# **Autocorrelation**

* One approach to finding the value **m** is that we find local maxima and minima in the time series, and try to analyze the intervals at which they are observed.
  + This approach makes sense.
* Another approach would be; given a time series 𝑦(𝑡),
  + What if we consider another time series where we introduce a **lag** of 1, i.e. **shift the series** by 1 unit of time: 𝑦`(𝑡)
  + We can then find the **correlation coefficient** between these two-time series: 𝑦(𝑡) and 𝑦`(𝑡).
  + Similarly, we find the correlation between the original time series 𝑦(𝑡) and a time series lagged by i units: 𝑦i(𝑡), where;

𝑖=1,2,3,... (i represents lag)

* + In doing so, we would find a value of 𝑖, where the lagged time series **roughly overlaps** over the original series.

A blackboard with colorful writing on it

Description automatically generated

* Then, we create a table containing: 𝑦(𝑡) and 𝑦i(𝑡) time series values
* When ***i=m,*** (the lagged time series roughly overlaps over the original series), the Pearson correlation coefficient would be very close to 1.
* This value of 𝑖 would indicate the optimal value of 𝑚.

A blackboard with white text and orange lines

Description automatically generated

* **What are the advantages of using the concept of correlation for finding the optimal value of m?**
  + Easily interpretable
  + Value ranges from -1 to +1
  + Captures linear relationship

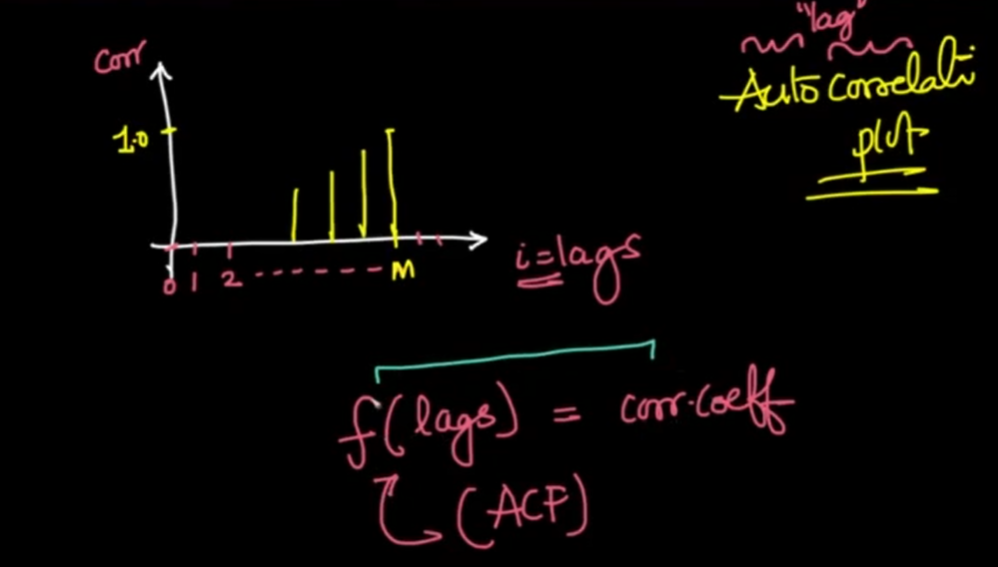
**Q. What will the plot between the correlation coefficient and lag (i) look like?**

* At 𝑖=𝑚, we would get a correlation value very close to 1.
* At 𝑖=𝑚/2, as the value of lagged time series increases, the value of the original series decreases, giving us a **strong correlation.**
* For a value of 𝑖 that is even close to 𝑚, though the correlation value would not be as strong as at 𝑚, it would be relatively strong.

A diagram of a graph

Description automatically generated

* So, the final plot between lag value (i) and correlation coefficient would look something like this:-



* This plot is called an **Autocorrelation Plot** because we are computing the correlation of the series with itself with various values of lag.
* This can be written as a function also. That is called the **Autocorrelation Function (ACF)**.
* The ACF plot shows both the direct and the indirect impact of the correlation values. It takes the average correlation between the time series original values and the lag values for different lags.
* If the time series is random the lag values will not have much autocorrelation with the original values.

### **ACF Plot of Trended Series:**

A graph of a graph

Description automatically generated

### 

### **ACF Plot of De-Trended Series:**

A graph with blue lines and dots

Description automatically generated

* After de-trending the series, we can see the correlation of the series is high at periods 6 and 12. This indicates there is some seasonality.
* Besides that, the correlation values are random and small.
* On removing the trend, we're **able to capture negative correlations** between the original series and lagged series with more ease.
* The blue color highlighted part is the **confidence interval** which gives the significance level of the correlation
* If the correlation is higher or outside of the blue highlighted area, that can be considered as a highly significant value.
* So the major values of lag we consider to study the seasonality pattern here are:-
* Correlation at lag=1
* Correlation at lag=6
* Correlation at lag=12
* Correlation at lag=24
* Other points lie within or very close to the blue shaded region of confidence intervals.
* This means that we are not very confident about these correlation values
* The confidence interval increases as we consider time series that lagged more.

# 

# **Partial Autocorrelation**

* This is similar to AutoCorrelation with only a small difference.
* We try to find a relationship between the original time series 𝑦(𝑡) and time series lagged with 𝑖 steps 𝑦𝑖(𝑡), where 𝑖=1,2,3,..., to find the optimal value of 𝑚.
* This plot only explains the values which have a direct impact on the original values.
* The difference is that: **All intermediate/indirect correlations are removed**.
  + The correlation between observations at successive time steps is a linear function of the indirect correlations.
  + These indirect connections are eliminated using the **partial autocorrelation function (PACF)**.

For example

* When we're considering the correlation between 𝑦(𝑡) and say, 𝑦12(𝑡),
  + Then, we do not want this correlation value to get corrupted by the correlations when 𝑖=1,2,3,...,11 (i.e. the intermediate correlations)
* The partial correlation for each lag is the **unique correlation** between the two observations after the intermediate correlations have been removed.
* This is also known as **Conditional AutoCorrelation**.

### **PACF Plot:**

A graph with blue lines and numbers

Description automatically generated

* Here, one can see:
  + Value at 𝑖=1 is high: This means that given the values of time series 𝑦(𝑡), we can compute the values of a time series with lag 1, 𝑦1(𝑡) with ease.
  + There is a high value at 𝑖=12
    - This means that even if we ignore the information given by all the time series with 𝑙𝑎𝑔𝑠=1,2,...,11, the information carried in time series 𝑦12(𝑡) alone is very high.
    - This shows strong seasonality.
* We calculate PACF on the original time series, whereas ACF is plotted on stationary time series.

# **Correlation Vs Causation**

* If a variable x is correlated with another variable y, does that mean that x causes y?
  + A variable x may be useful for forecasting a variable y, but that does not mean x is causing y.

For example,

* + It is possible to model the number of drownings at a beach resort each month with the number of ice creams sold in the same period.
  + The model can give reasonable forecasts, not because ice-creams cause drownings, but because people eat more ice creams on hot days when they are also more likely to go swimming.
  + So the two variables (ice cream sales and drownings) are correlated, but one is not causing the other. They are both caused by a third variable (temperature). This is an example of **confounding**.
* Correlations are useful for forecasting, even when there is no causal relationship between the two variables.
  + For example, It is possible to forecast if it will rain in the afternoon by observing the number of cyclists on the road in the morning.
  + When there are fewer cyclists than usual, it is more likely to rain later in the day.
  + This model can give reasonable forecasts, not because cyclists prevent rain, but because people are more likely to cycle when there is less or no chance of rain.
  + In this case, there is a causal relationship, but in the opposite direction to our forecasting model.
* Though we can get good forecasts based on correlated variables, if we try and understand the causality behind those variables, we can identify better features, thereby creating a better model.
  + A better model for the example of the drowning will probably include temperatures and visitor numbers and exclude ice cream sales.
  + A good forecasting model for rainfall will not include cyclists, but it will include atmospheric observations from the previous few days.

# **ARIMA Family of Forecasting Techniques**

## **Auto Regression (AR)**

* For stationary time series, one is helpless because of the absence of a pattern in the series.
* Any trend or seasonality that was present has been removed from the series, and it will be added back later, in the final prediction.
* In AR​​ we forecast the variable of interest using a linear combination of past values of the variable.
* For such cases we study a new family of models called **ARIMA**.
* Though such series look like they are completely random, there is still some extent of forecastability here, there is still information left to be extracted from stationary series.

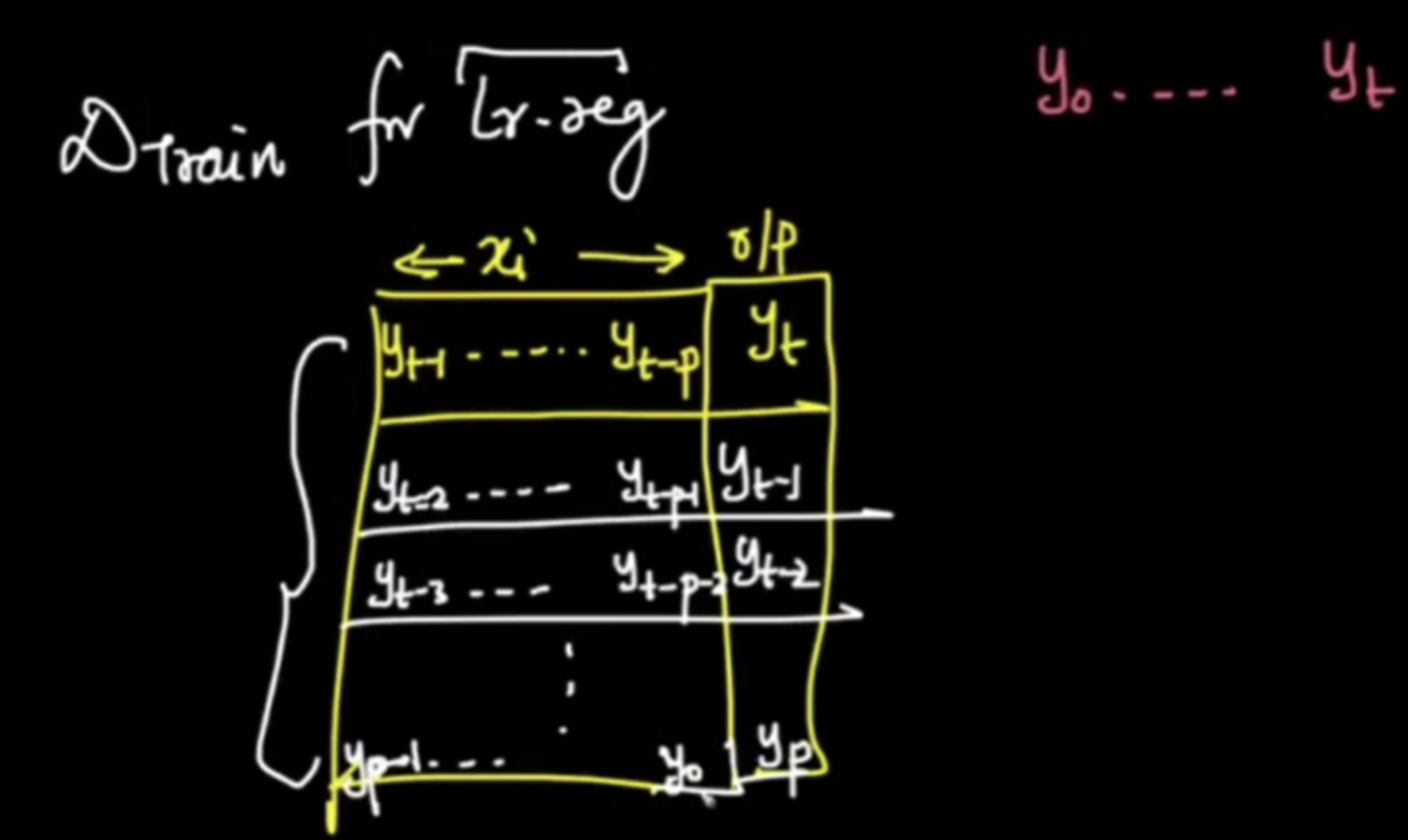
**What if we had a feature in our time series, besides the value to be predicted?**

* In that case, one could just utilize Linear Regression, by mapping this feature's values with the value to be forecasted.
* For creating a new feature, we can map the value of the stationary time series at time

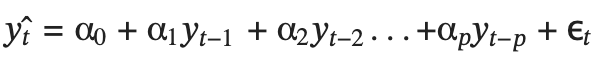
𝑡 with the value of series at time 𝑡−1,𝑡−2,𝑡−3,...,𝑡−𝑝, where p could be a **hyperparameter** we set.

* This way, now our data becomes as shown.
* It contains date as the index,
* Past values 𝑦𝑡−1, 𝑦𝑡−2, ..., 𝑦𝑡−𝑝 as features and
* value at time 𝑡 as the value to be predicted (𝑦^t)

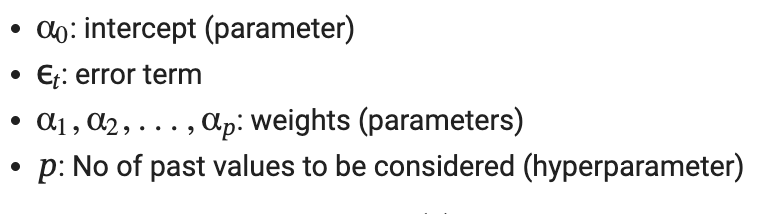
Now we can successfully implement Linear Regression using these features.



* Since the aim is to convert the forecasting problem to Linear Regression, what we’re doing is;
  + Future value 𝑦̂𝑡  = LinearRegression(Past p values)
* Thus, the forecasting problem is converted into the following form:



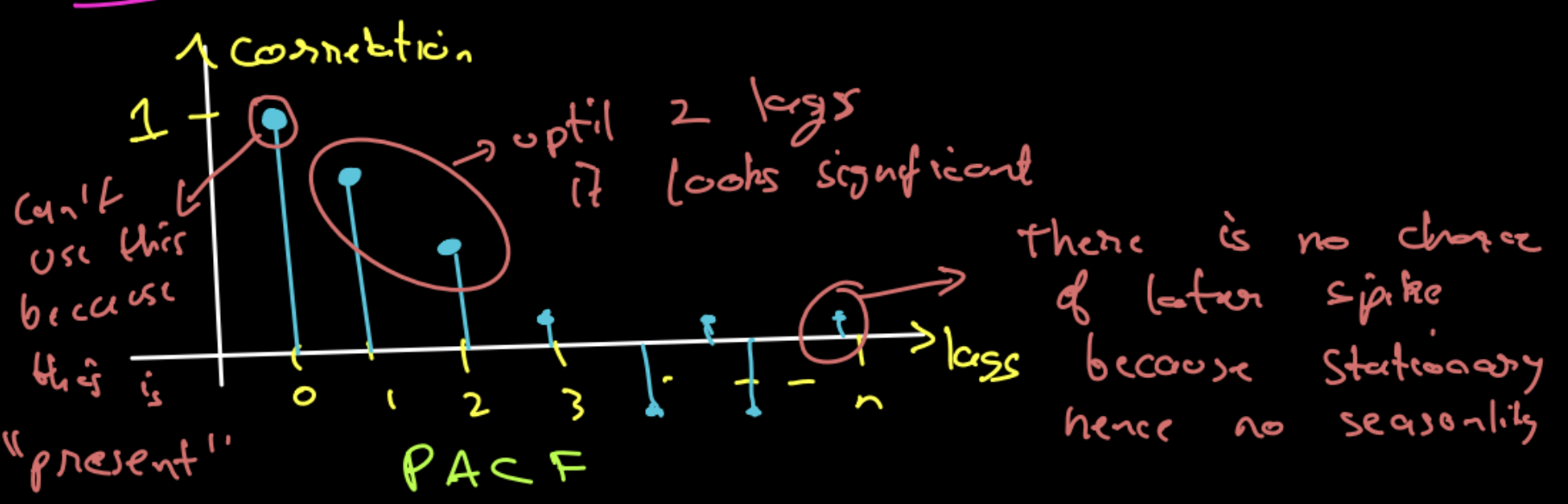
where;



* is a residual term considered as a purely random series with mean 0, variance and covariance is 0.
* This seems similar to Simple Exponential Smoothing (SES).
* Though we are essentially doing a **weighted average** of the past time series values in both SES and AR models, there is a fundamental difference.
  + In the case of SES,
    - The weights are **exponentially decaying**
    - The hyperparameter is **α**
  + In the case of **AR**,
    - The weights are **learnt** by multiple iterations.
    - The hyperparameter is 𝑝
* **Pre-requisites of the AR model:**
  + The idea is that this assumption will be true if the Partial Auto Correlation(PACF) Plot has a high value at lag = k.
  + This way, from the plot we know that the future value is highly correlated with one value in the past, which means that it makes sense for us to compute linear regression on the past 1 value.
  + We are using PACF because we don't want features to be correlated with each other in LR. PACF can help to identify that

**Deciding the value of p**

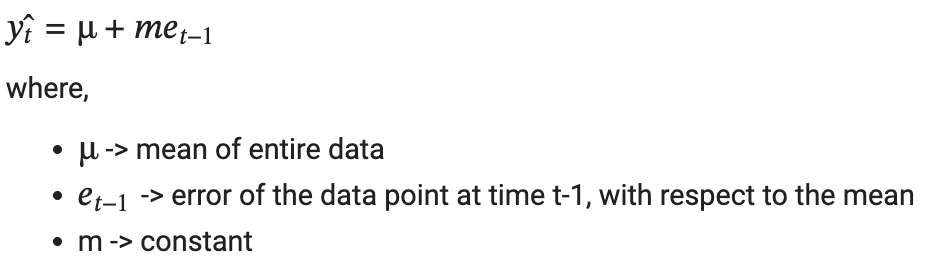
* We look at the PACF plot, and based on the values there, we decide how many lag values we can consider.
* For example, in the given plot,
  + we consider 2 lags only,
  + as for the third lag, the PACF value is 0.1, so considering it will not give as promising results.



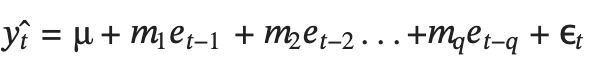
* The simplest AR process is AR(0), which has no dependence between the terms. In fact, AR(0) is essentially white noise.
* We can select the order 𝑝 for AR(𝑝) model based on significant spikes from the PACF plot.
* One more indication of the AR process is that the ACF plot decays more slowly.

## **Moving Averages (MA)**

* The idea here is to use the value of the series at time t-1, we use the error of value at t-1 from the mean of data in the regression setup
* The error of each data point in the series from the mean/average, would be different.
* This should also work, as we are able to successfully create a new feature that is unique for each point.
* In fact, this idea is called the **Moving Averages (MA)** technique.
* Moving Average process considers the past residual values to predict the current time period values.
* One problem of the AR model is the ignorance of correlated noise structures (which are unobservable) in the time series.
* Contrary to the AR model, the finite MA model is always stationary because the observation is just a weighted moving average over past forecast errors.
* Though the name is the same as the smoothing technique, this has nothing to do with that and is a completely different concept.
* The formulation of MA is as follows:



* This idea can be extended to the order of **q.** In that case, the formulation becomes:



Here, **ε** represents the final error remaining that is actually truly random, which we cannot help. This is also added for representation.

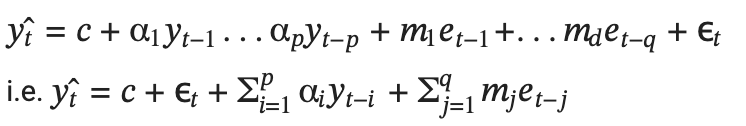
**NOTE:**

* q becomes the **hyperparameter** for the MA(q) model
* In the case of MA, there is a fixed way to determine the value of q, we need to try a bunch of different values to find the best fit.
* ACF provides a considerable amount of information about the order of the dependence q for the MA(q) process.
* Identification of an MA model is often best done with the ACF rather than the PACF.
* In contrast to the AR model, we can select the order q for model MA(q) from ACF if this plot has a sharp cut-off after lag q.
* One more indication of the MA process is that the PACF plot decays more slowly.

## 

## **Auto Regression - Moving Averages (ARMA)**

* The combined technique of Auto Regression (AR) and Moving Averages (MA) is called the ARMA model.
* While combining the two ideas, **p**: order of AR and **q**: order of MA, **p** may or may not be equal to **q**
* α1, α2,..., α𝑝: coefficients of AR
* 𝑚1, 𝑚2,..., 𝑚𝑞: coefficients of MA
* Hence the formulation becomes:



* Here, **p** and **q** are hyperparameters. Thus it is also called **ARMA(p,q)** model.
* The major limitation of this technique is that it cannot handle non-stationary time series because, if we're training a Linear Regression, the variables can not be dependent on each other.
* This method can not handle if there is a trend or seasonality present in the data.
* For the AR term refer to the PACF plot which tells about the lag terms and for the MA term refer to the ACF plot which tells about the error terms.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **AR(p)** | **MA(q)** | **ARMA(p,q)** |
| **ACF Plot** | Slow Decay | Sharp Cutoff | Sharp Cutoff after lag 1 |
| **PACF Plot** | Sharp Cutoff | Slow Decay | Sharp Cutoff after lag 1 |

**Limitations of ARMA:**

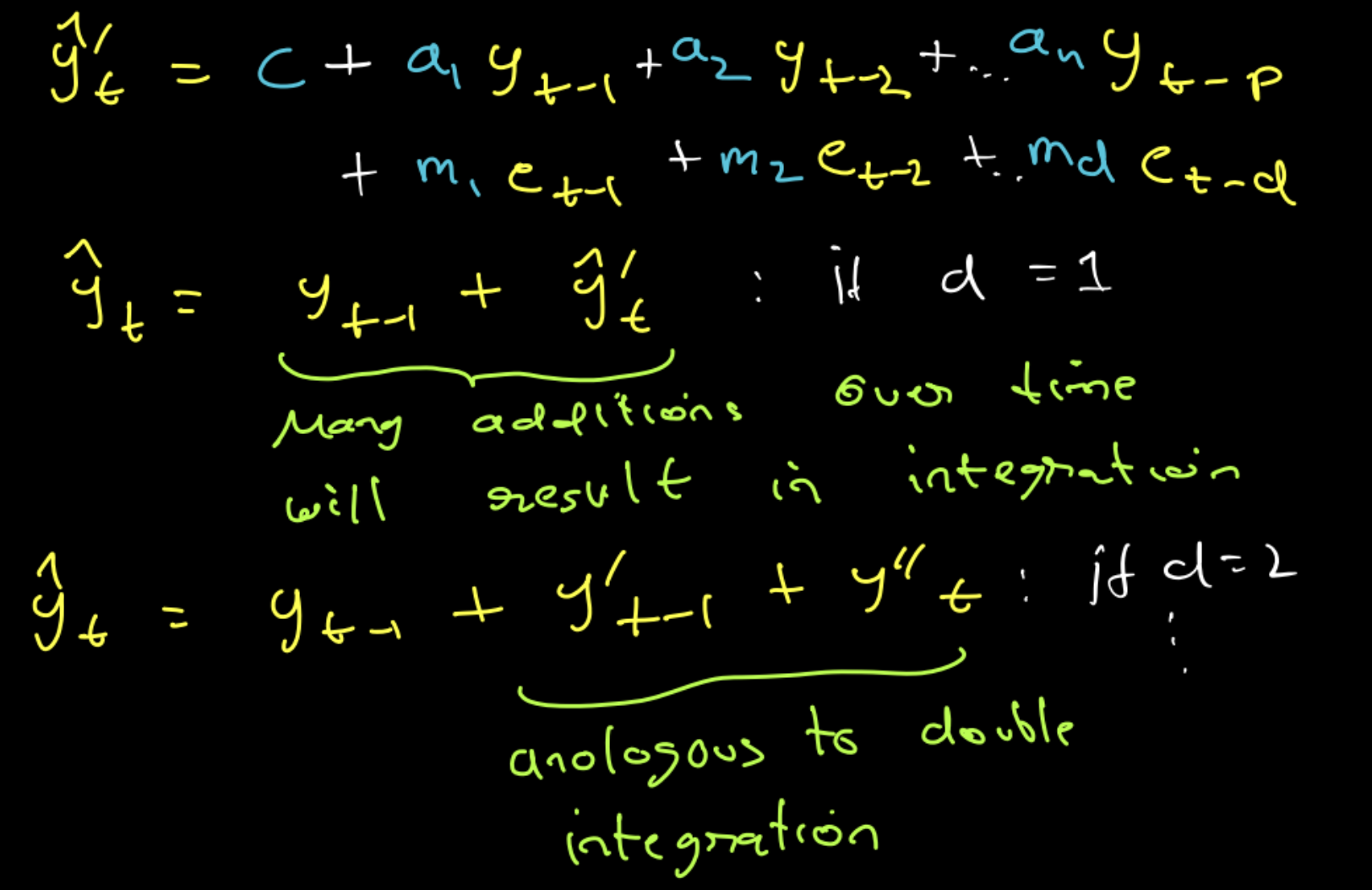
* Cannot handle non-stationary series and we cannot handle seasonal lags.
* Differencing causes the loss of one data point. If the order is 1 and in the case of the differencing the series the value will be NA.
* After the differencing the scale of the series will be changed. So, the forecast will also be on a different scale and we need to manually adjust the by retransforming by doing integration.

## **ARIMA**

* If a time series is not stationary,
  + We perform **differencing** to de-trend
  + Then we apply the ARMA technique, to get an approximation.
  + Now to get a good forecast, we need to **integrate** the trend **back** to get the final result
* This technique is called **ARIMA**
* Instead of us manually doing integrations (and note that it can get very hard when you need to double / tripple differentiate), we can simply use the ARIMA model, which does this job for us.
* ARIMA model is denoted as ARIMA(p,d,q)
  + Where,
    - p - order of the autoregressive part.
    - d - degree of first differencing involved.
    - q - order of the moving average part.

**Formulation of ARIMA**

* ARIMA can be formulated as a summation of:
  + Differencing of order d
    - d=1 for linear trends, d=2 for quadratic trends, ...
  + Auto Regression of order p
  + Moving Averages of order q
  + Getting the resulting forecasts
  + Integrating d times to restore the trend back



* We need to try some combinations of p and q parameters and compare results using a validation set.
* In order to find the best combination of p and q, we need to have some objective function that will measure model performance on a validation set.
* The high-level logic behind that is the same as the logic behind hyperparameter tuning of any other machine learning model. We need to try some combinations of p and q parameters and compare results using a validation set.
* Since our search space is not big, usually values p and q are not higher than 10, we can apply a popular technique for hyperparameter optimization called grid search.
* We can also use AIC and BIC for that purpose. The lower the value of these criteria, the better the model is.

# **SARIMA**

* If we wish to account for seasonality, the process becomes:-

1. differentiating (𝑥[ 𝑖 ] − 𝑥[ 𝑖−𝑇 ]), to remove seasonality
2. perform AR and MA
3. integrating the seasonality back

* This is very tiring.
* Instead of doing so much work to utilize the ARIMA model, we can just apply another model called the **SARIMA model** directly.
* There are 7 parameters for SARIMA: **P, Q, D, p, q, d, s**

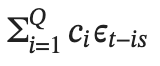
## **Hyperparameter s**

* The parameter s represents **seasonality**
  + This value we can find using **ACF and PACF plots**.
  + Alternately, we can treat s as a **hyperparameter** and tune it to get the best value.
* In case of lag of order 1, at time t, translates to 𝑦𝑡−1
* For example,
  + For a normal time series, a lag of order of 2 means 𝑦𝑡−1 and 𝑦𝑡−2
    - And this makes sense as if we're in March 2022, then order of 2 means, lag of 1, i.e. February 2022 and lag of 2, i.e. January 2022
  + In case of yearly seasonality, if we're in March 2022, a lag of order 2, means March 2021, and March 2020.
  + Therefore, here, an order of m gets translated as 𝑦𝑡−𝑠, 𝑦𝑡−2𝑠,..., 𝑦𝑡−𝑚𝑠.
* There is one **problem** with the SARIMA Model in terms of capturing seasonality.
* **We can only use one value of seasonality.**

## **Hyperparameter P**

* When we set the order of AR as p, the formulation becomes:



* **The effect of setting the `P` hyperparameter for SARIMA is: **
  + Since our seasonality is 12
  + Essentially, P enables in creating of an AR Model on data that is 12 months old, 24 months old, ..., 12P months old: 𝑦𝑡−12, 𝑦𝑡−24,...,𝑦𝑡−12𝑃.
  + So this is exactly like an AutoRegression, but with seasonality.
* Suppose we wish to forecast y^100 (i.e. t=100) with the following hyperparameter values:-
* 𝑝=4
  + Contribution to final forecast:

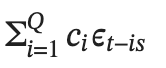


* s=12
* P=3
  + Contribution to final forecast:



## **Hyperparameter Q**

* Similarly, the effect of setting the Q hyperparameter for SARIMA is:



* Suppose if seasonality in a certain time series data is 12.
* Essentially, Q enables in creating of an MA Model on data that is 12 months old, 24 months old, ..., 12Q months old: 𝑦𝑡−12, 𝑦𝑡−24,..., 𝑦𝑡−12𝑄.
* So this is exactly like a Moving Average, but with seasonality.
* Suppose we wish to forecast 𝑦̂100 with the following hyperparameter values:-
  + q = 4
    - Contribution to final forecast:

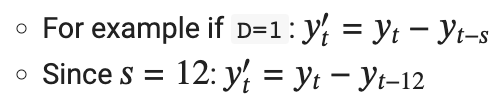


* + s = 12
  + Q = 3
    - Contribution to final forecast:



## **Hyperparameter D**

* Recall that hyperparameter d performs differencing on the time series d times, before applying the model.
  + For example if d=1: 𝑦′𝑡 = 𝑦𝑡 − 𝑦𝑡−1
* Similarly, D helps in doing **seasonal differencing** on the time series.



**So, to sum it up SARIMA can be formulated as:**

𝑦̂t+1 = p(AR term) + q(MA term) + d(differencing) + D(differencing) + P(AR-Seasonality) + Q(MA - Seasonality)

* The parameters p, q, d represent the non-seasonal part of the model
* P, Q, D, s represent the seasonal part of the model.
* The seasonal part of an AR or MA model will be seen in the seasonal lags of the PACF and ACF.
* For example, an ARIMA(0,0,0)(0,0,1)12 the model will show:
* a spike at lag 12 in the ACF but no other significant spikes;
* exponential decay in the seasonal lags of the PACF (i.e., at lags 12, 24, 36, …).
* Similarly, an ARIMA(0,0,0)(1,0,0)12 the model will show:
* exponential decay in the seasonal lags of the ACF.
* a single significant spike at lag 12 in the PACF.

# 

# 

# **SARIMAX Model**

* We can utilize the exogenous variable and incorporate it into our SARIMA Model.
* This idea of incorporating exogenous variables into SARIMA gave rise to a new model: **SARIMAX Model**
* Here, the X represents an exogenous variable
* The only difference between SARIMA and SARIMAX is that here, we can incorporate exogenous variables into the calculations of our forecasts.
* Exogenous variables are assigned a weight, say 𝑤𝑖.
* We don't need to initialize this, it is learned and trained by SARIMAX, and taken care of under the hood.
* This is done in addition to the SARIMAX operations.
* The SARIMAX model takes exogenous variables into account
  + i.e. variables measured at time 𝑡 that influence the value of our time series at time 𝑡, but that are not autoregressive.
* To do this, we simply add the terms on the right-hand side of our ARIMA and SARIMA equations.

