Temporal Analysis (Coursera) F0813T08XX

노트북:	DeepLearning		
만든 날짜:	2018-08-13 오후 1:03	**수정한 날짜:**	2018-08-15 오후 5:32
작성자:	Choi, Hyunjun		
URL:	file:///C:\Users\CHJ\Desktop\iREMS\99%20기타\Coursera_Temporal_Analysis_HW_01\DTW_MATRIX.xlsx		

Temporal Data

- 2D XY Time Series data는 Cyclical Pattern을 찾기 어렵다

  이런 경우, spiral graph를 활용하여 패턴 인식
  Spiral Graph의 주요 산식
  - Archimedian Spiral
   r = a + b * theta
  - Logarithmic Spiral
   r = a*exp(b*theta)
   theta = (1/b) * ln(r/a)

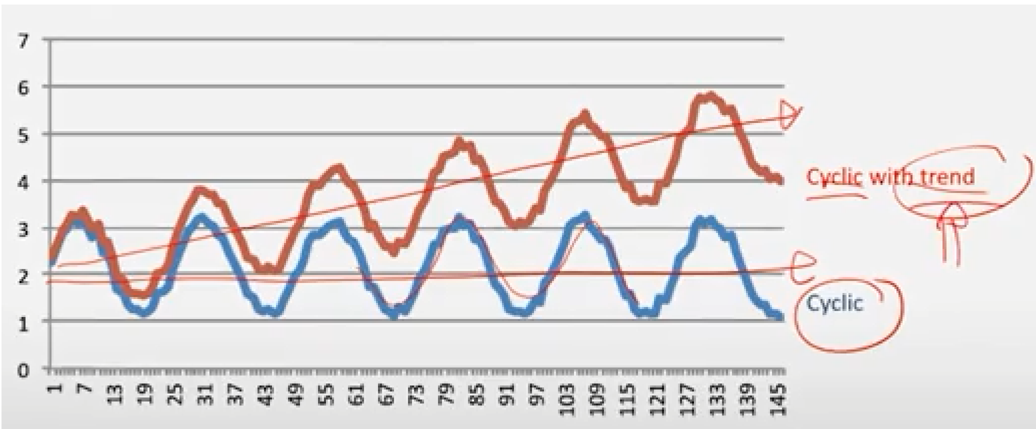
- Spiral Graph is method to encode data over time.
- Trend Analysis > Seasonality and Forecasting
- Control Chart Analysis
  - Objective: Detect Patterns and Anomalies in big data
  - Exploring patterns and anomalies in time series data is a Control Chart.
  - In temporal data, We can find patterns through contol chart.
  - Mean and Standard deviation from all data
  - Upper Control Limit (mu + k*sigma) / Center Line (mu) / Lower Control Limit (mu - k*sigma)
  - Moving Average Chart >> monitors the process location over time (detect small shifts in the process mean, control limits are deriven from average range on Range Chart)
  - Range Chart >> monitors the process variation over time, should be reviewed before Moving Average Chart
    - exmaple
      - << Standard Moving Average >>
      - Daily Closing Prices: 11, 12, 13, 14, 15, 16, 17
      - First day of 5-day SMA: (11+12+13+14+15)/5 = 13
      - Second day of 5-day SMA: (12+13+14+15+16)/5 = 14
      - Third day of 5-day SMA: (13+14+15+16+17)/5 = 15
      -
      - << Exponentially Weighted Moving Average >>
      - if SMA = 10 period sum / 10,
      - Muliplier: (2 / (Time periods+1)) = (2 / (10+1)) = 0.1818 (18.18%)
      - EMA: {Close - EMA(previous day)} x multiplier + EMA (precious day)

- SMA with longer window is good for tracking slow moving historical trends and changes
- EMA can capture quick upticks in thoses things
- Shorter Moving (nimble and quick to change)
- Longer lag (Longer the moving average, more the lag)
- Longer Moving (Longer moving slow to change)

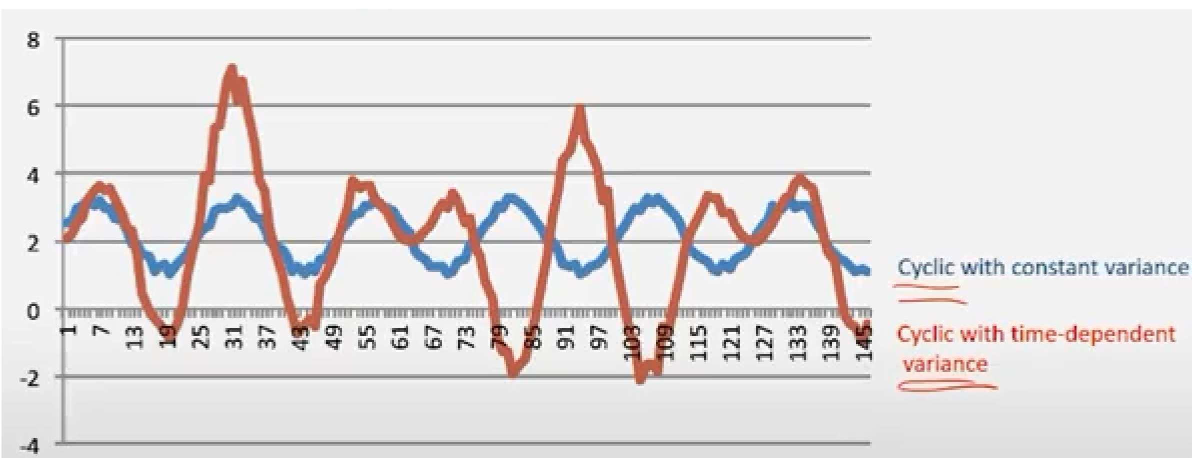
Time Series

- With Time Series Data, We can compare pattern among the different Time Series Data
- With Time Series Data, We can forecast future Data
- With Time Series Data, We can find motif(Repeating Pattern)
- With Time Series Data, We can classify Time Series Data   (Because time series usually Bayesian is used to record real-world events)

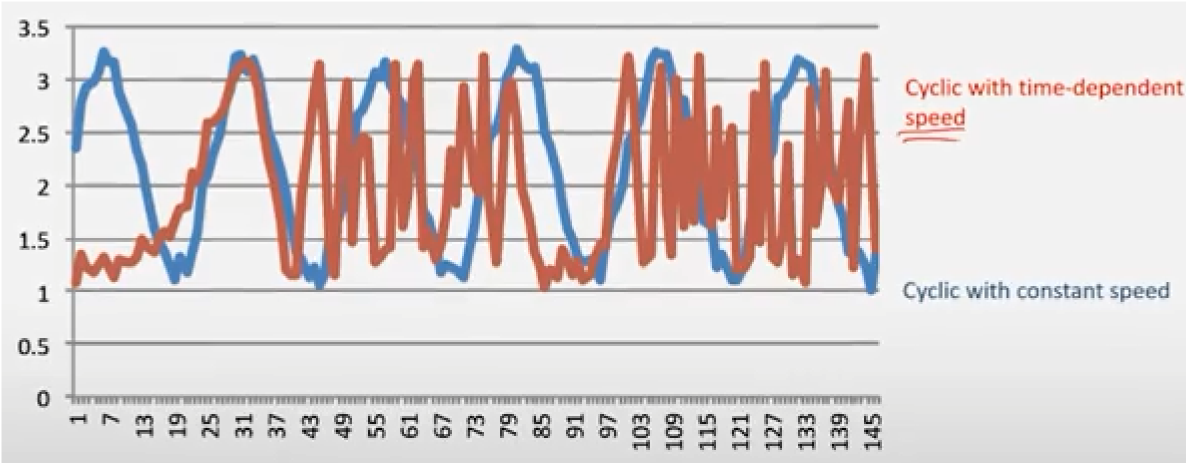
- Time Series Models (High-level Properties of Time Series : type of time series)
- >> Stationary Series
- >>> **Statistical properties**, such as mean and variance, ***are constant over time***
- >> Non-Stationary Series
- >>> **Statistical properties *change over time***
- >>>> 1. Cyclic (periodic behavior)
- >>>> 2. Cyclic with trend



- >>>> 3. Cyclic with constant variance
- >>>> 4. Cyclic with time-dependent variance



- >>>> 5. Cyclic with constant speed
- >>>> 6. Cyclic with time-dependent speed



- !! Why important?
    - most statistical forecasing methods assume that the series can be rendered(approximately) stationary through mathematical transformation.

- **<< Time Series Modeling >>**   Models of Time Series

- Random Time
- Autoregressive Time
- Moving Average
- ARMA
- Differencing
- ARIMA

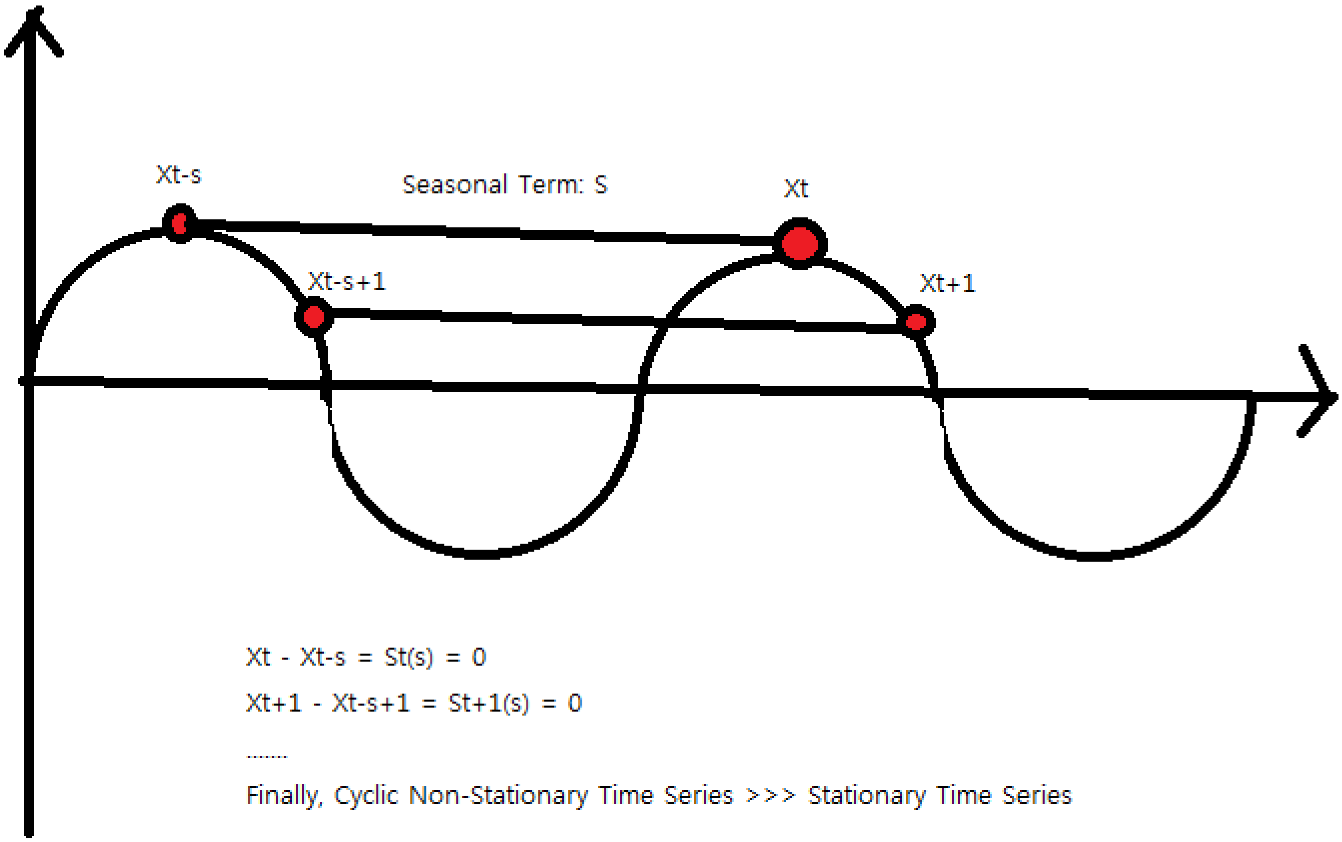
- Random Time Series Model
    - Current observations only reflect current(random) error:
        - Xt = Et (Et ~ N(0, sigma^2) -> Generally use Gaussian. Zero mean with some variance   <----- Et is Error at that time t (or External Effect at that time t)
    - Intuitively, the "random error" represents a stochastic process that does not depend on the past (e.g. an external input to the system)
    - Generally, like as the case that Stock market easily depend on past values many real-world phenomena are not Random Time Series Model.. (But, I think the case of Nature Environment like weather data sometimes regard as random time series model.. for example, When we construct constraint system, the weather trend is random time series models)
- Autoregressive Time Series Model
    - AR(1): the **current observation** depends only the **previous time instance** and the **current error**
        - Xt = alpha*Xt-1 + Et + lambda
    - AR(2): the currnet observation depends only the previous 2 time instances and the current error
        - Xt = alpha1 * Xt-1 + alpha2 * Xt-2 + Et + lambda
    - AR(n): the current observation depends only the prevoius n times instances and the current error
        - Xt = SIGMA(i from 1 to n) [alphi * Xt-i] + Et + lambda
- Moving Average Time Series Model
    - MA(1): the current observation depends only to the error in the previous time instance and the current error
        - Xt = beta * Et-1 + Et + lambda
    - MA(2): the current observation depends only to the error in the previous 2 time instances and the current error

            - Xt = beta1 * Et-1 + beta2 * Et-2 + Et + lambda
- **ARMA Model**
    - AR + MA (a,m):
    - AR(a): The model has a auto-regressive terms
    - MA(m): The model has m moving-average terms
        - Xt = (alpha1 * Xt-1 + .. + alphaa * Xt-a) + (beta1 * Et-1 + .. + beta * Et-m) + Et + lambda
        - >>> We usually don't use large a and m!! because large a and m make model too difficult to discover accurately
    - Shortcoming of ARMA models:
        - These models can't models where the current value is determined by taking into account the speed of change or degree of acceleration observed in the past
        - The time is constant or assume that the speed of the events are constant
        - **don't take account speed of change or the degree of acceleration of the time series**

- **Models with Differencing**
    - **"differencing"** enables models to also **consider speed of change and degree of acceleration in determining the current value:**
        - Order-1 differencing:
            - Xt(1) = Xt - Xt-1 (speed of change)
        - Order-2 differencing:
            - Xt(2) = Xt(1) - Xt-1(1) (degree of acceleration)
        - ...
        - Order-d differencing:
            - Xt(d) = Xt(d-1)-Xt-1(d-1)
- **ARIMA(a, d, m)**
    - AR(a): The model has a auto-regressive terms
    - MA(m): The model has m moving-average terms
    - Intergrated with order-d differencing
        - Xt = (alpha1 * Xt-1 + .. + alphaa * Xt-a) + (beta1 * Et-1 + .. + beta * Et-m) + Et + lambda + (theta1 * Xt-1(d) + ... + thetam * Xt-m(d)

[[[ARIMA just takes care about immediate past values and errors!! So, It is hard to recognize cyclic pattern by using ARIMA]]]

- **<< Time Series Seasonal Differencing >>**
- Models with Seasonal Differencing
- **Seasonal "differencing"** enables models to also **consider *<u>speed of changes of values across a gap</u>*** (or a lag):
    - lag "s' seasonal differencing
    - St(s) = Xt - Xt-s
    - We can convert Non-stationary Time Series to stationary Time Series when original time series have cyclic property.



- Seosonal ARIMA models also incorporate seasonal terms
  - additive seasonal models
    - Xt = ARIMA(a,d,m) + SEASONALs(A,D,M)
      A = # seasonal autoregressive tems, D = # seasonal differences, M = # seasonal moving average terms
  - Multiplicative seasonal models
    - Xt = ARIMA(a,d,m) X SEASONALs(A,D,M)
      A = # seasonal autoregressive tems, D = # seasonal differences, M = # seasonal moving average terms

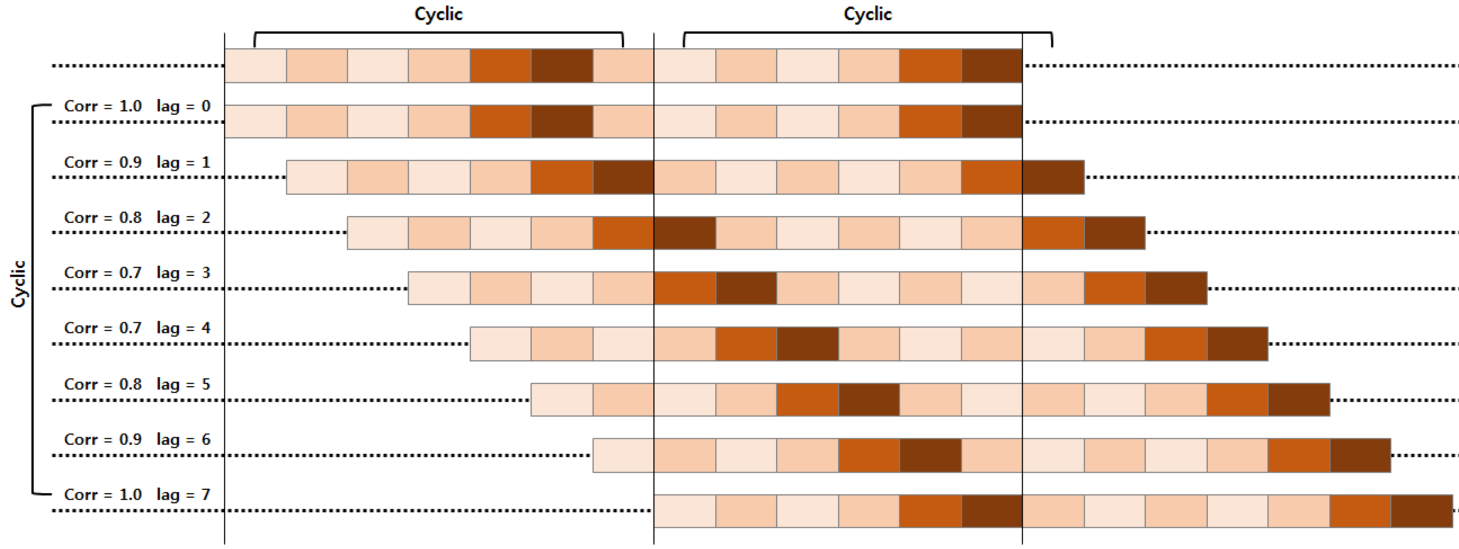
SO~~ What we want to do is Time Series Data Discovery! >> closed-form formula.

Model Discovery
(Box-Jenkins Procedure)

- Model based Time Series Analysis
  - Find the parameters of the model
    - Model fitting **(Degree of fit and Complexity Trade off Relationship)**
      - ther model should be as simple as possible (contain as few terms as possible)
      - the fit to historic data should be as good as possible
- **Box-Jenkins procedure**
  - **Differencing (for obtain as possible as stationary time series data)**
    - **Remove any seasonal patteerns and deterministic trends** that may hide valuable information and patterns through differencing **(for obtain stationary time series form)**
    - When the **mean trend is stochastic**, **differencing the series may yield a stationary stochastic process** and, thus, may help convert a non-stationary series to a stationary one
  - **Plot Analysis**

- **Autocorrelation Function(ACF) helps *observe linear relationships* between *lagged values* of a time series**

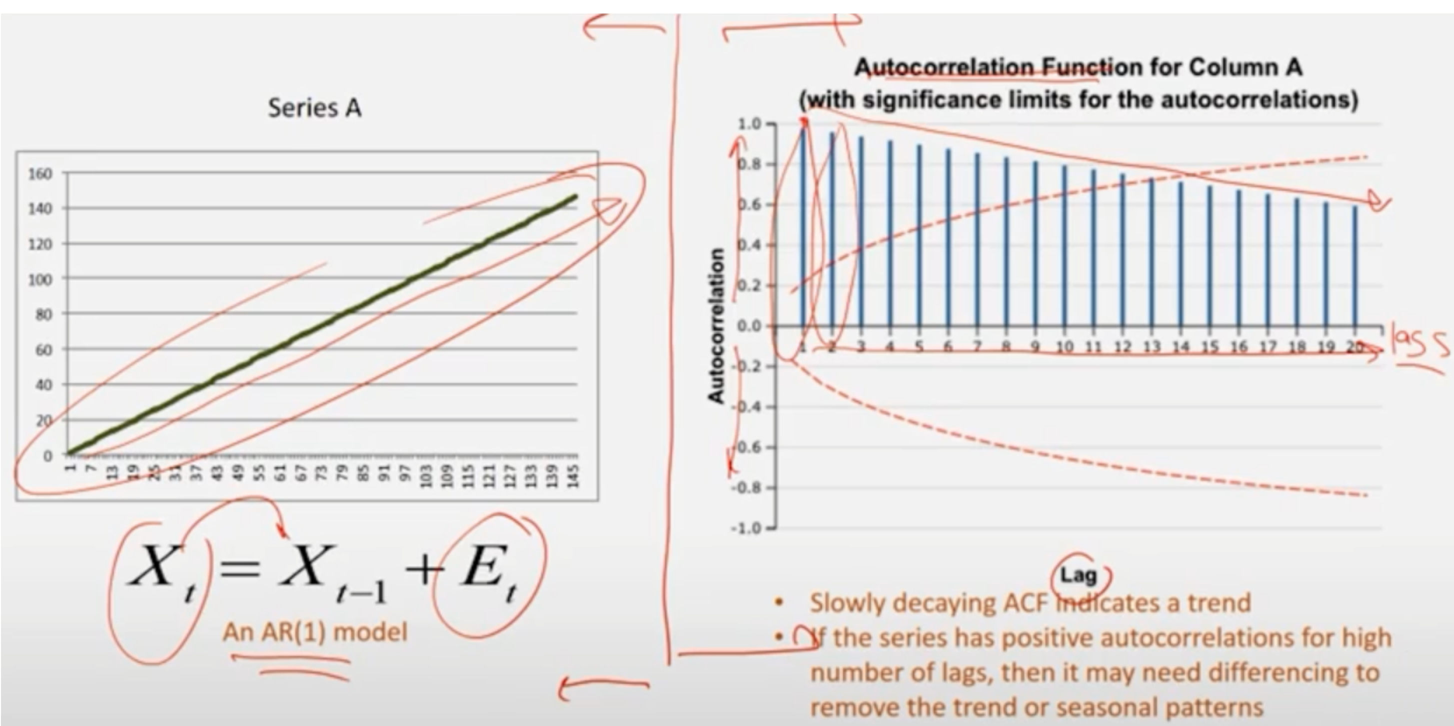
$$\mathrm{ACF}(X, \mathrm{lag}) = \frac{E[(X_t - \mu)(X_{t+lag} - \mu)]}{\sigma^2} = \frac{Covar(X_t, X_{t+lag})}{\sigma^2}$$



- **Partial Autocorrelation Function(PACF) adjusts for the presence of intermediate values**

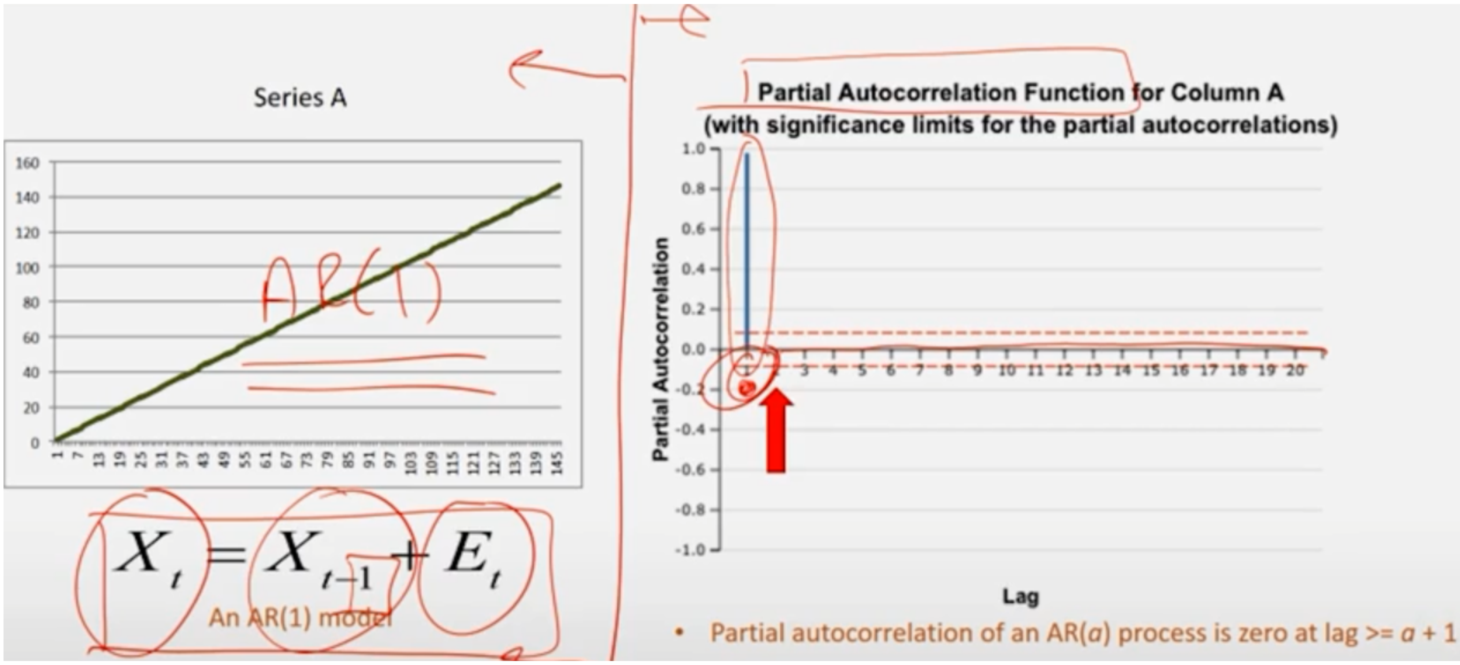
$$\mathrm{PACF}(X, \mathrm{lag}) = \frac{Covar(X_t, X_{t+lag} | X_{t+1}, \dots, X_{t+lag-1})}{\sigma^2}$$

- **Key Properties of ACF and PACF**
- For trend series autocorrelation function (ACF) slowly decays.

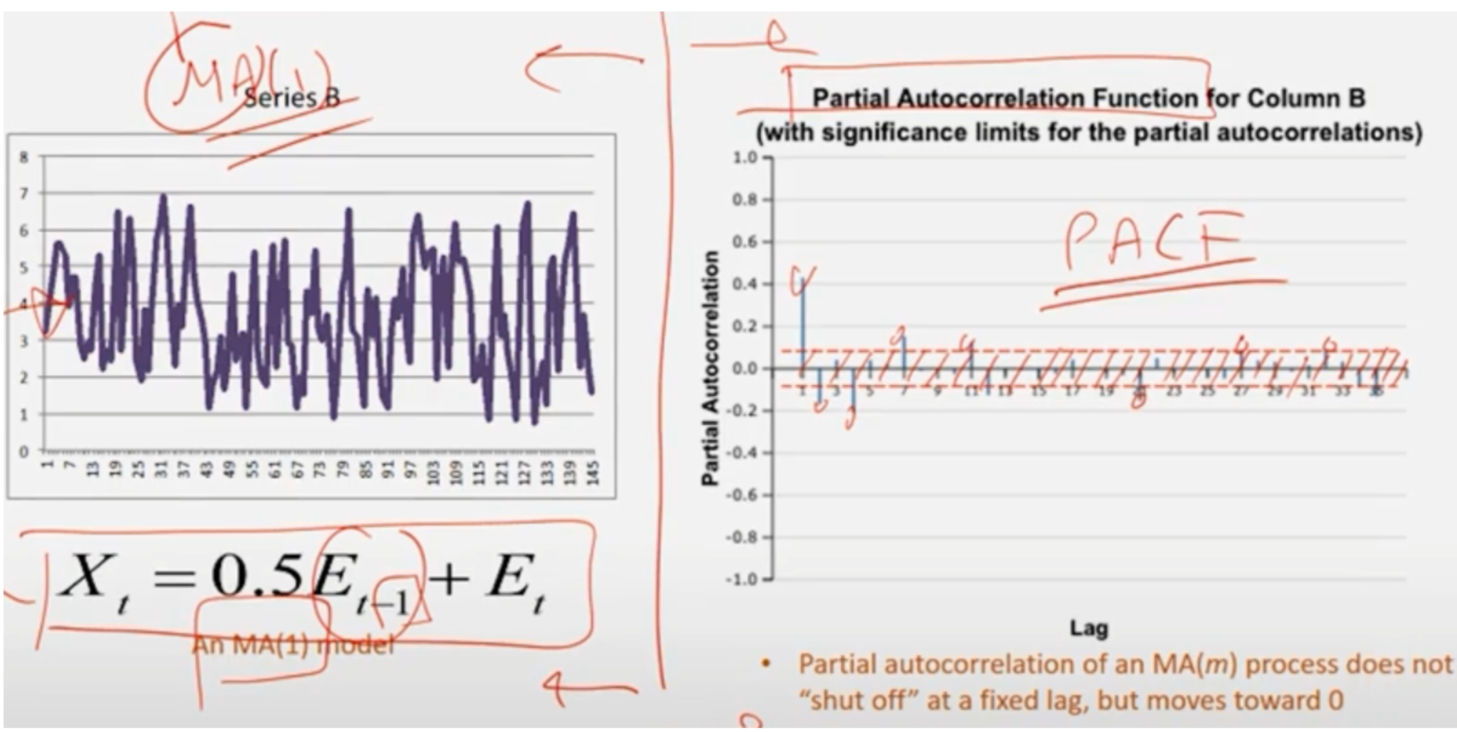


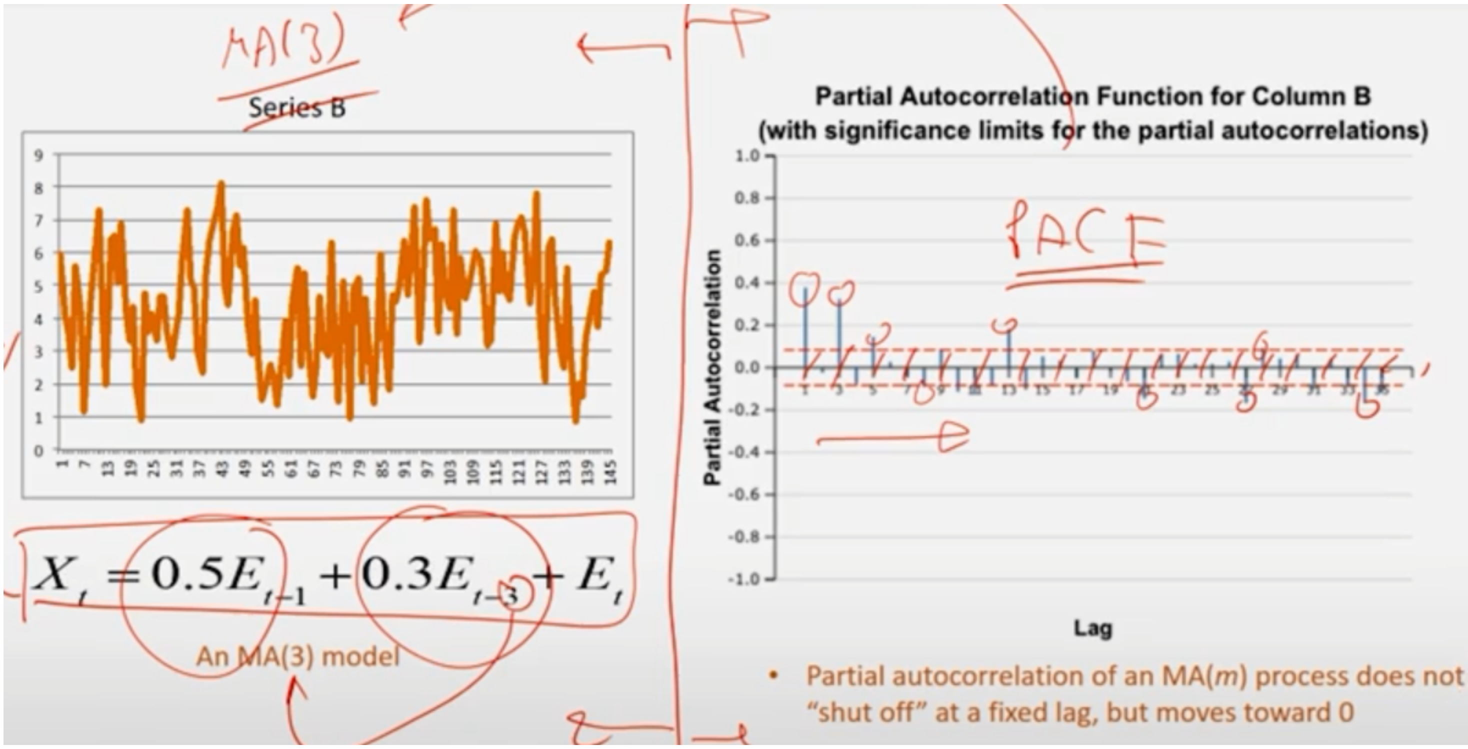
- Upper Right Graph's X-Axis is lag and Y-Axis is Autocorrelation Value

- **For an autoregressive, AR(a), series, the partial autocorrelation function (PACF) gives 0 at lag >= a+1**
  - **below righ-hand side graph indicate the time series is AR(1) !!**



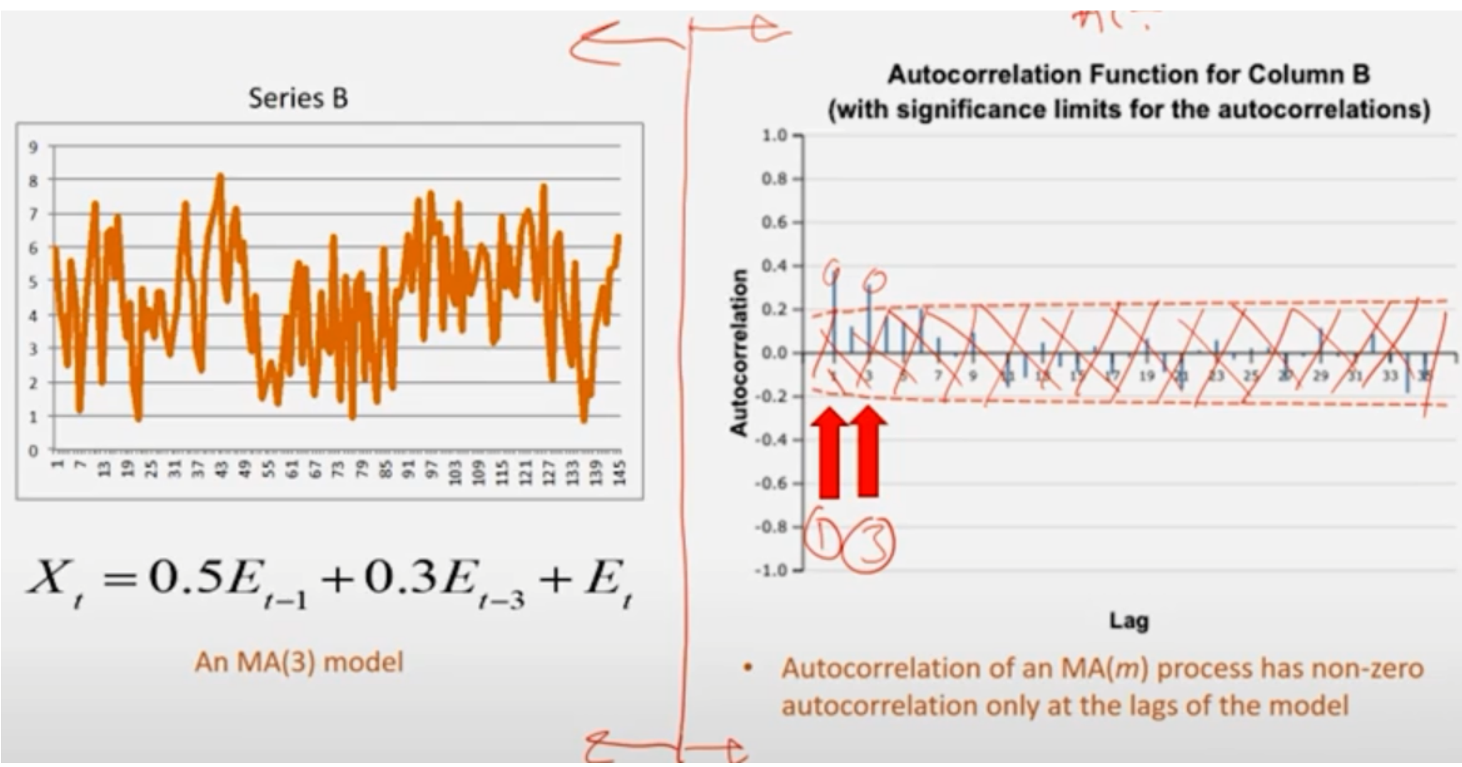
- For a **moving average, MA(m)**, series,
  - **PACF doesn't "shut off" at a fixed lag, but moves toward 0.**





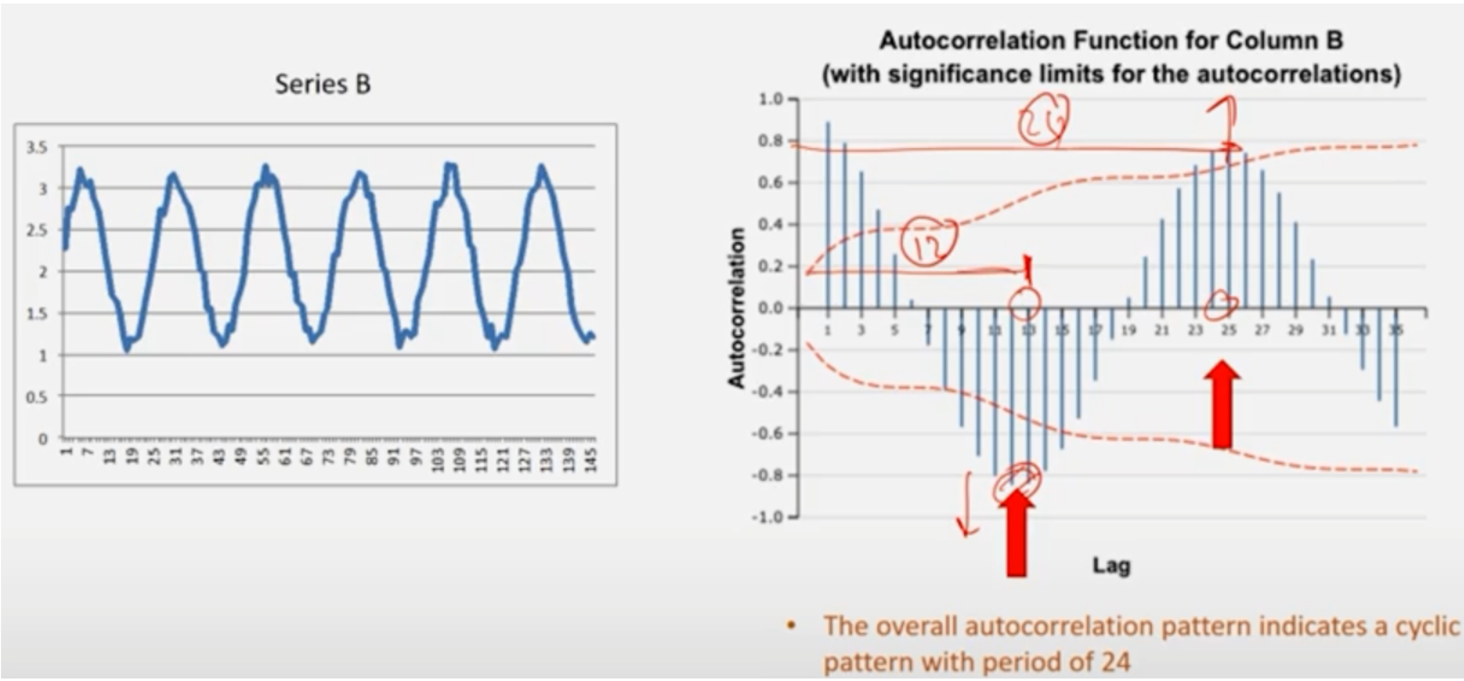
- **>> upper graphs show that PACF (Not Shutted off plots) tell us "your time series is moving average !!**

- **AND ACF's non-zero autocorrelation value only at lag means the time series is moving average !! (Below Graph)**

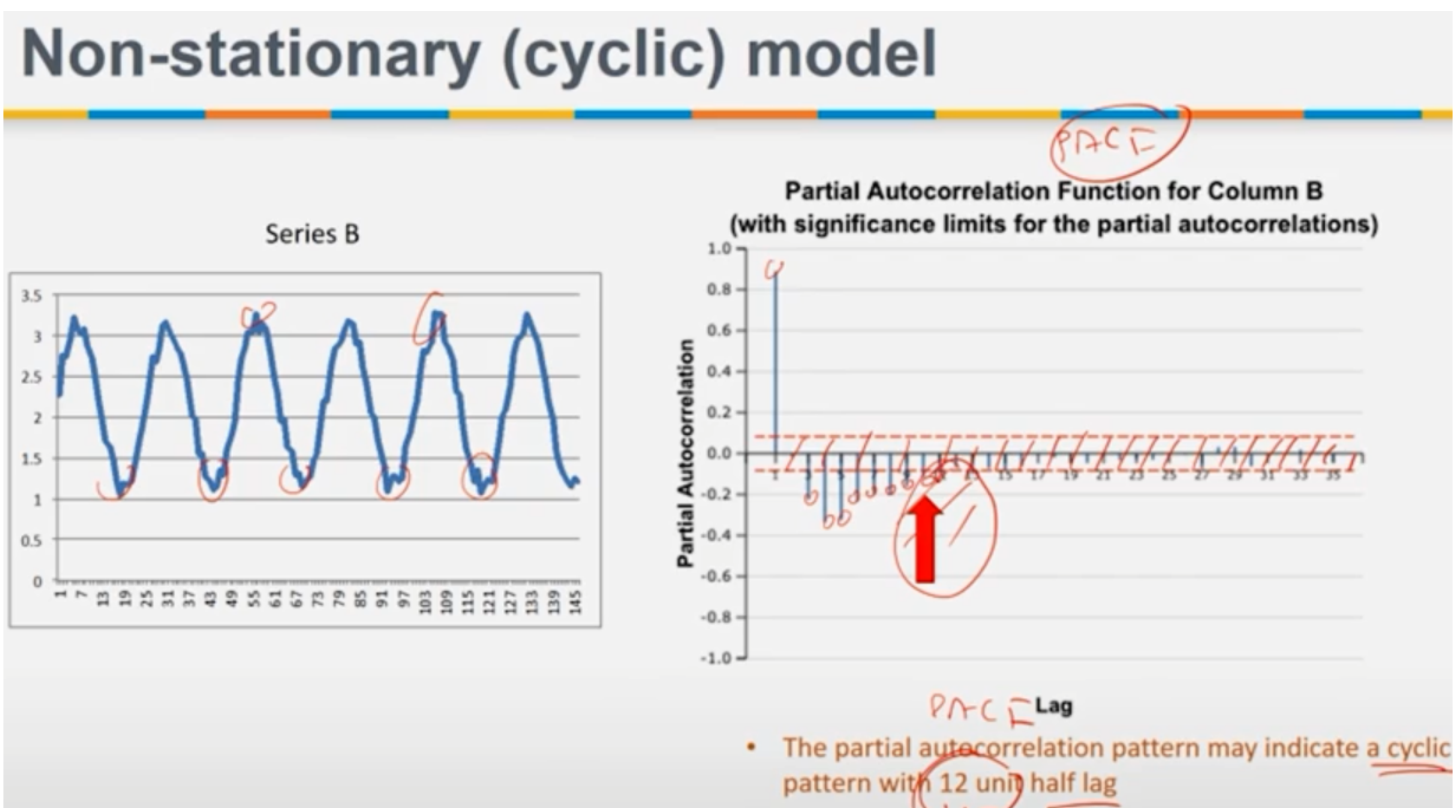


- **<< Seasonal Series Using ACF and PACF 5 >>**

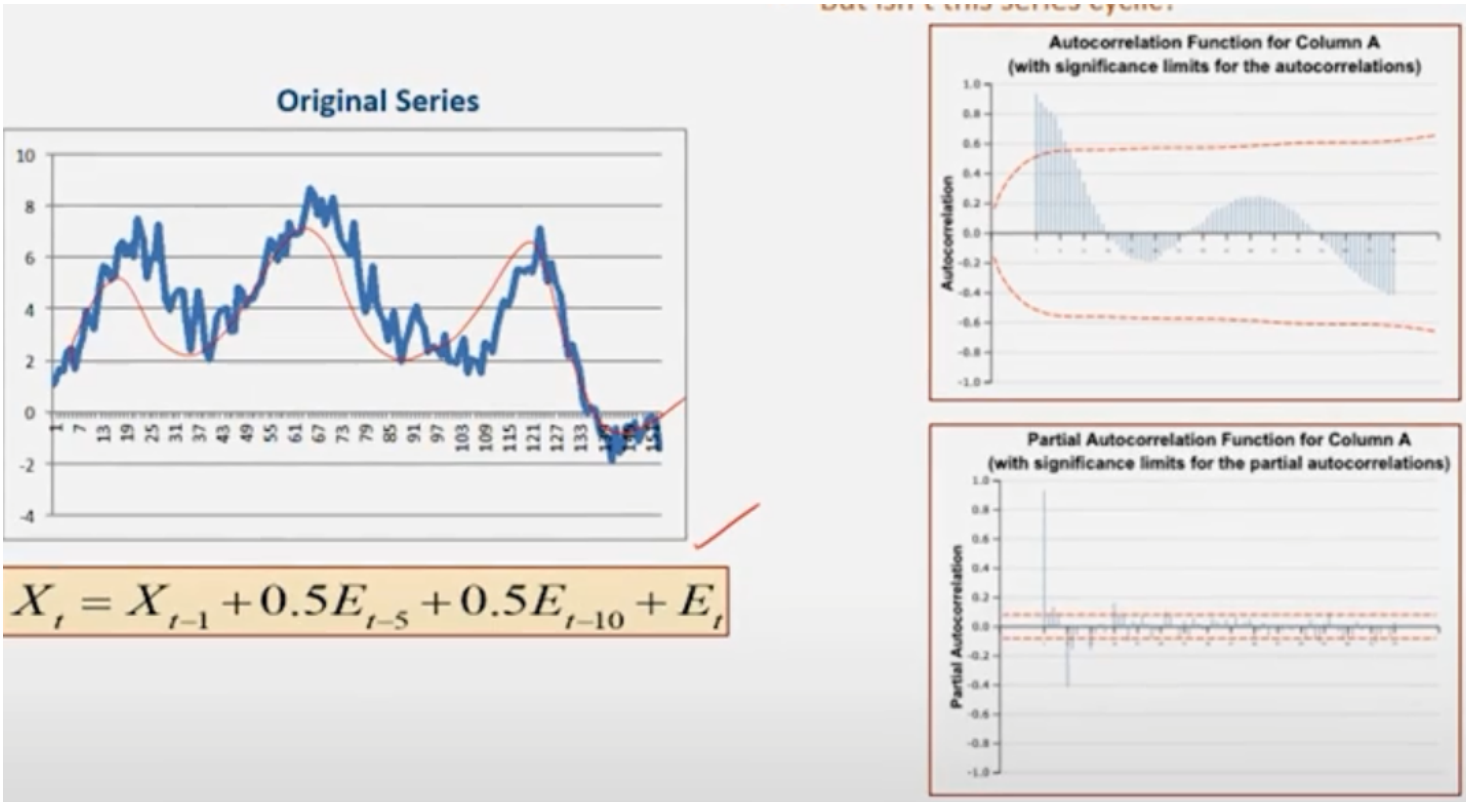
- **For seasonal** series autocorrelation function**(ACF) at the seasonal lag will be large and positive !!**
    - Below by using ACF



- **Below by using PACF**



- **Complex model (Real World)**
- The differenced series appears to be a monving aegra
  - D =Xt - X-1 = alpha * E(t-5) + beta*Et-5 + beta*Et-10 + Et
  - D =Xt - X-1 = alpha * E(t-5) + beta*Et-10 + Et
  - Xt = Xt+1 + 0.5Et-5 + 0.Et-0) G



- **<< Time Series Matching >>**

- **< Quantify similarity or distance btw time series >**

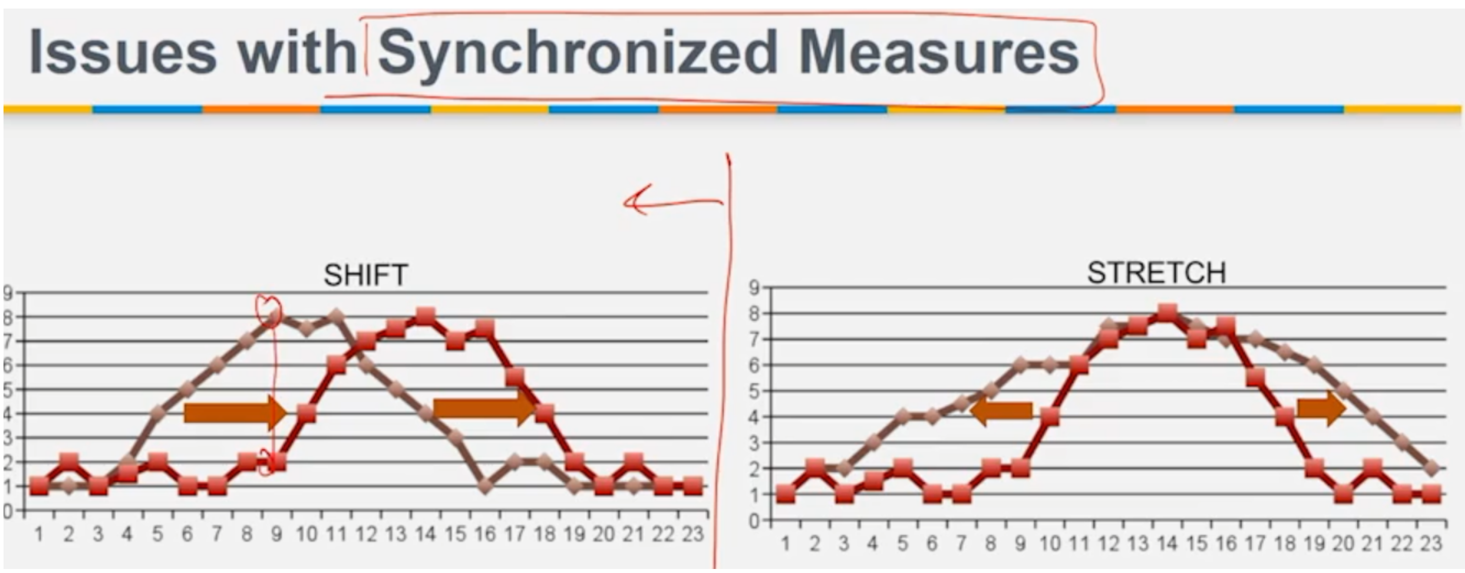
- **Euclidean Distance**

$$\Delta_{Euc}(B, M) = \sqrt{\sum_{i=1 \dots N} (B_i - M_i)^2}$$

- **Correlation Similarity**

$$\mathrm{Sim}_{correl}(B, M) = \frac{E[(B - \mu_B)(M - \mu_M)]}{\sigma_B \sigma_M}$$

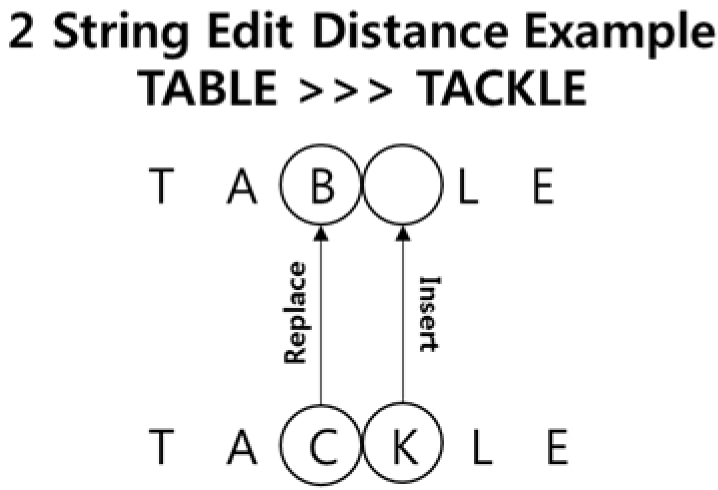
- **Synchronized Measures (time synchronized)**



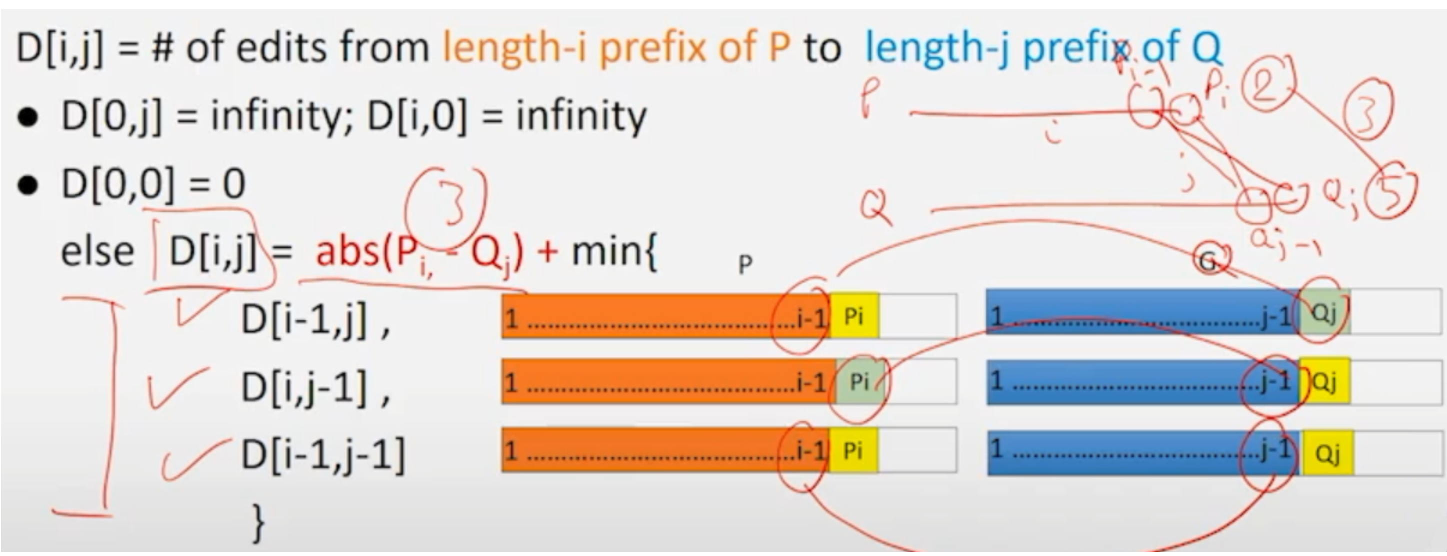
- **Left (Time Shift) | Right (Time Stretch)**
- ***** It can not measure similarity in case of time shift!! So we need something else method**

- **< Edit Distance and Dynamic Time Warping >**
- **Edit cost**
  - Let **E** be a sequence of edit operations to convert one **string** to another
  - Let us associate a cost, C, to each edit operation
    - Costs of edit operatinos can be different from each other
      - Type of the operation (replace, delete, insert)
      - Symbols involved in the operation
      - Position of the edit operation
  - Given a sequence of edit operations, E

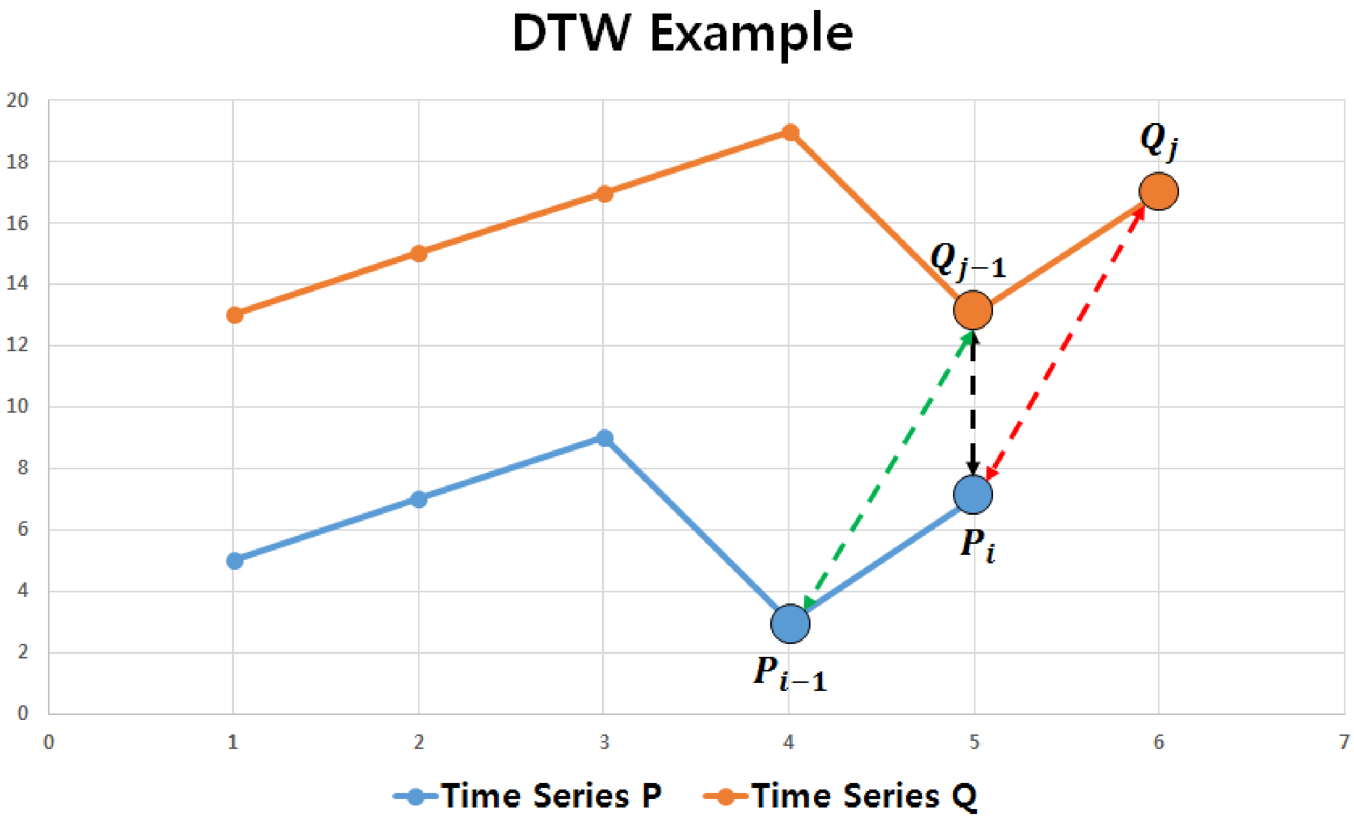
$$C(E) = \sum_{e_i \in E} C(e_i)$$



- **Dynamic Time Warping**
  - **Given two time series P and Q, of lengths N and M**



- abs(Pi - Qj) is replacement operation



Distance btw P_i and Q_j is abs($P_i - Q_j$) + min

- **Below Image is simple example of construction of DTW Matrix**

Dynamic Time Warping (DTW)

Complexity : O(M,N)

Dynamic Programming based implementation (Recursive Computation using memorable table)

N	PN							
...	...							
...	...							
3	P3							
2	P2							
1	P1							
	O	Q1	Q2	Q3	...		...	QM
		1	2	3	...		...	M

M = 9	**Q**	**7**	**9**	**2**	**9**	**2**	**7**	**6**	**5**	**4**
N = 7	**P**		**9**	**5**	**2**	**4**	**5**	**5**	**8**	

P7	8	1	1	6	1	6	1	2	3	4
P6	5	2	4	3	4	3	2	1	0	1
P5	5	2	4	3	4	3	2	1	0	1
P4	4	3	5	2	5	2	3	2	1	0
P3	2	5	7	0	7	0	5	4	3	2
P2	5	2	4	3	4	3	2	1	0	1
P1	9	2	0	7	0	7	2	3	4	5
	O	7	9	2	9	2	7	6	5	4
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9

P7	8	17	17	19	14	20	15	16	16	**17**
P6	5	16	18	13	14	14	15	14	13	14
P5	5	14	16	10	11	13	13	13	13	14
P4	4	12	14	7	10	11	12	14	15	15
P3	2	9	11	5	12	9	14	18	18	17
P2	5	4	6	5	9	12	14	15	15	16
P1	9	2	2	9	9	16	18	21	25	30
	O	7	9	2	9	2	7	6	5	4
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9

- Pink box is real value of P and Q Time series
- Sky blue box is cost of Pi and Qj
- Green box is accumulative distance by axis respectively
- Ligth Yellow box is Minimum distance among the alternative replacement
  (in case of [P2,Q2] = MIN (SUM ([P2,Q1], COST[P2,Q2]) , SUM ([P1,Q1], COST[P2,Q2]), SUM ([P1,Q2], COST[P2,Q2]))
- Pure Yellow box is fianal cost between P and Q time series.

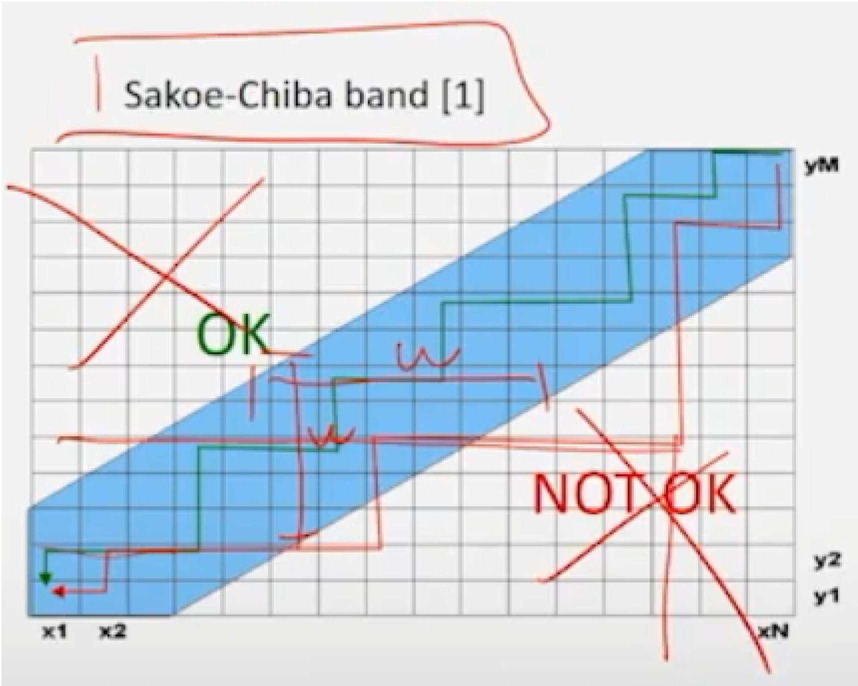
- **Below Image is mapping path between P and Q Time Series (Case 1)**

P7	8	17	17	19	14	20	15	16	16	**17**
P6	5	16	18	13	14	14	15	14	**13**	14
P5	5	14	16	10	11	13	13	**13**	13	14
P4	4	12	14	7	10	11	**12**	14	15	15
P3	2	9	11	5	12	**9**	14	18	18	17
P2	5	4	6	**5**	**9**	12	14	15	15	16
P1	9	**2**	**2**	9	9	16	18	21	25	30
	O	7	9	2	9	2	7	6	5	4
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9

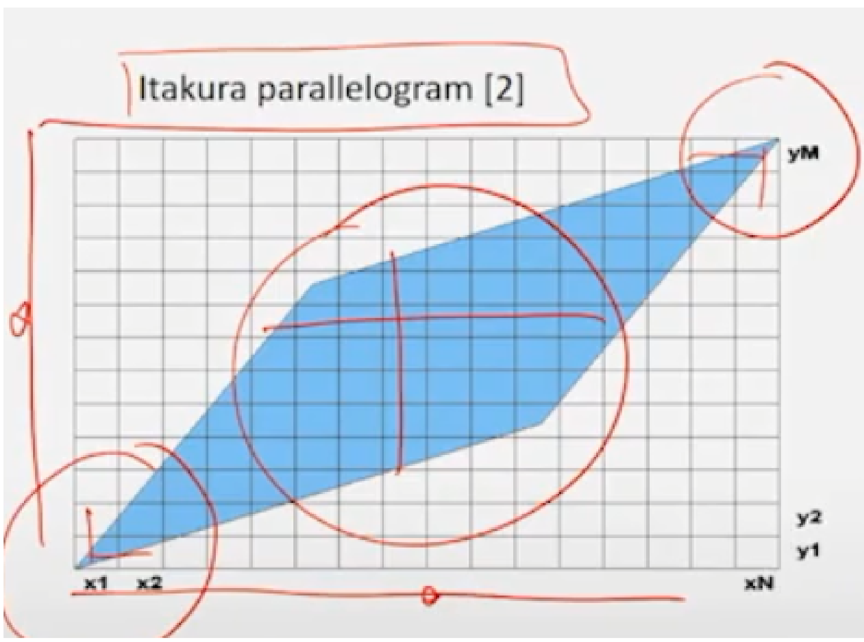
- **Below Image is mapping path between P and Q Time Series (Case 2)**

P7	8	17	17	19	14	20	15	16	16	**17**
P6	5	16	18	13	14	14	15	14	**13**	14
P5	5	14	16	10	11	13	13	**13**	**13**	14
P4	4	12	14	7	10	11	**12**	14	15	15
P3	2	9	11	5	12	**9**	14	18	18	17
P2	5	4	6	**5**	**9**	12	14	15	15	16
P1	9	**2**	**2**	9	9	16	18	21	25	30
	O	7	9	2	9	2	7	6	5	4
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9

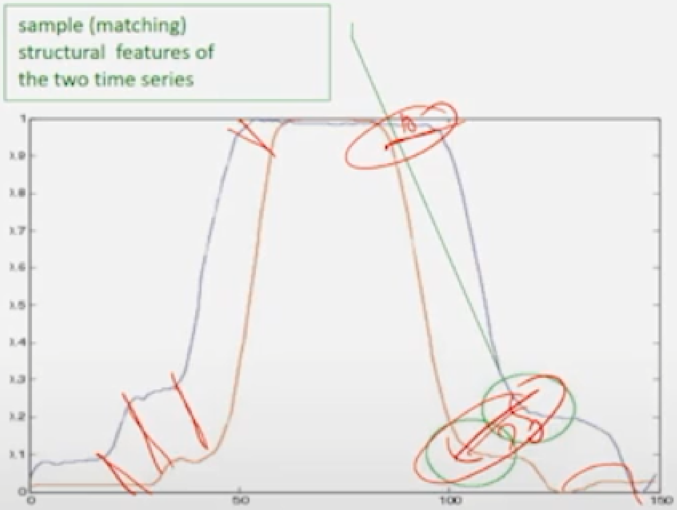
- But, DTW Computing is expensive casue of O(M,N)
- So, for solving this problem we can constraint some conditions.
- Generally, time shift is localized. not on whole time period..
- So, we gonna restrict that
- **There are Sakoe-Chiba band and Itakura Parallelogram constraints.**
- **Sakoe-Chiba band (FIXED WINDOW SIZE to ALL TIME PATH)**



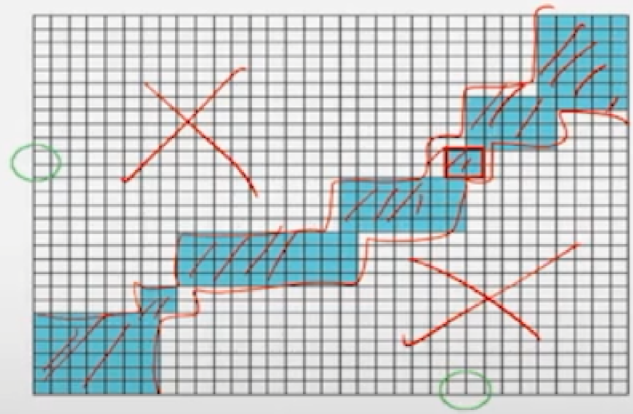
- **Itakura Parallelogram (MIDDLE of TIME PATH have a LARGER WINDOW SIZE than start / end time)**



- **sDTW is adaptive window size decision**



Two time series



Adaptive constraints on the DTW grid