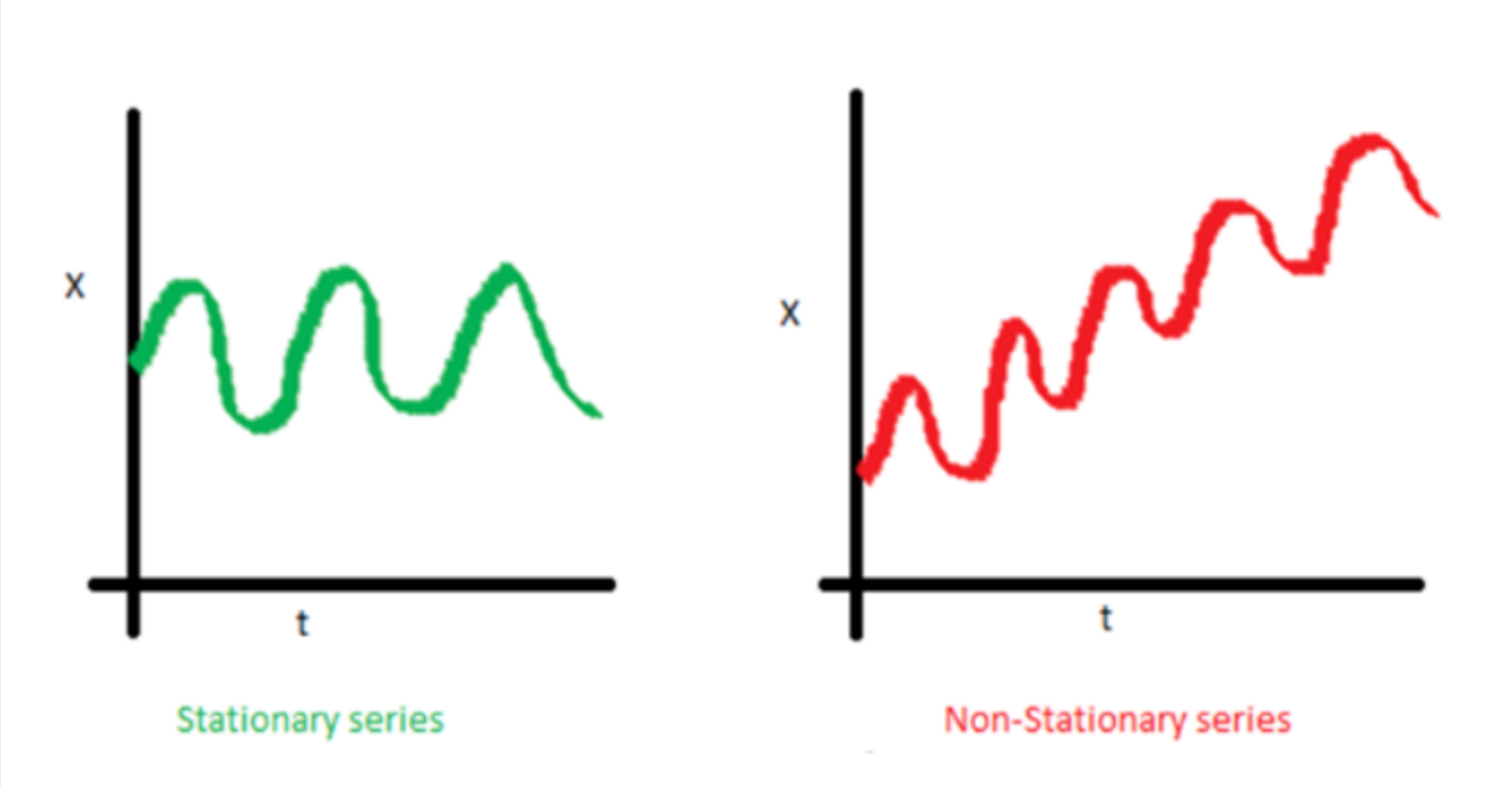
A. Stationary time series

Def: Time series $\{X_t\}$ is stationary if

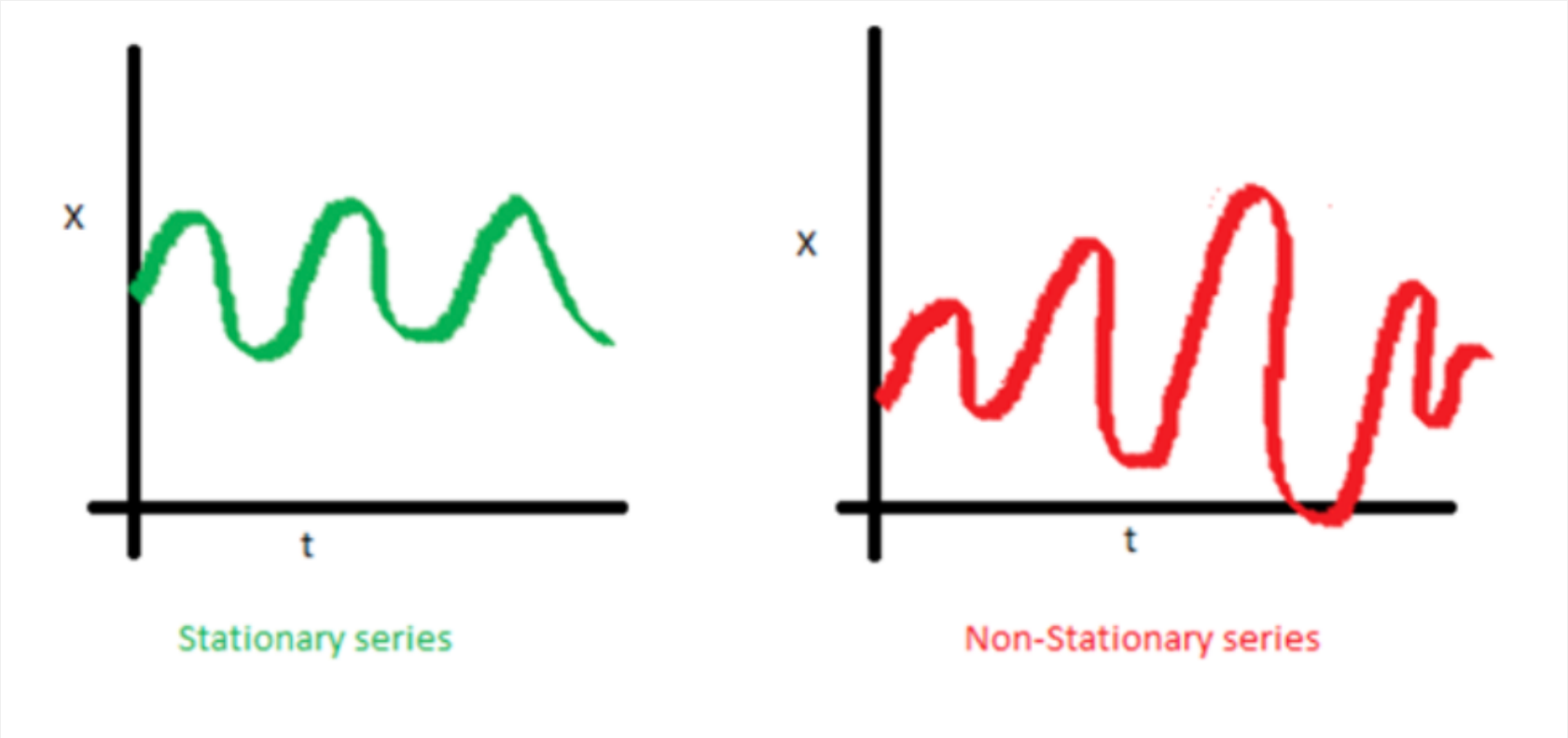
1. $E(X_t)$ is independent of t
2. $\text{Var}(X_t)$ is independent of t
3. $\text{Cov}(X_{t+h}, X_t)$ is independent of t for each h.

Remarks: The key characteristics of the stationary time series do not change over time, which makes it easy to model and predict:

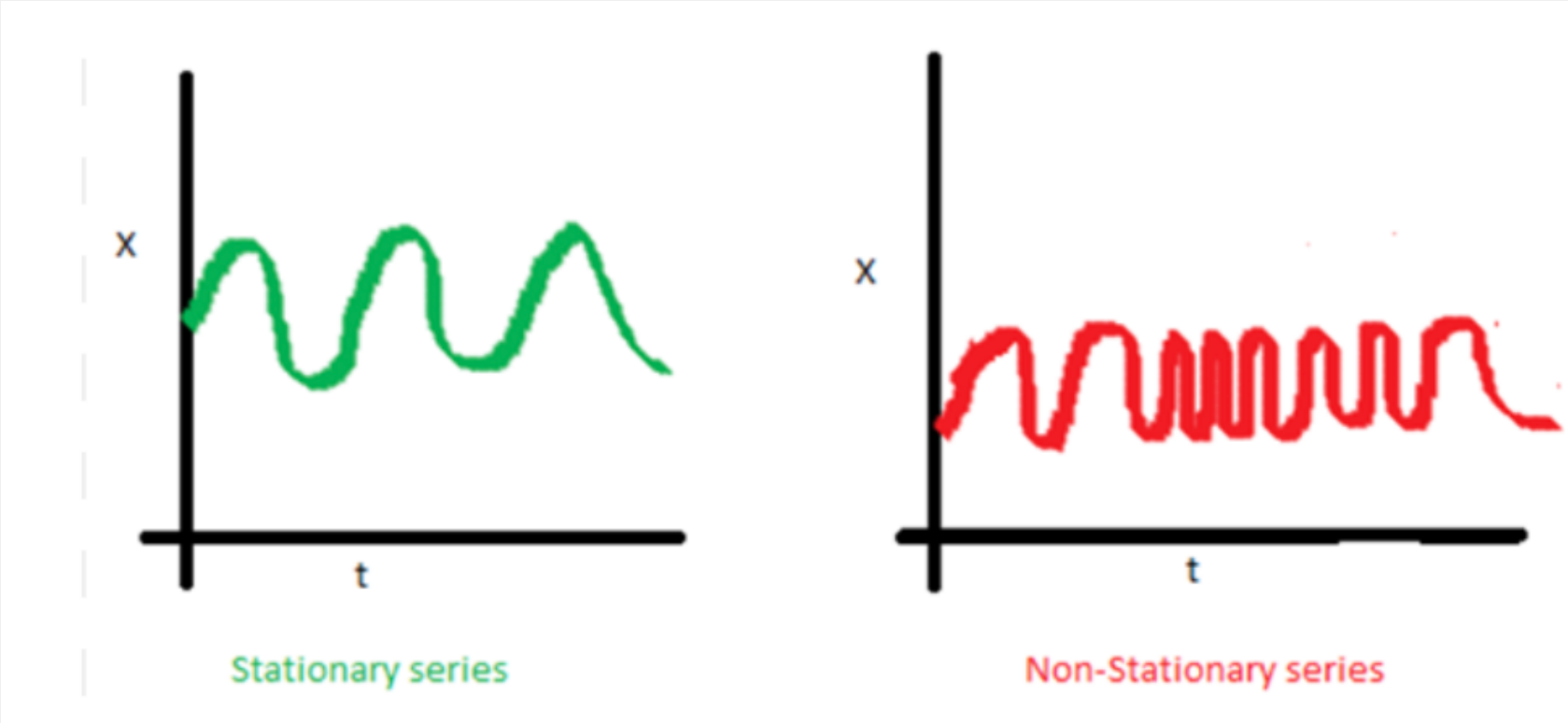
The mean does not change over time (flat trend)



The variance does not change over time



The correlation for observation between the same time lag does not change



B. Autocorrelation (ACF)

The Autocovariance Function (*ACVF*) of $\{X_t\}$ at lag h is

$$\text{Cov}\,(X_{t+h}, X_t) = \gamma_x(h)$$

The Autocorrelation Function (ACF) of $\{X_t\}$ at lag h is

$$\text{Cor}\,(X_{t+h}, X_t) = \frac{\text{cov}\,(X_{t+h}, X_t)}{\text{Var}\,(X_t)} = \frac{\gamma_x(h)}{\gamma_x(0)} = \rho(h)$$

Notes

In Time Series Analysis. It’s often useful to understand the

- Theoretical/population case, and the

- Computational / sample case.

Theoretical cases would help understand how the model is developed, sample case help understand the algorithm and calculation using a data.

Sample ACF and Stationary Test

For observed data, we don't have an assumption of given model to start with, so we need to develop sample methods to explore the potential model choice. We will start with

- Sample ACF

- Augmented Dicky Fuller test for stationary.

Sample ACF

Recall, the sample covariance between two random variables X and Y is given by

$$\hat{Cov}(X, Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{n - 1}$$

the sample correlation between two random variables X and Y is given by

$$\hat{Corr}(X, Y) = \frac{\hat{Cov}(X, Y)}{S_X S_Y}$$

where X_1, ..., X_n is a random sample of X, Y_1, ..., Y_n is a random sample of Y, S_Y and S_Y are the sample standard deviation of X and Y respectively.

For a given Time Series data $x_0, x_1, \ldots, x_n$,
The Sample Autocovariance (ACVF) of lag h is given by:

$$\hat{r}(h) = \frac{1}{n+1} \sum_{t=0}^{n-|h|} \left(x_{t+|h|} - \bar{x}\right)\left(x_t - \bar{x}\right)$$

The Sample Autocorrelation (ACF) of lag h is given by

$$\hat{\beta}(h) = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)}$$

This statistic is similar to sample covariance, but with two twists:

1. It uses the total of sample size instead of the sample used in the equation

2. It uses grand mean of the whole data, instead of the average of data used in the equation.

Because of these twists, the sample ACF plots generates unique behaviors for different time series patterns.

Behavior of Sample ACF

1. $h = 0, \quad \hat{\rho}_X(0) = 1$

2. $h \uparrow n$, there one fewer samples to he used to calculate $\hat{\rho}_X(h)$, often result in very small values or noises.

3. All the sample ACF for $h = 0, 1, \cdots$ are calculated using the sample total mean and variance, so the trend, seasonality and/or pattern of the data is reflected in the ACF plot.

Example

Data: $x_0 \quad x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5$ When lag $h = 3$,

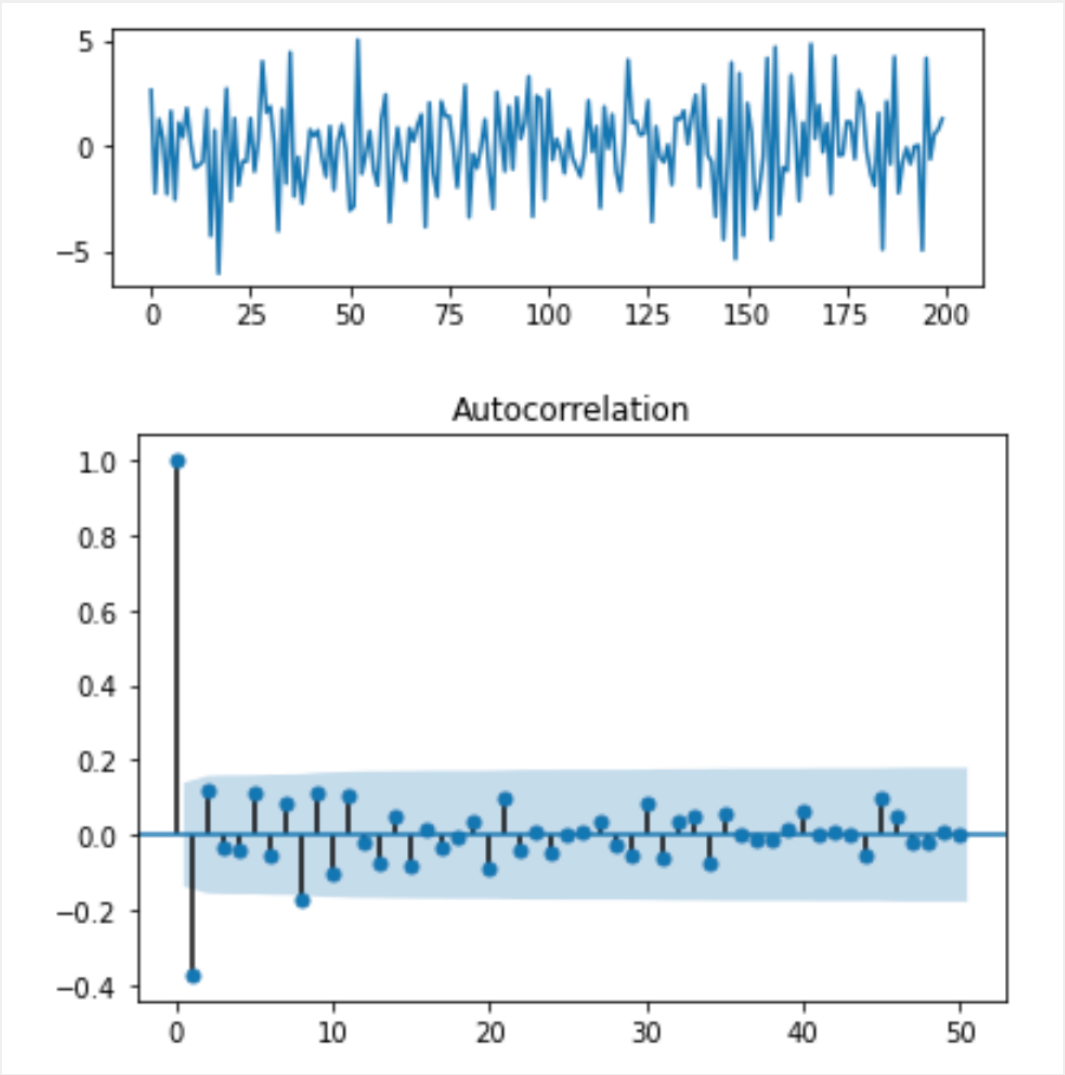
x_{t+h}	x_t
x_3	x_0
x_4	x_1
x_5	x_2

$$\hat{\gamma}(h) = \hat{\gamma}(3) = \frac{1}{6}\left[(x_3 - \bar{x})(x_0 - \bar{x}) + (x_4 - \bar{x})(x_1 - \bar{x}) + (x_5 - \bar{x})(x_2 - \bar{x})\right]$$

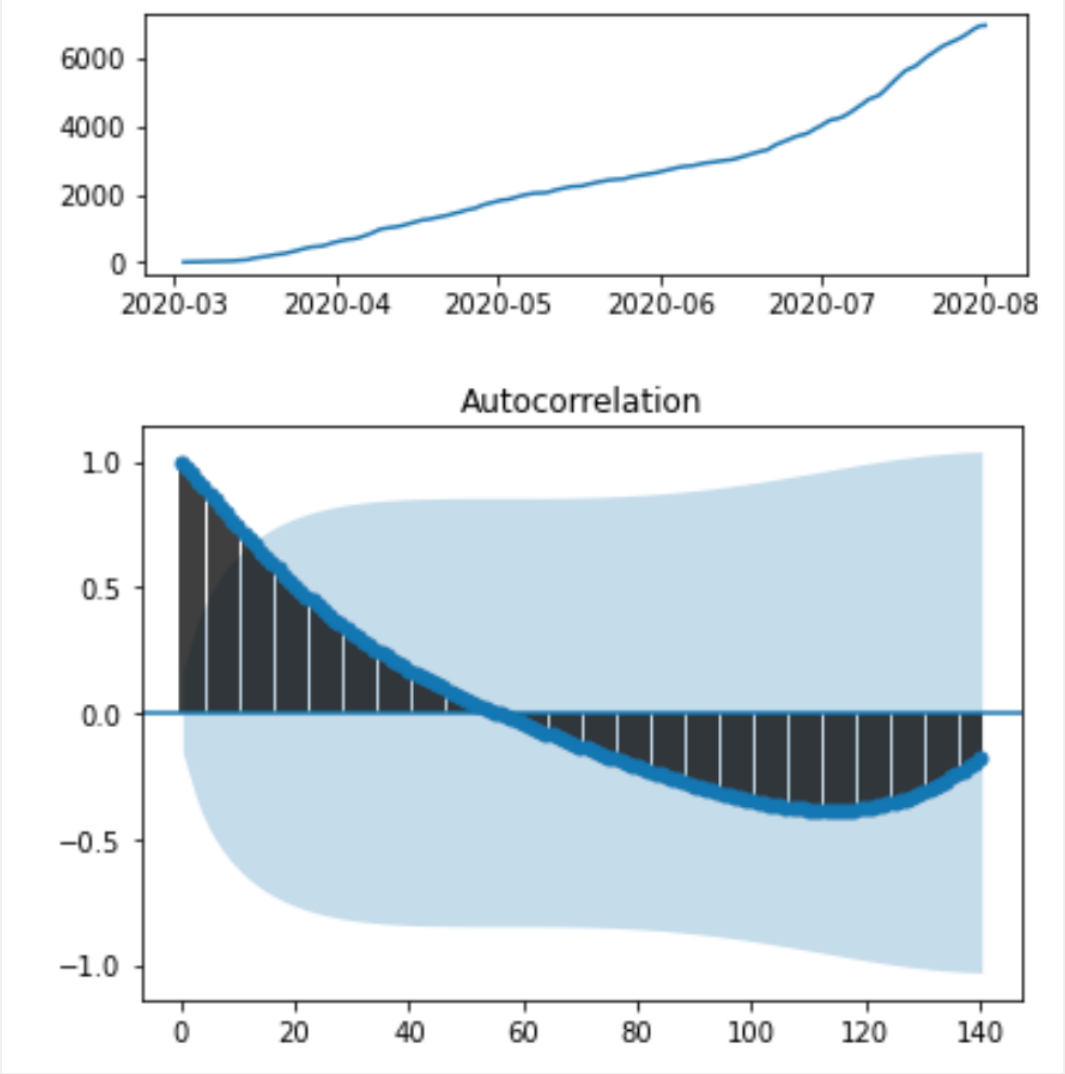
$$\hat{\gamma}(0) = \frac{1}{6}\left[\sum_{i=0}^{5}(x_i - \bar{x})^2\right]$$

The sample ACF for lag=3 is $\hat{\rho}(3) = \frac{\hat{\gamma}(3)}{\hat{\gamma}(0)}$

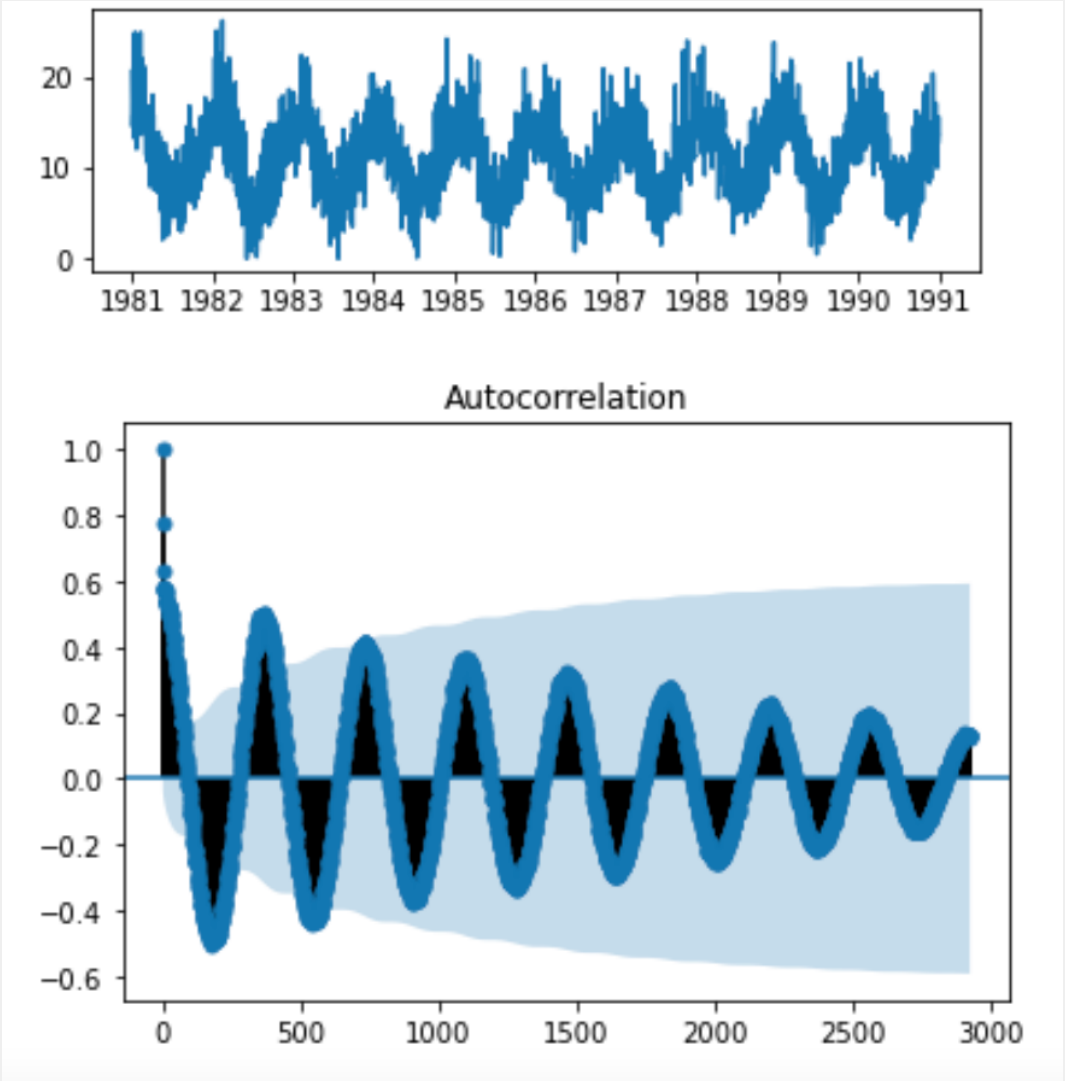
Example: ACF plot for white noise without autocorrelation



Example: ACF plot for ts with linear trend



Example: ACF plot for ts with seasonality

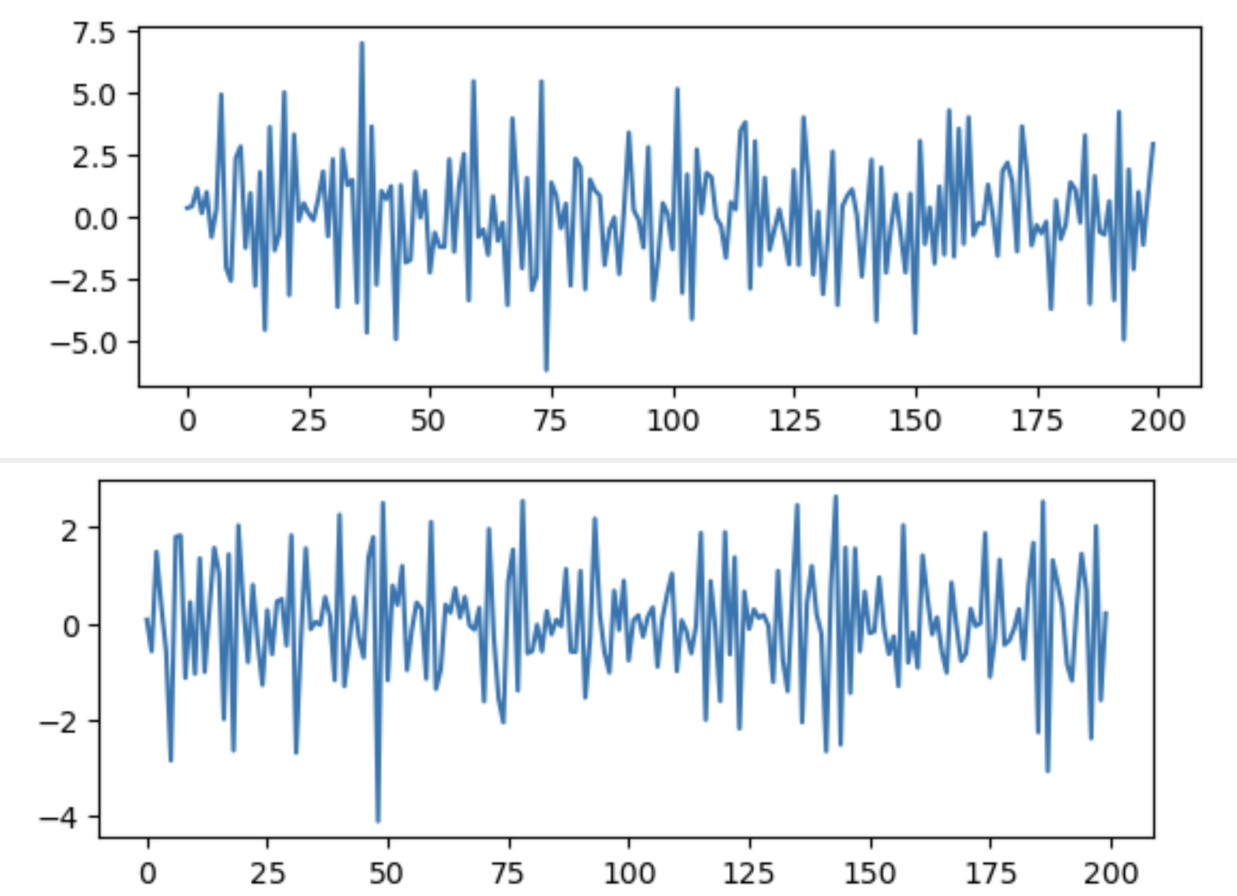


Check for Stationary

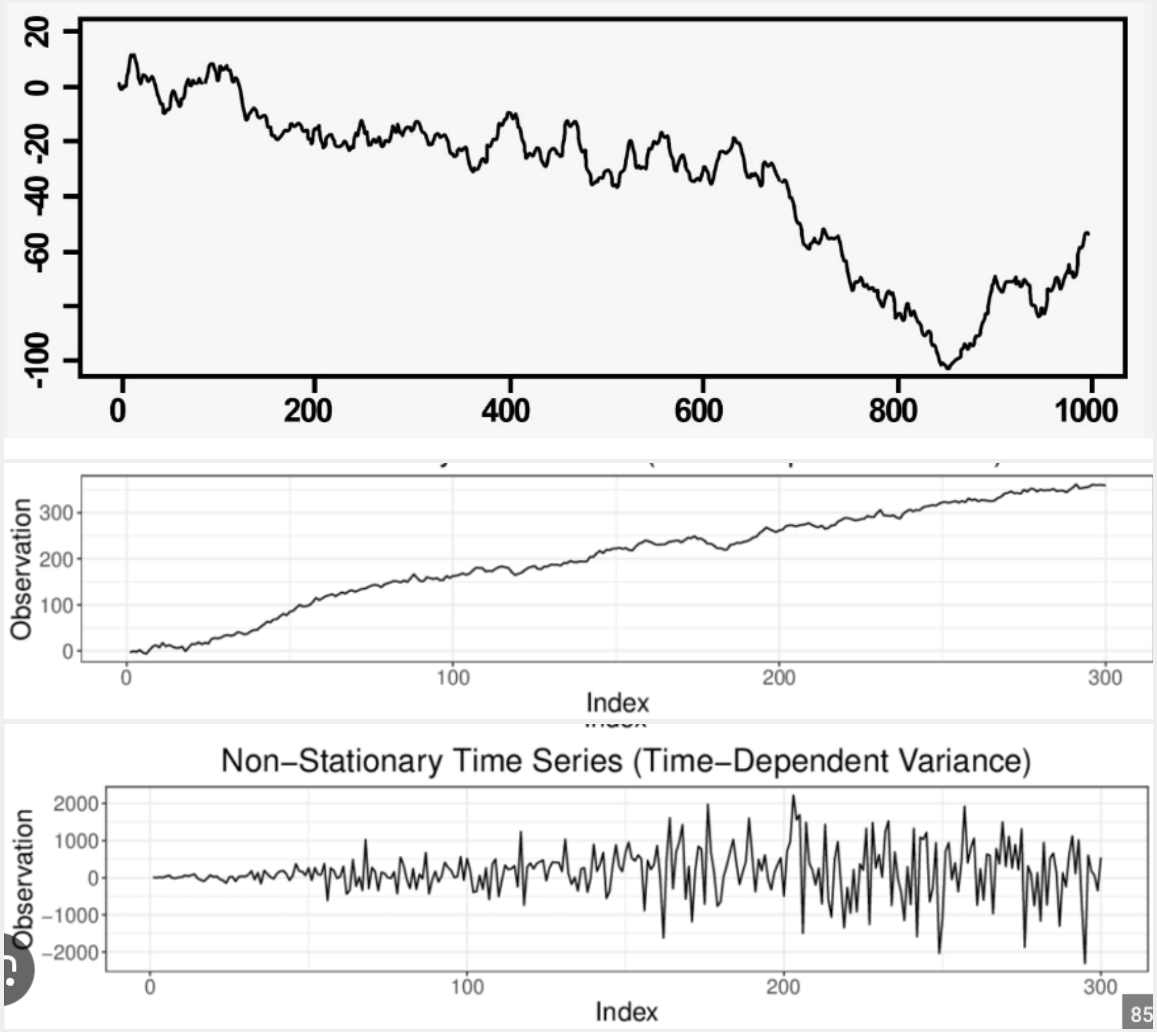
Stationary seems to be a challenging theoretical concept. If given a data, how do we know if it's stationary or not?

1. From time series plot: the data should not show obvious trend (flat); the data should show constant variance (consistent bandwidth); the data should show similar pattern for the window with same period h.

Examples of Stationary Data



Examples of Non-Stationary Data



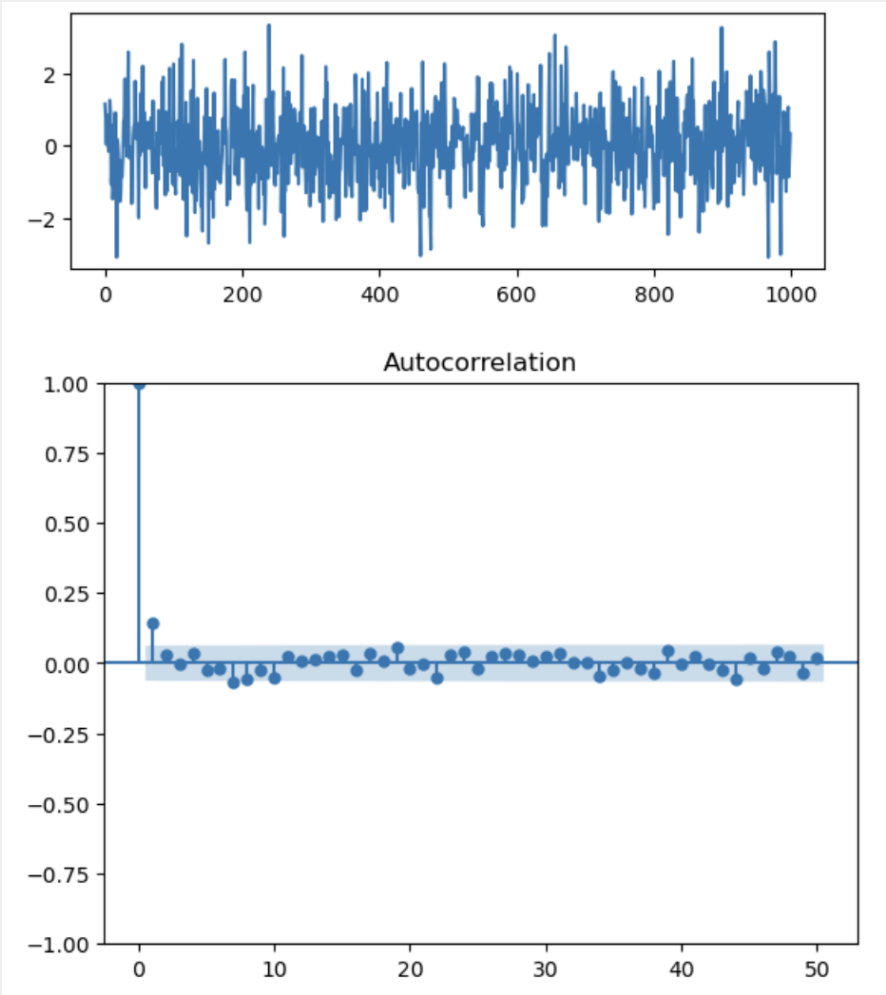
Check for Stationary

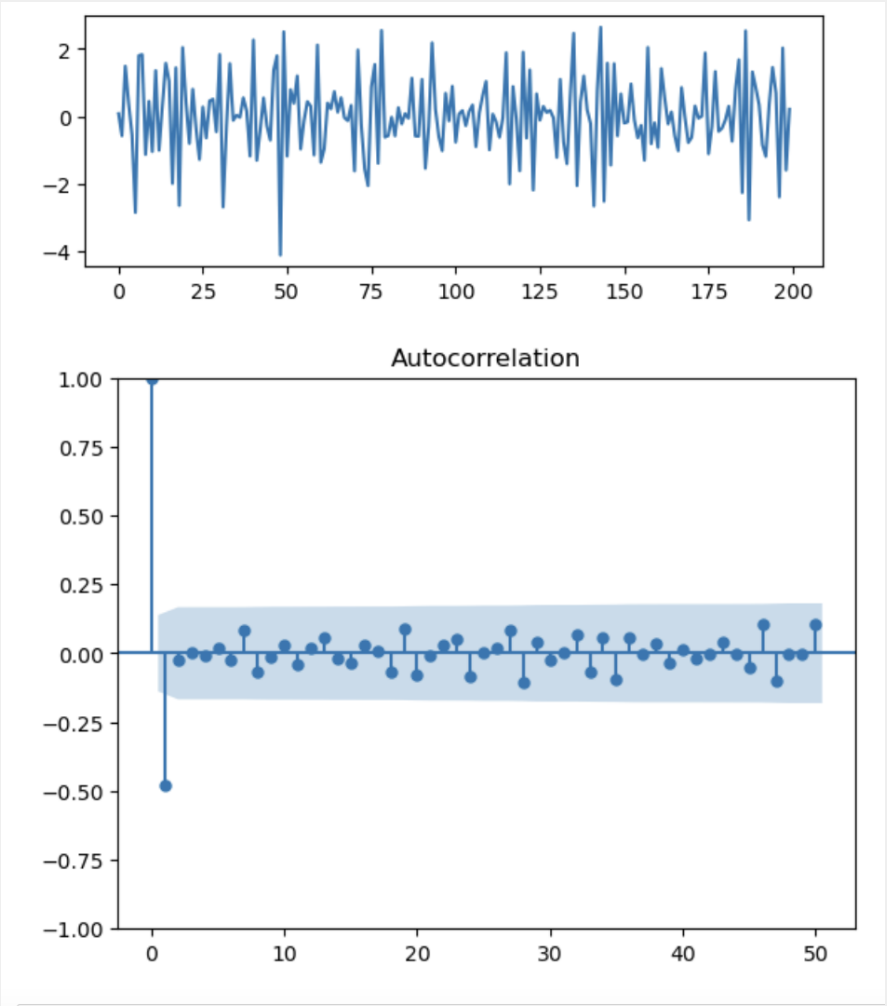
2. From ACF plot: As well as looking at the time plot of the data, the ACF plot is also useful for identifying non-stationary time series.

- For a stationary time series, the ACF will drop to zero relatively quickly (normally for h<10)

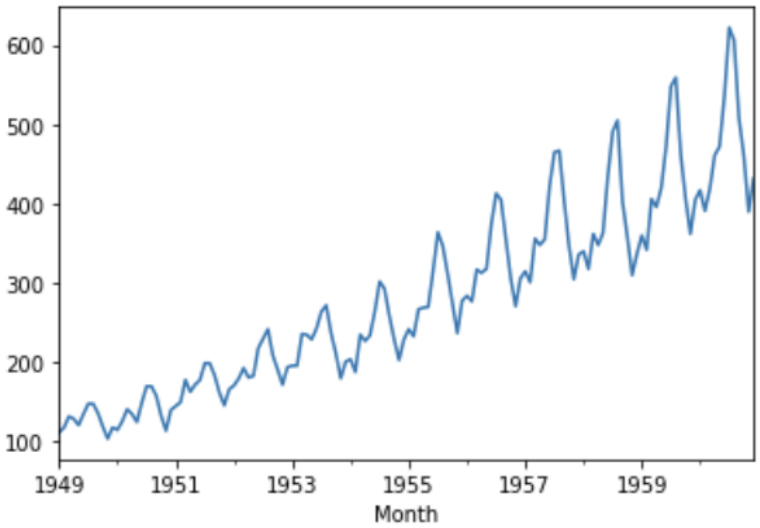
- While the ACF of non-stationary data decreases slowly (see ACF for linear trend).

ACF PLOT FOR STATIONARY DATA

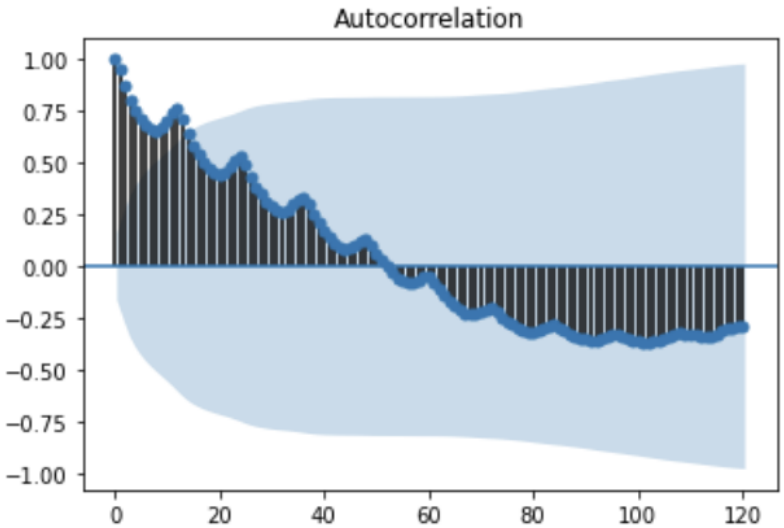


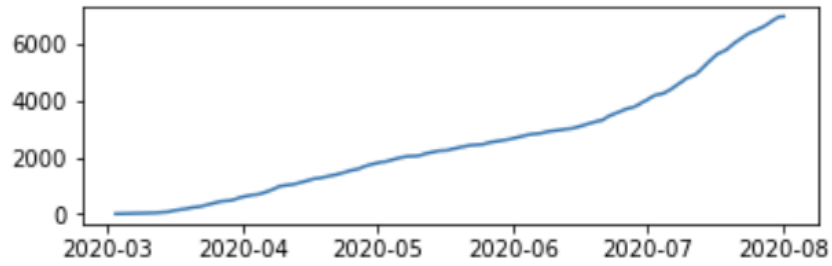


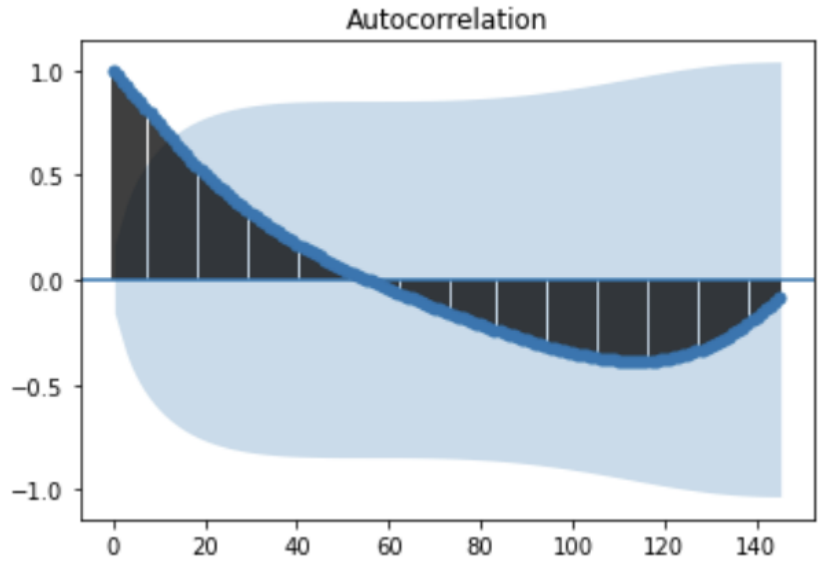
ACF PLOT FOR NON-STATIONARY DATA




```
: plot_acf(series, lags=120)
  plt.show()
```







Hypothesis Test for Stationary [Optional]

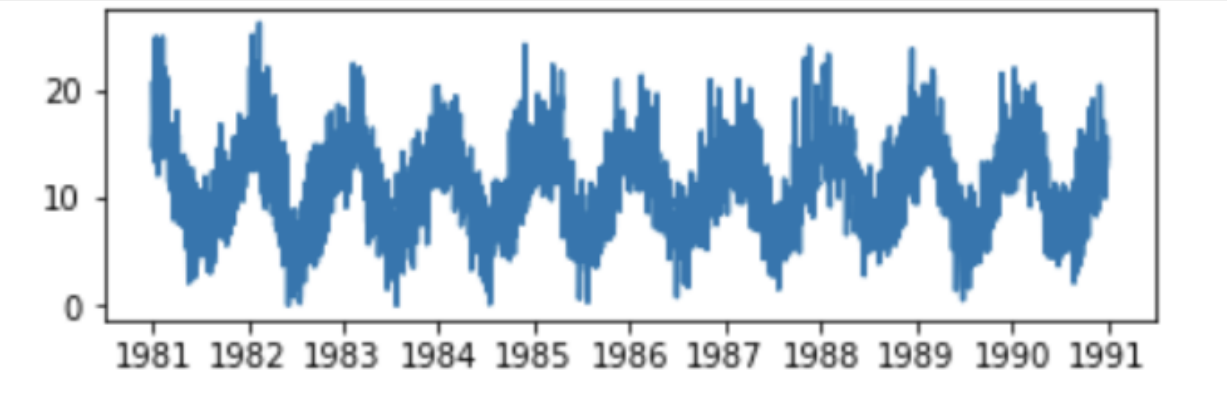
Most common: Augmented Dickey–Fuller test

Idea:

- The Dickey–Fuller tests the null hypothesis that a unit root is present in an autoregressive (AR) time series model. If not (rejection), the time series is stationary.
- Check supplemental materials for details.

Discussion on ts with only seasonality

From the definition of stationary, the plot and test, time series with only seasonality might seem to be stationary:



However, to achieve better models for predictions, we often do not treat it as stationary data but rather model the seasonal component separately.

II. ARIMA

Statistical Modeling for Time Series forecasting

- A time series model for observed series $\{X_t\}$ Is a specification of the joint distribution of $\{X_1, X_2, \cdots, X_t, \cdots\}$, and most models estimate the 1^{st} and 2^{nd} moment of the joint distribution. AKA $E(X_t)$, $\text{Var}(X_t)$, $\text{cov}(x_t, x_{t+h})$.

- Particularly, most models focus on analyzing time series that can be separated to Trend + Seasonality + Remaining Nose.

ARIMA Approach

ARIMA model class takes the approach of analyzing the remaining as Stationary Process, and add back the Trend and Seasonality using differencing method. Recall that stationary process has stable $E(X_t)$, Var (X_t) and cov $(x_t + h, x_t)$ with fixed window doesn't change over time, which makes it easier to model.

- ARIMA(p,d,q) stands for Autoregressive Integrated Moving Average. It has three key components:

  1 AR(p): Autoregression Autoregressive means that we regress the target variable on its own past p values:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + Z_t$$
$$\text{where } \{z_t\} \sim \text{WN}\left(0, \sigma^2\right)$$

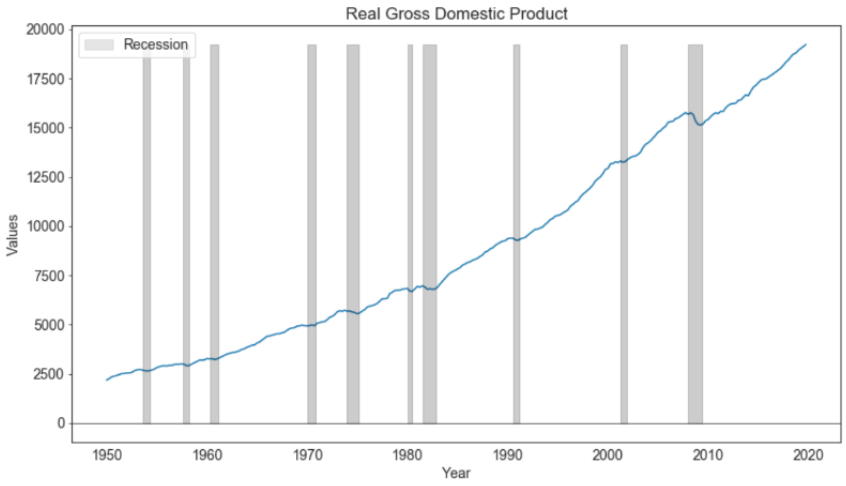
  That's pretty straightforward. All this equation is saying is that the currently observed value of X is some linear function of its past p values (where p is a parameter we choose).

2 MA(q): Moving Average (Not the same as MA smoothing). MA model forecasts X using the model's past q errors:

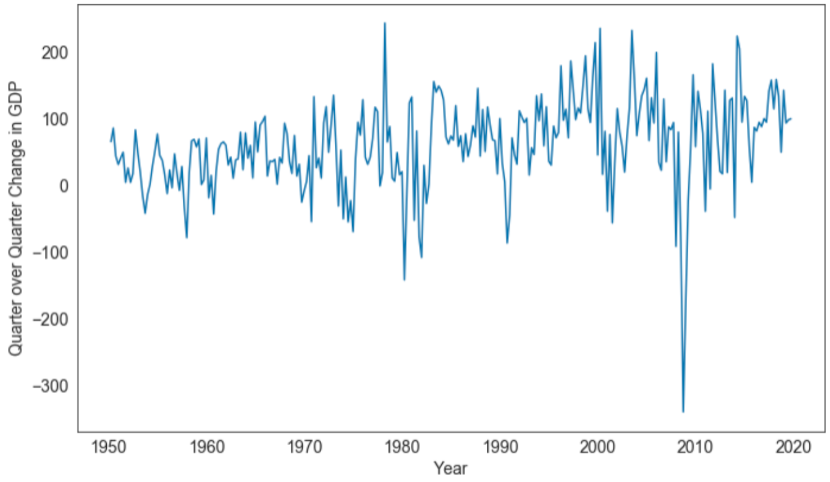
$$X_t = Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2} + \cdots + \theta_q Z_{t-q}$$

The purpose of this is to improve the model at each step based on the previous errors: we know we keep making positive errors, then our model should add some positive value to be more "accurate".

3 I: Integrated with order d: Integrated denotes that we apply a **differencing** step to the data; d is the number of differences needed for stationary.



U.S. Real GDP (source: Federal Reserve Bank of St. Louis)



Change in Real GDP is more stationary

1. ARMA(p,q) for Stationary Time Series

If we examine a data to be stationary therefore no differencing needed, we can use an ARMA(p,q) model to estimate and forecast: $\{X_t\}$ is an Autoregressive Moving-average (ARMA) process with order (p, q) if

$$\begin{aligned} & X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} - \cdots \phi_p X_{t-p} \\ =& Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2} + \cdots + \theta_q Z_{t-q} \end{aligned}$$

$$Z_t \sim WN(0, \sigma^2)^*$$

*White Noise $\{Z_t\} \sim WN(0, \sigma^2)$ is defined as time series that $E(Z_t) = 0$, $\text{Var}(Z_t) = \sigma^2$, $\text{Cov}(Z_{t+h}, Z_t) = 0, \forall t, h$.

Apply ARMA to a dataset

Let's assume now we have stationary data and we want to fit an ARMA model on it, then we need to know

(i) Order Selection: choose appropriate p and q through model selection process (later).

(ii) Model Estimation: $\hat{\phi}_1, \hat{\phi}_2, \cdots, \hat{\phi}_p$, and $\hat{\theta}_1, \hat{\theta}_2, \cdots \hat{\theta}_q$;

(iii) Forecasting: predict x_{t+1} using the model fitted by the historical data $x_1, x_2, \cdots, x_t$;

By default, the ARMA model is estimating the model

$$\begin{aligned}&(X_t - c) - \phi_1(X_{t-1} - c) - \phi_2(X_{t-2} - c) - \cdots \phi_p(X_{t-p} - c)\\ =&Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2} + \cdots + \theta_q Z_{t-q}\end{aligned}$$

You can choose to exclude the constant by choosing *trend* $=' n'$

Example

```
arma_mod20 = ARIMA(dta, order=(2, 0, 0)).fit()
print(arma_mod20.params)
```

```
const       49.746198
ar.L1        1.390633
ar.L2       -0.688573
sigma2     274.727182
dtype: float64
```

Example

```
y = pd.Series(y, index=dates)
arma_mod = ARIMA(y, order=(2, 0, 2), trend="n")
arma_res = arma_mod.fit()
```

```
print(arma_res.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:
Model:                 ARIMA(2, 0, 2)   Log Likelihood
Date:                Tue, 17 Oct 2023   AIC
Time:                        10:31:34   BIC
Sample:                    01-31-1980   HQIC
                         - 10-31-2000
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025
------------------------------------------------------------------------------
ar.L1          0.7905      0.142      5.566      0.000       0.512
ar.L2         -0.2314      0.124     -1.859      0.063      -0.475
ma.L1          0.7007      0.131      5.344      0.000       0.444
ma.L2          0.4061      0.097      4.177      0.000       0.216
sigma2         0.9801      0.093     10.514      0.000       0.797
==============================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
Prob(Q):                              0.96   Prob(JB):
Heteroskedasticity (H):               0.92   Skew:
Prob(H) (two-sided):                  0.69   Kurtosis:
==============================================================================
```

Forecasting

Now we have our estimated model, for example, an $ARMA(2,2)$:

$$X_t - c = \hat{\phi}_1(X_{t-1} - c) + \hat{\phi}_2(X_{t-2} - c) + \hat{\theta}_1 Z_{t-1} + \hat{\theta}_2 Z_{t-2}$$

How do we use this model to forecast the future?

For in-sample estimation, it always works well, because it has the real previous observed value and estimation errors to correct the next estimation:

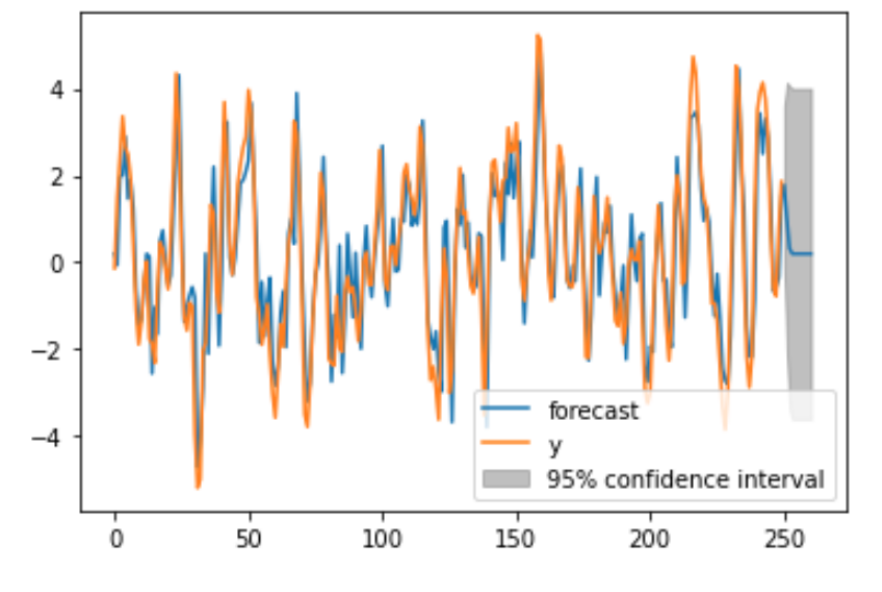
Observed	Estimation	Error
x_0	$\hat{x}_0 = c$	$x_0 - c$
x_1	$\hat{x}_1 = c + \hat{\phi}_1(x_0 - c) + \hat{\theta}_1(x_0 - c)$	$x_1 - \hat{x}_1$
...	...	...
x_n	$\hat{x}_n = c + \hat{\phi}_1(x_{n-1} - c) + \hat{\phi}_2(x_{n-2} - c)$ $+\hat{\theta}_1(x_{n-1} - \hat{x}_{n-1}) + \hat{\theta}_2(x_{n-2} - \hat{x}_{n-2})$	$x_n - \hat{x}_n$

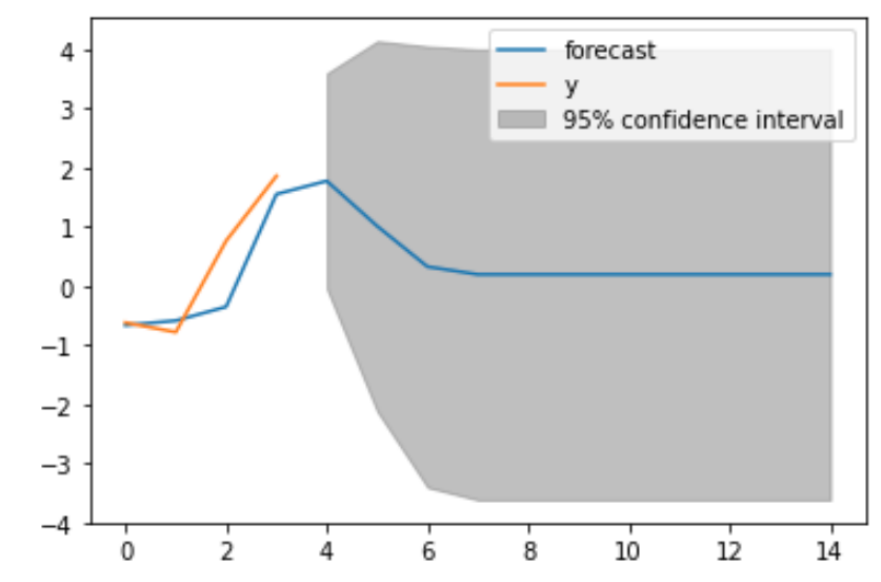
For out-of-sample forecasting, it gradually loses the support of real data:

Observed	Estimation	Error
x_{n+1}	$\hat{x}_{n+1} = c + \hat{\phi}_1(x_n - c) + \hat{\phi}_2(x_{n-1} - c)$	
	$+\hat{\theta}_1(x_n - \hat{x}_n) + \hat{\theta}_2(x_{n-1} - \hat{x}_{n-1})$	0
x_{n+2}	$\hat{x}_{n+2} = c + \hat{\phi}_1(\hat{x}_{n+1} - c) + \hat{\phi}_2(x_n - c)$	
	$+\hat{\theta}_1 * 0 + \hat{\theta}_2(x_n - \hat{x}_n)$	0
x_{n+3}	$\hat{x}_{n+3} = c + \hat{\phi}_1(\hat{x}_{n+2} - c) + \hat{\phi}_2(\hat{x}_{n+1} - c)$	0

ARMA(p, q) is good at predicting short-term future, but long-term forecasting tends to become "flat". it becomes flat faster if orders are small.

EXAMPLE: MA(3) OR ARIMA(0,0,3)





2. ARIMA: TIME SERIES WITH TREND

Idea:

(i) the key method to eliminate trend and seasonality in ARIMA is to difference the original Data

(ii) After differencing, we get stationary data. We fit an ARMA model to the differenced data.

(iii) Add trend and seasonality back by reversing the differencing.

We start with introducing the **ARIMA** model for time series with only trend.

If time series $\{Y_t\}$ exhibits trend but no seasonality:

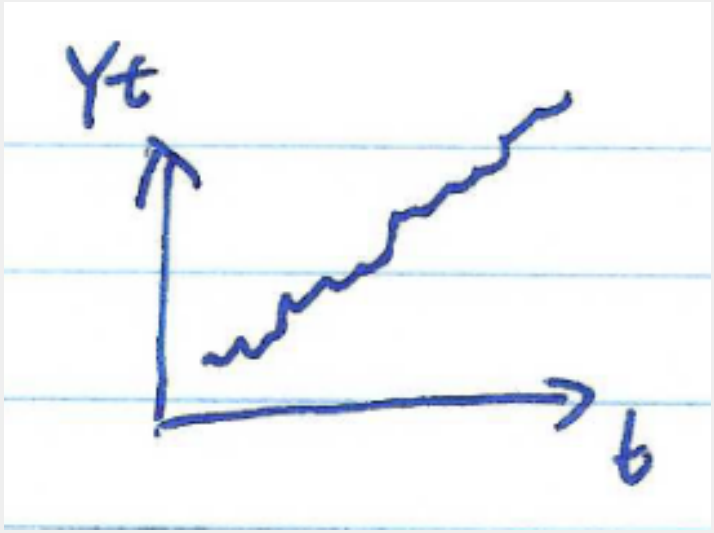
$$Y_t = m_t + R_t$$

- m_t is a trend component that can be estimated by a d-order polynomial function of t:

$$m_t \approx a_0 + a_1 t + \cdots + a_d t^d$$

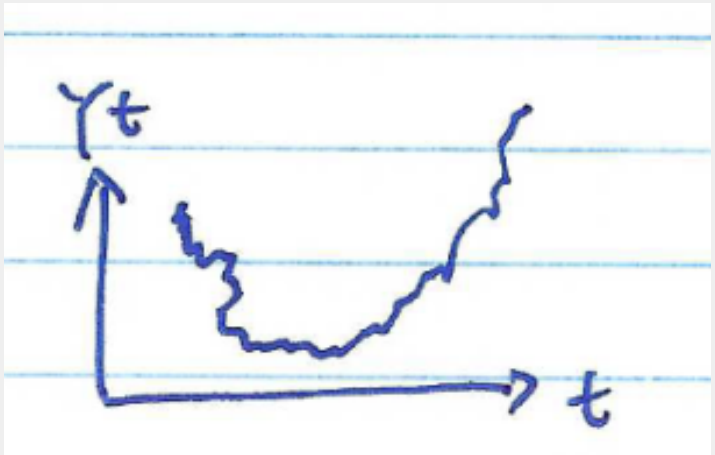
- Then differencing Y_t by d times to get $\{X_t\}$. $\{X_t\}$ is a stationary process.

$d = 1$:



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

$d = 2$:



$$X_t = (Y_t - Y_{t-1}) - (Y_{t-1} - Y_{t-2})$$

statsmodels.tsa.arima.model.ARIMA

```
class statsmodels.tsa.arima.model.ARIMA(
    endog,
    exog=None,
    order=(0, 0, 0),
    seasonal_order=(0, 0, 0, 0),
    trend=None,
    enforce_stationarity=True,
    enforce_invertibility=True,
    concentrate_scale=False,
    trend_offset=1,
    dates=None,
    freq=None,
    missing='none',
    validate_specification=True
) ¶                                                          [source]
```

In particular, in this function, the model being estimated is

$$Y_t - \delta_0 - \delta_1 t - \ldots - \delta_k t^k - X_t \beta = \epsilon_t$$

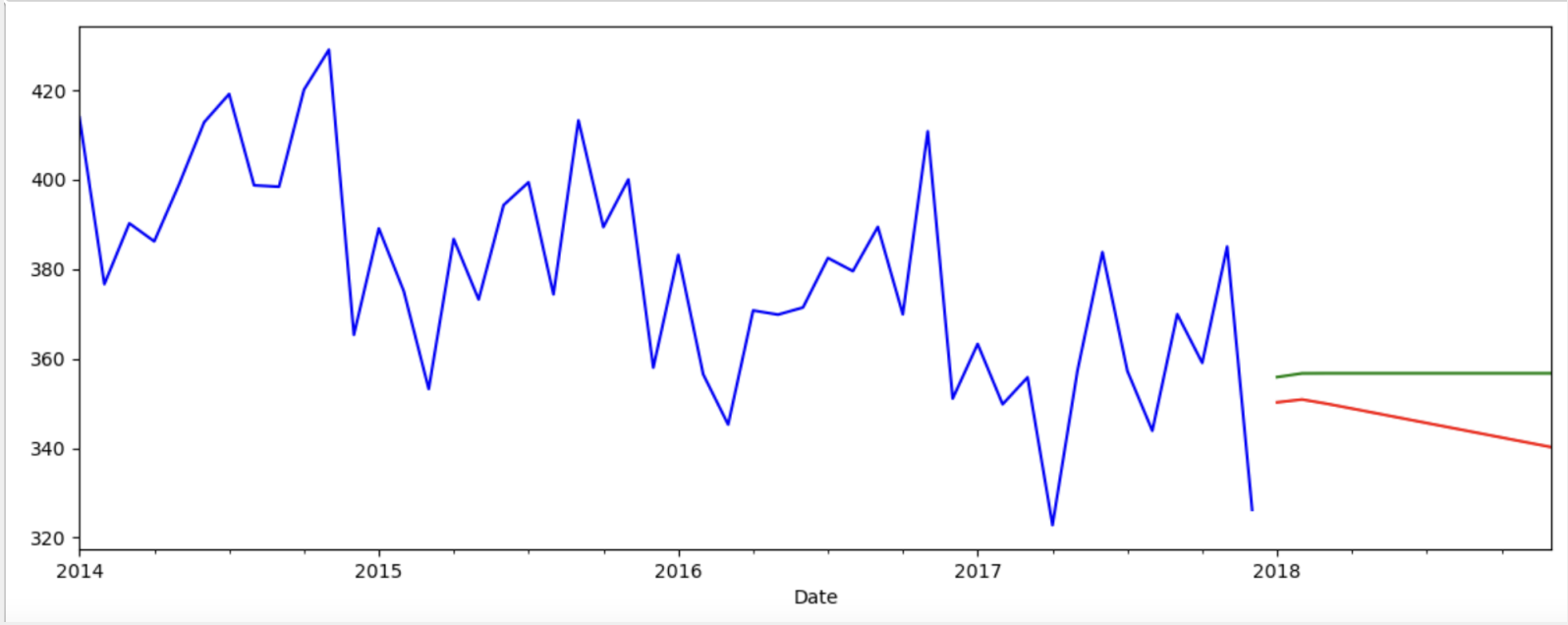
$$\epsilon_t \sim ARIMA(p, d, q)$$

It has added two more components: a general trend controlled by *trend* argument and exogenous regressors.

trend : `str` {'n','c','t','ct'} `or` iterable, `optional`

Parameter controlling the deterministic trend. Can be specified as a string where 'c' indicates a constant term, 't' indicates a linear trend in time, and 'ct' includes both. Can also be specified as an iterable defining a polynomial, as in *numpy.poly1d*, where *[1,1,0,1]* would denote $a + bt + ct^3$. Default is 'c' for models without integration, and no trend for models with integration. Note that all trend terms are included in the model as exogenous regressors, which differs from how trends are included in `SARIMAX` models. See the Notes section for a precise definition of the treatment of trend terms.

```
forecast.plot(kind='line',color='green', label='ARIMA(1,1,1)', ax=ax)
forecast2.plot(kind='line',color='red', label='ARIMA(1,1,1) with t', ax=ax)
plt.show()
```



An alternative function to use is SARIMAX

statsmodels.tsa.statespace.sarimax.SARIMAX

```
class statsmodels.tsa.statespace.sarimax.SARIMAX(
    endog,
    exog=None,
    order=(1, 0, 0),
    seasonal_order=(0, 0, 0, 0),
    trend=None,
    measurement_error=False,
    time_varying_regression=False,
    mle_regression=True,
    simple_differencing=False,
    enforce_stationarity=True,
    enforce_invertibility=True,
    hamilton_representation=False,
    concentrate_scale=False,
    trend_offset=1,
    use_exact_diffuse=False,
    dates=None,
    freq=None,
    missing='none',
    validate_specification=True,
    **kwargs
```

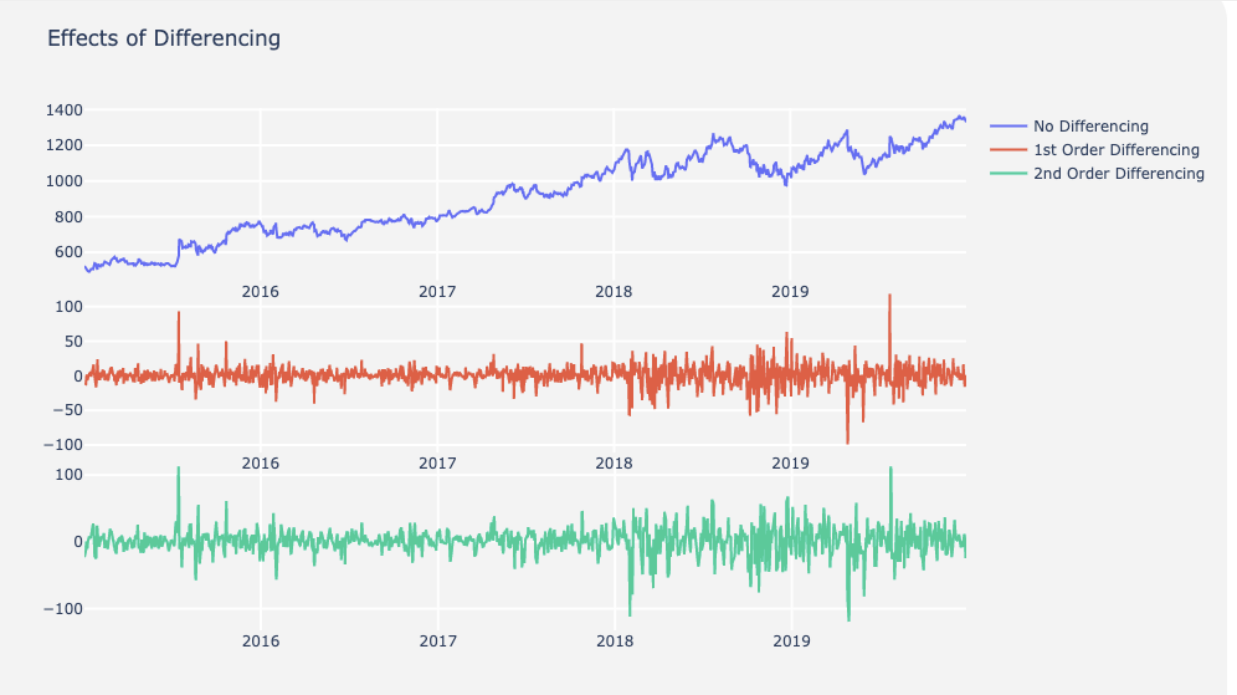
[source]

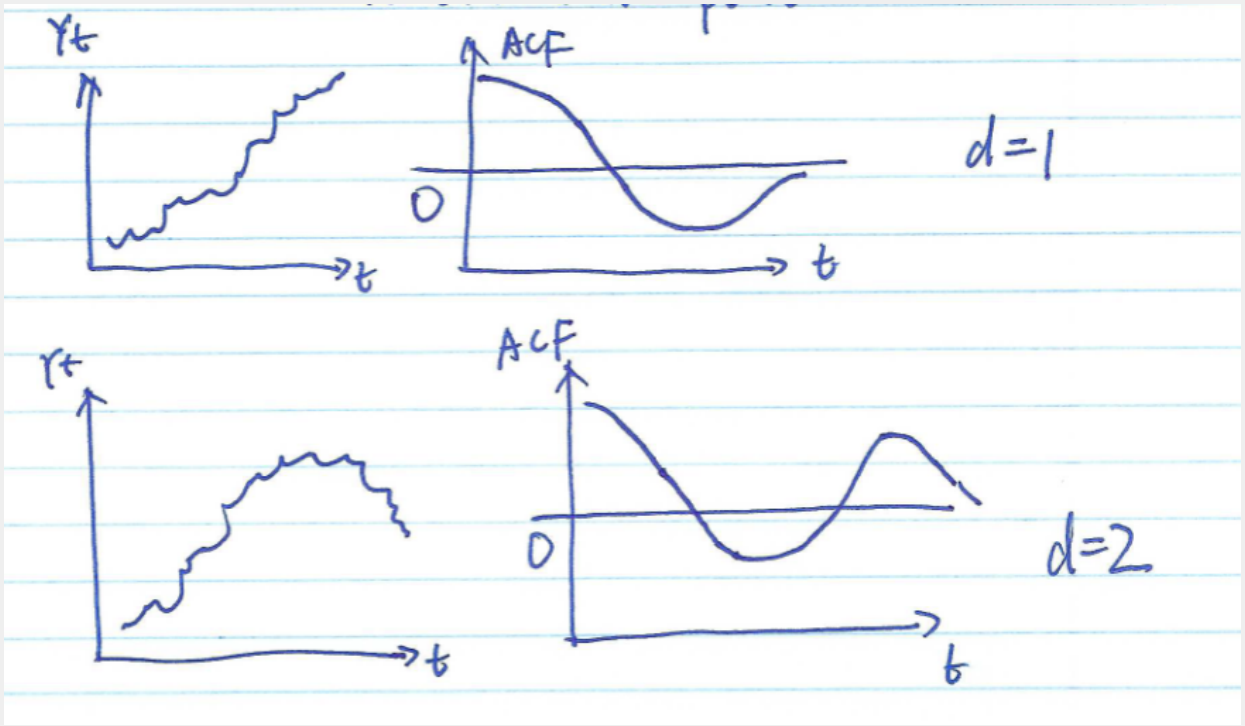
Even considering the trend, ARIMA model will still prioritize the closest historical data and, therefore sometimes does not continue the general trend we have observed from the historical data.

How to choose the differencing order?

How to choose d?

(1) We can use a combination of plots and stationary checks: keep differencing until the data reaches stationary.

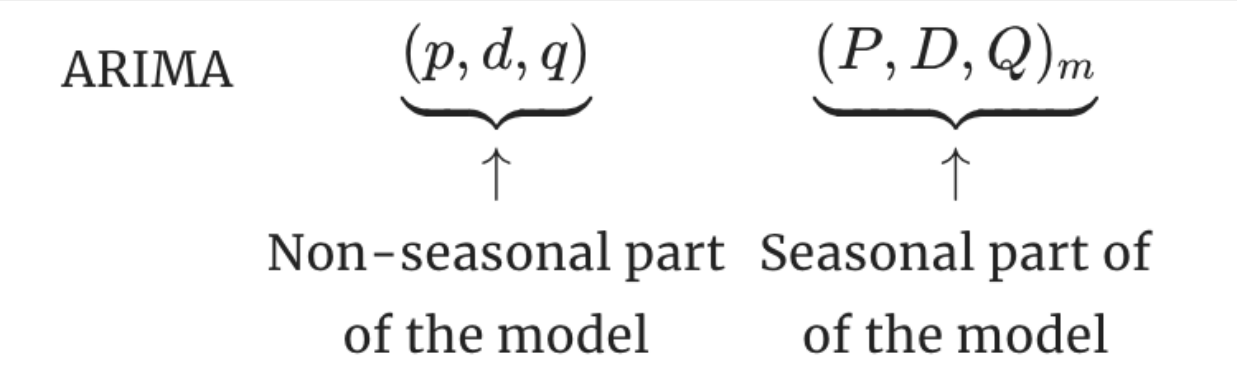



Yt
t
ACF
0
t
d=1
Yt
t
ACF
0
t
d=2

(2) Use order selection methods to search for the best d order. (most common, later)

SARIMA: Time Series with Trend and Seasonality

A SARIMA$(p, d, q)(P, D, Q)_m$ process has



- The trend component with parameters (p, d, q)

- The seasonal component with parameters $(P, D, Q)_m$ where
  - m : the number of time steps for a single seasonal period (how long one season pattern is)
  - D : number of seasonality differencing needed to eliminate seasonality and reach stationary on the de-trended data

Additive Seasonality

Additive seasonality assumes the seasonal magnitude from the same season stays the same over time. In this case, for the de-trended data, only differencing once eliminate the seasonality and reach stationary. Therefore D=1.

Multiplicative Seasonality

$$Y_t = m_t \cdot S_t \cdot R_t$$

If we apply the natural-log transformation on Y_t:

$$\ln Y_t = \ln m_t + \ln S_t + \ln R_t$$

then $\{\ln Y_t$ should have additive seasonality. In this case, the analysis will be on $\{\ln Y_t\}$ and we still choose D=1.

In practice, the two cases:

- Modeling $\{Y_t\}$ with $D = 1$ for approx. additive seasonality
- Modeling $\{\ln Y_t\}$ with $D = 1$ for approx. multiplicative seasonality then transform back to get forecast.

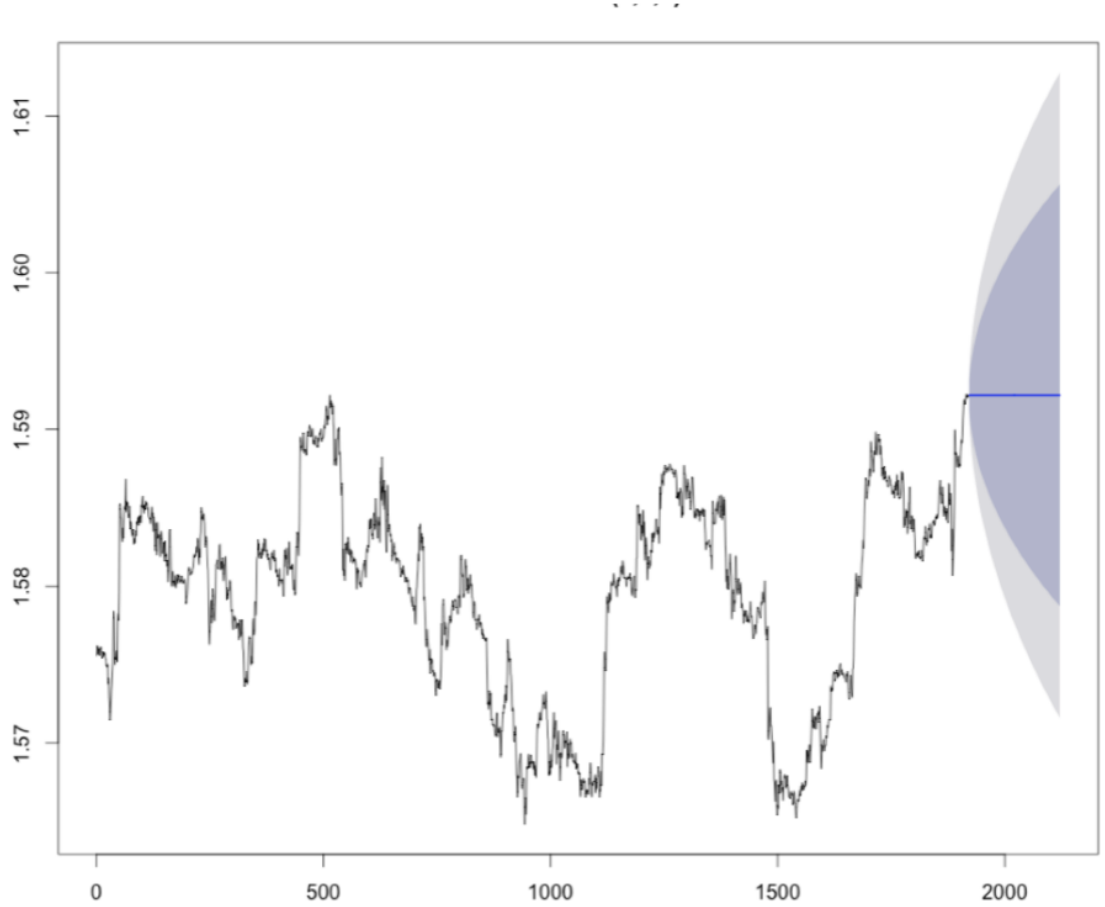
covered most of the applications using SARIMAX.

Disadvantage of ARIMA family

1. It's hard to understand mathematically, therefore, hard to choose appropriate parameter values unless doing a grid search

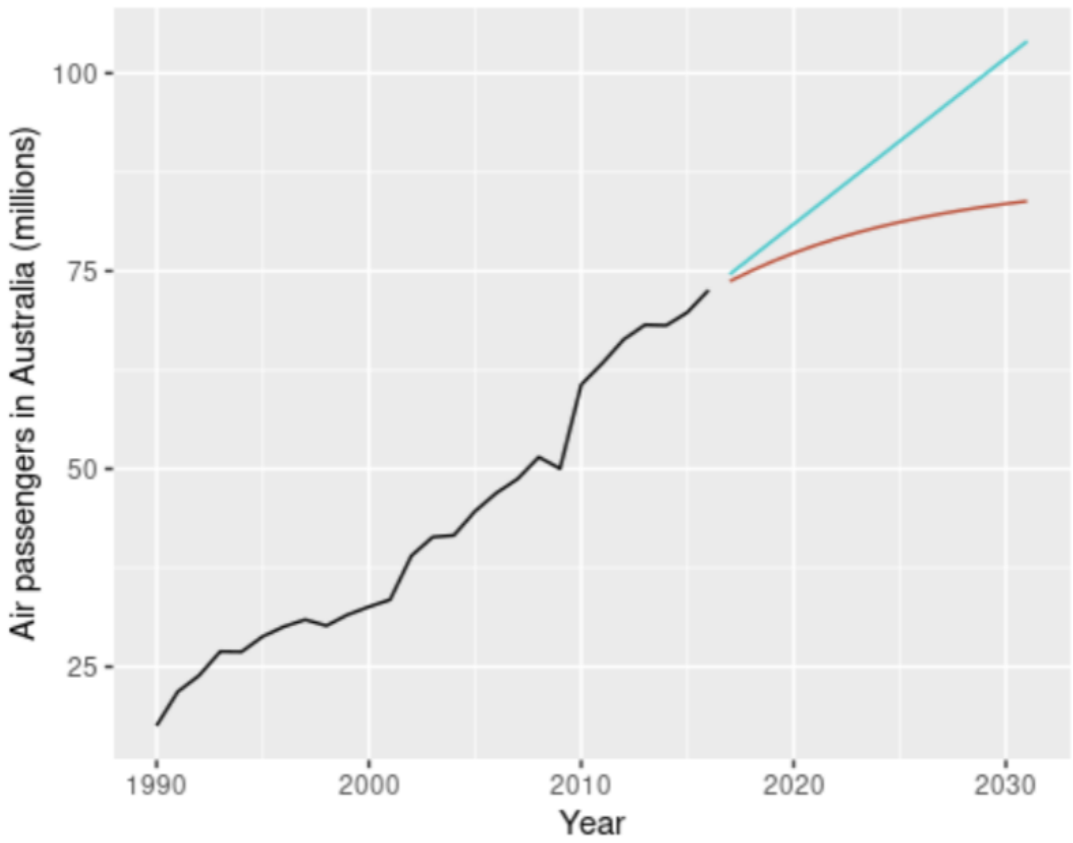
2. When fitting the data with different orders, the forecasting performance can look very different and surprising: it's all about how much the forecast depends on the past values. We thought it should be an upward trend following the majority of the trend, but it's a flat line because it only depends on one previous observation. We see a lot of "flat" forecasting .

Example: prediction of ARIMA(0, 1, 1)



3. There are some special cases we can look at the behavior, but more generally, it relies heavily on the order selection strategy: what prediction metric to use? how to define train and validation set? how to define the cross-validation, can all give you different result.

Example: prediction of ARIMA(0, 2, 2) and ARIMA(1, 1, 2)



SARIMAX

SARIMA model assumes Y_t is only related to its historical values. If there are exogenous predictors X_1, ..., X_k, we can consider using the SARIMAX model. Then the model for Y_t is separated into two parts:

$$Y_t = \beta_0 + \beta_1 X_{1,t} + ... + \beta_1 X_{k,t} + \epsilon_t$$

$$\epsilon_t \sim SARIMA(p, d, q)(P, D, Q, m)$$

In Python, as discussed earlier, the ARIMA function takes this function further and adds a polynomial function to model the general trend:

$$Y_t - (\delta_0 + \delta_1 t + \ldots + \delta_k t^k) - (\beta_0 + \beta_1 X_{1,t} + \ldots + \beta_1 X_{k,t}) = \epsilon_t$$

$$\epsilon_t \sim SARIMA(p, d, q)(P, D, Q, m)$$

The data for Y_t, $X_{t,1}$, ..., $X_{t,k}$ will be collected together at each time step:

	realgdp	realcons	unemp
1959-03-31	2710.349	1707.4	5.8
1959-06-30	2778.801	1733.7	5.1
1959-09-30	2775.488	1751.8	5.3
1959-12-31	2785.204	1753.7	5.6
1960-03-31	2847.699	1770.5	5.2
...	...	...	...

Note

To use the SARIMAX to forecast the future values of Y_t, we need to provide the future values of regressors.