POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Seasonal-Trend Decomposition in Time Series

POSTECH
Department of Creative IT Engineering
Doyup Lee

Index

- What is Time Series ?
- Stationarity vs. Non-stationarity
- Basic Approaches of Time Series Modeling
- **RobustSTL** (AAAI 2019 paper)
- Q & A

Caution

- This material with English.
- Many typings and mathematics.
- Need Background Knowledge about Time Series and Optimization.

Caution

- This material with English.
- Many typings and mathematics.
- Need Background Knowledge about Time Series and Optimization.
- But if you concentrate on the presentation and follow me,
  you can understand and know I am a liar.

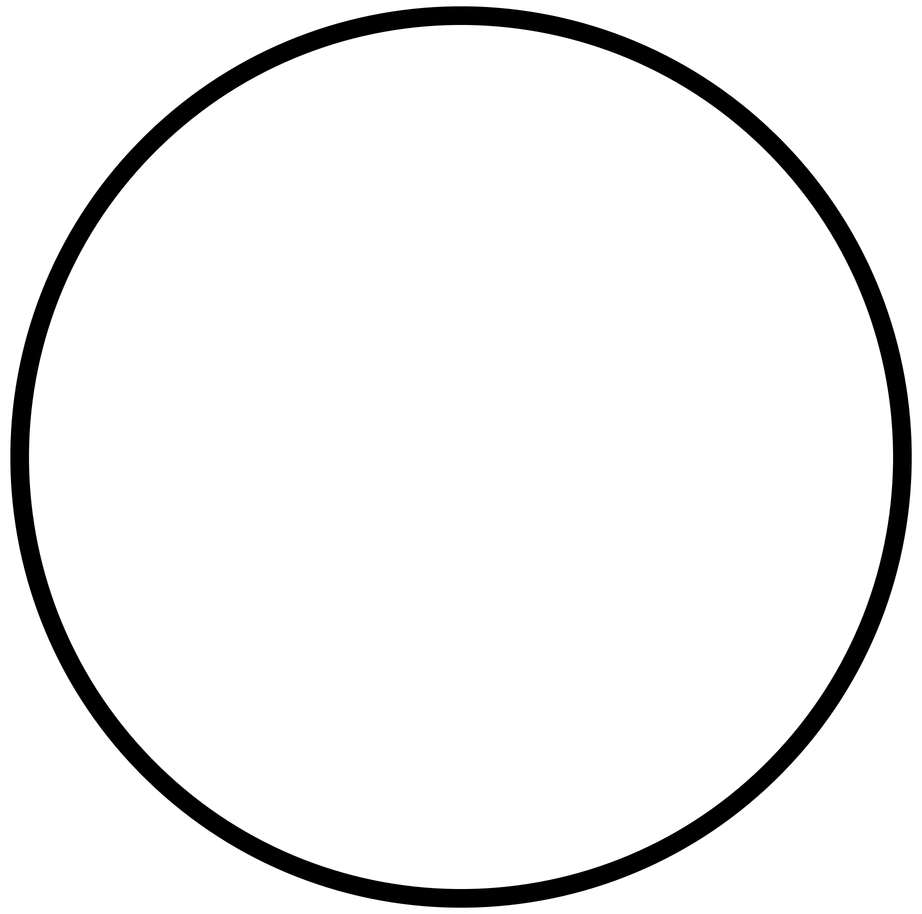
What Machine Learning Do ?

- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability $P(X)$ or $P(X, Y)$
  - Discriminative model learns to estimate conditional probability $P(Y|X)$

What Machine Learning Do ?

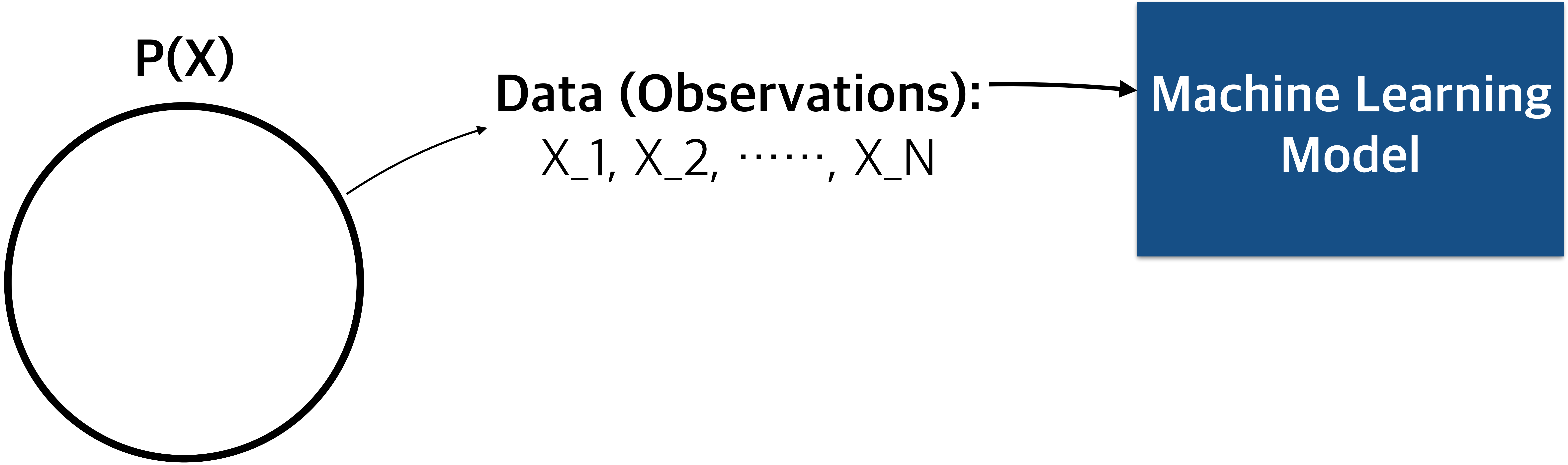
- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability P(X) or P(X, Y)
  - Discriminative model learns to estimate conditional probability P(Y|X)

P(X)



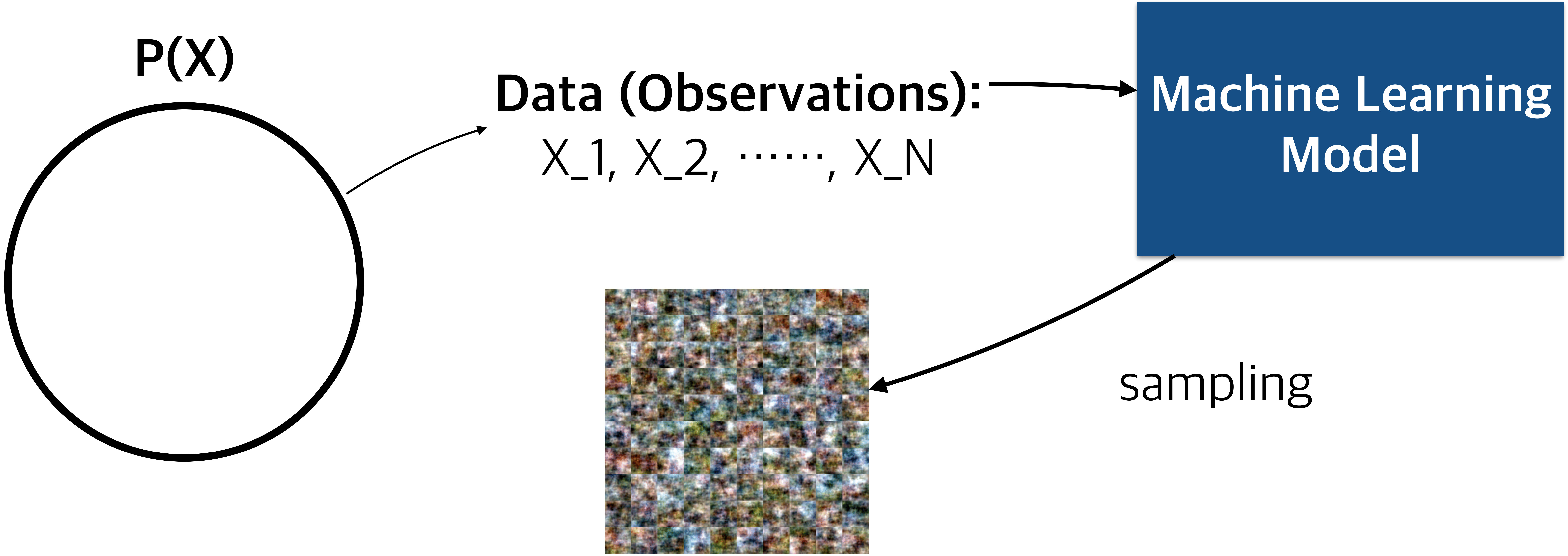
What Machine Learning Do ?

- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability P(X) or P(X, Y)
  - Discriminative model learns to estimate conditional probability P(Y|X)



What Machine Learning Do ?

- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability P(X) or P(X, Y)
  - Discriminative model learns to estimate conditional probability P(Y|X)



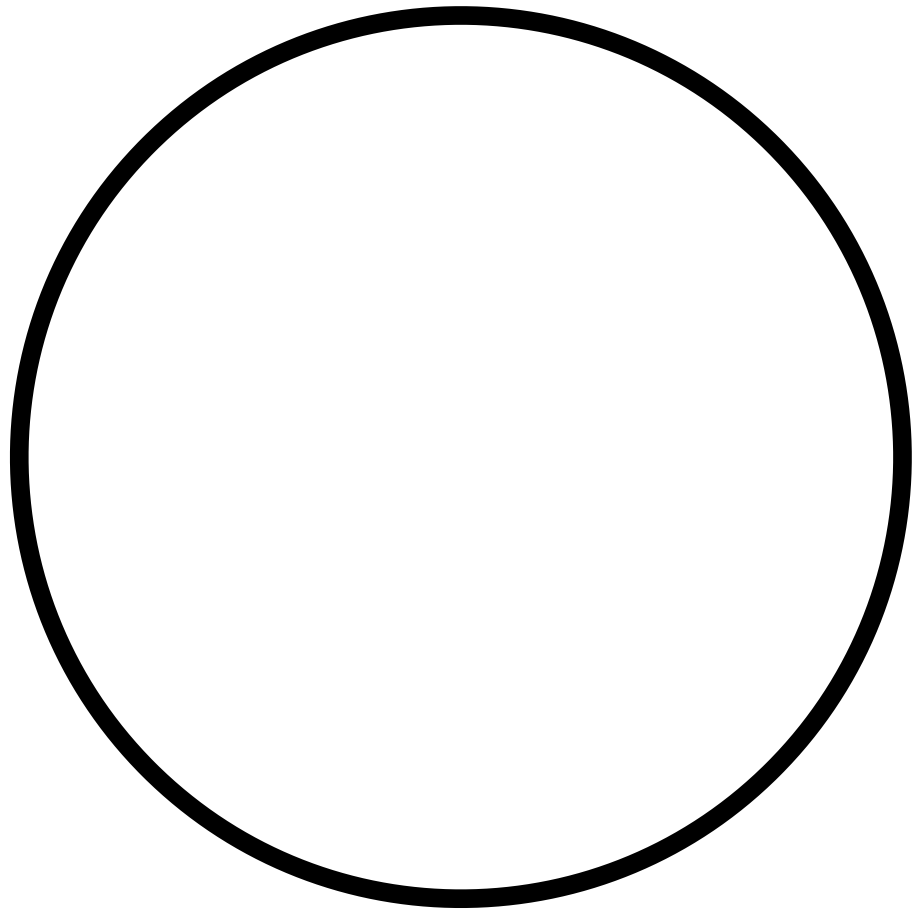
What Machine Learning Do ?

- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability $P(X)$ or $P(X, Y)$
  - Discriminative model learns to estimate conditional probability $P(Y|X)$

What Machine Learning Do ?

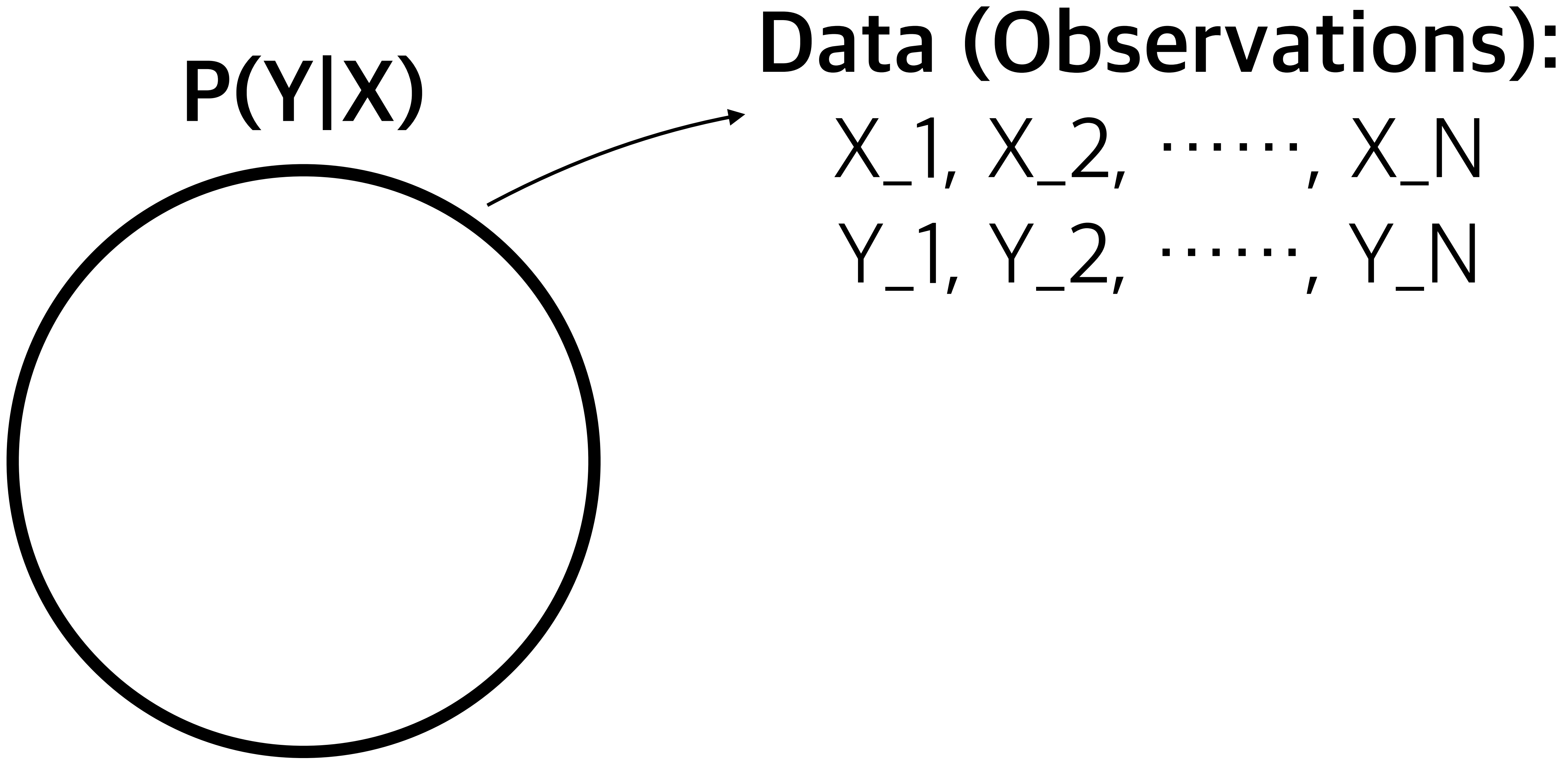
- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability P(X) or P(X, Y)
  - Discriminative model learns to estimate conditional probability P(Y|X)

P(Y|X)



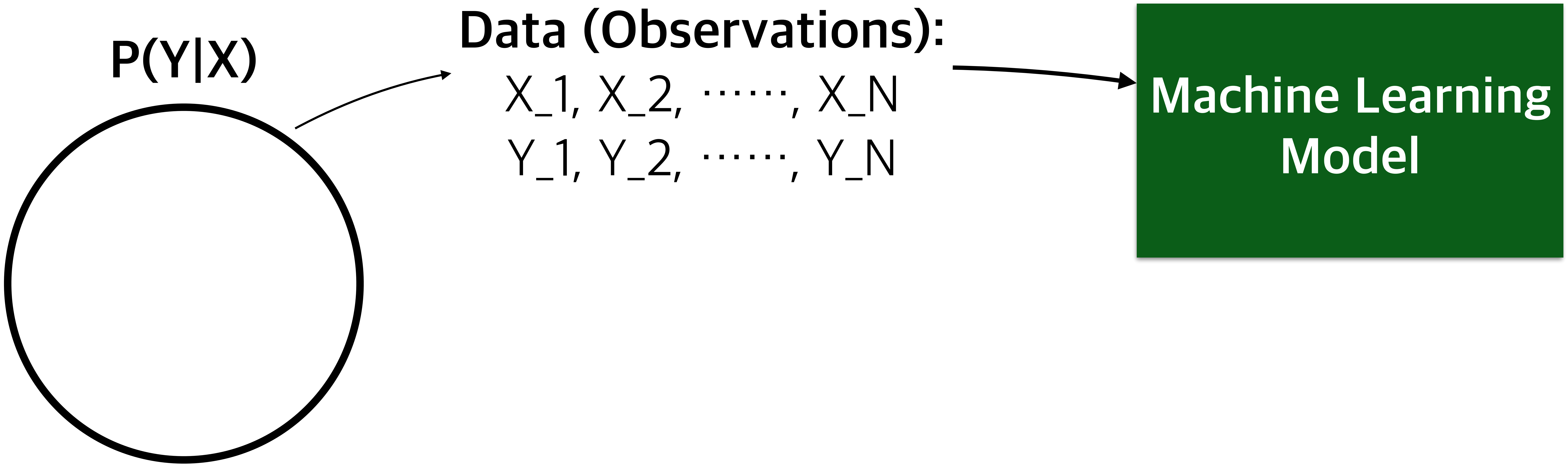
What Machine Learning Do ?

- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability P(X) or P(X, Y)
  - Discriminative model learns to estimate conditional probability P(Y|X)



What Machine Learning Do ?

- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability P(X) or P(X, Y)
  - Discriminative model learns to estimate conditional probability P(Y|X)



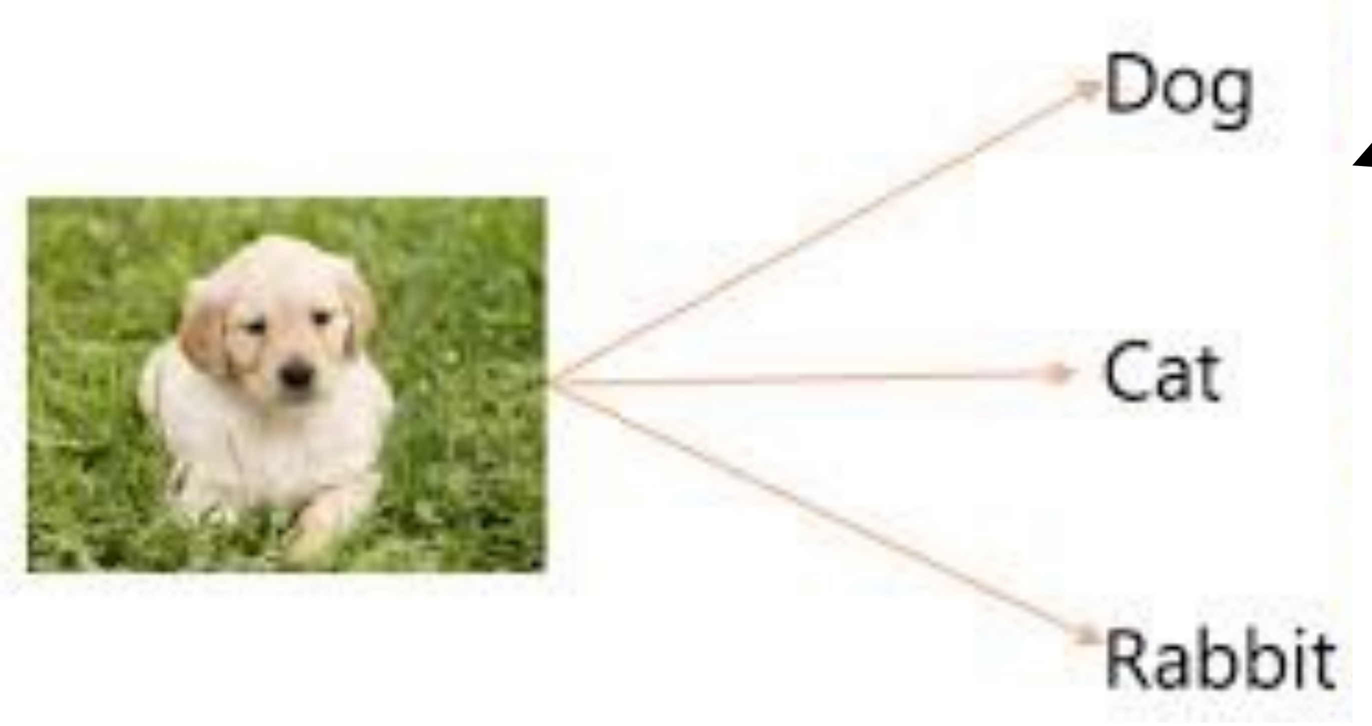
What Machine Learning Do ?

- **In general, machine learning estimates probability distribution.**
  - Generative model learns to estimate joint probability P(X) or P(X, Y)
  - Discriminative model learns to estimate conditional probability P(Y|X)

P(Y|X)

Data (Observations):
X_1, X_2, ……, X_N
Y_1, Y_2, ……, Y_N

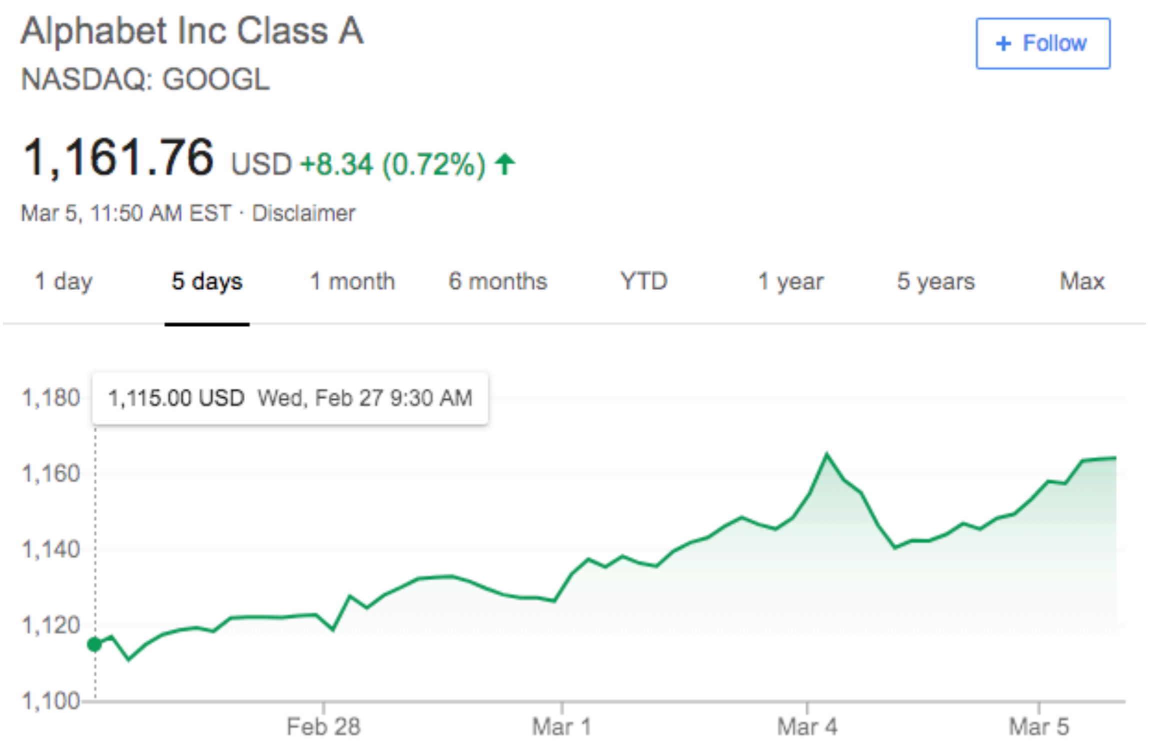
Machine Learning Model

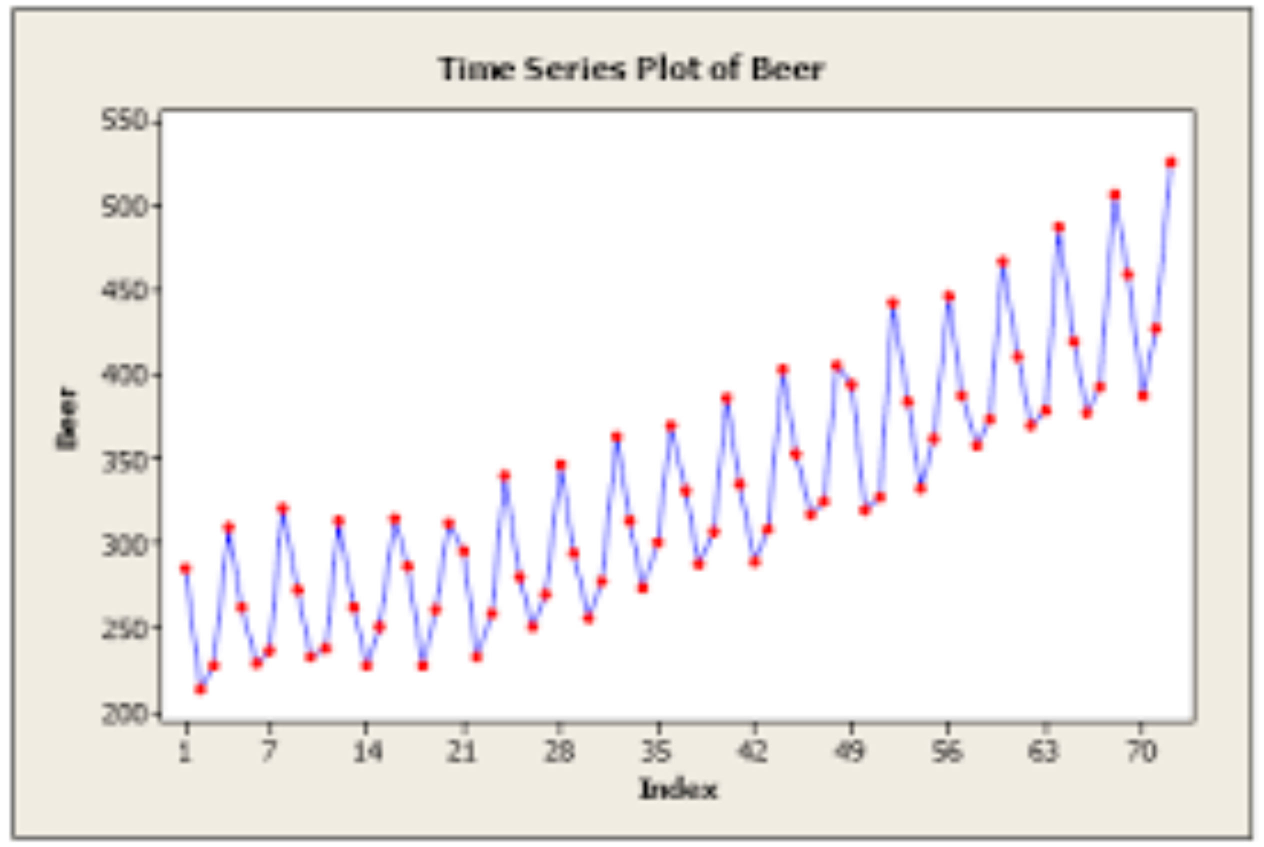


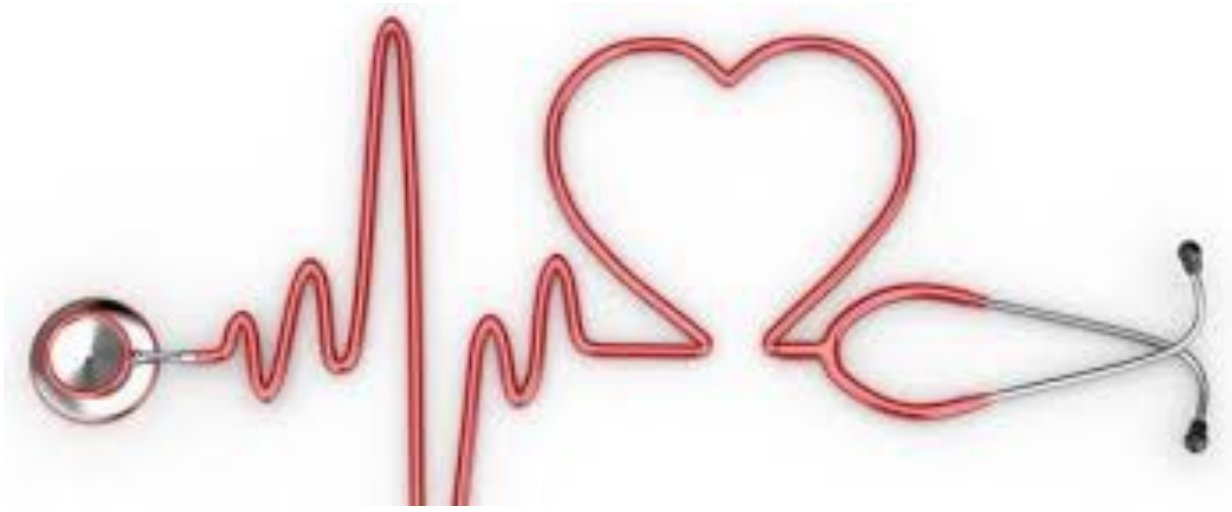
Given test data X

Time Series Data is all around us

- Services: User Log
- Finance: Many types of Price
- Manufactoring: Smart Factories
- Health Care: EEG, ECG, …
- Meteorological Data
- Etc.…







Because

- **NO TIME NO LIFE** : Our life is defined by “TIME”

- Everything in our life is connected with time changes.
  - Does a state change or not ?
  - How it changes ?
  - Which characteristics in the state according to time change?

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Time Series

Time Series

- **Time Series:**
  An ordered sequence of values of a variable at equally spaced time intervals.

Time Series

- **Time Series:**
  An ordered sequence of values of a variable at equally spaced time intervals.

- **Time Series Modeling:**
  Model a stochastic process with **autoregressive** manners.

Time Series

- **Time Series:**
  An ordered sequence of values of a variable at equally spaced time intervals.

- **Time Series Modeling:**
  Model a stochastic process with **autoregressive** manners.

- In the end, time series modeling can be to **find probability distribution** of a variable at time t, conditioned on past time t-1, t-2, …… .

Time Series Modeling Intuition

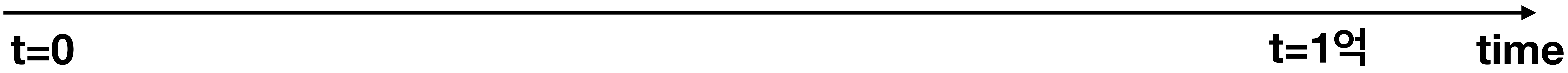
Time Series Modeling Intuition

Time Series Modeling Intuition

Time Series Modeling Intuition

Time Series Modeling Intuition

Time Series Modeling Intuition



$$sample_i = \{Y_i, Y_{i+1}, \ldots, Y_{i+T}\}$$

Time Series Modeling Intuition

t=0 t=1억 time

T+1

sample 1

sample 2

.
.
.

sample N

$$sample_i = \{Y_i, Y_{i+1}, \ldots, Y_{i+T}\}$$

observation
(realization)

Distribution of
Ordered (T+1) Series

$$P(Y_i, Y_{i+1}, \ldots, Y_{i+T})$$

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Time Series Modeling Intuition

t=0 t=1억 time

T+1

sample 1

sample 2

.
.
.

sample N

Distribution of
Ordered (T+1) Series

$$P(Y_i, Y_{i+1}, \ldots, Y_{i+T})$$

Time Series Modeling Intuition

t=0 t=1억 time

T+1

sample 1

sample 2

.
.
.

sample N

God

Distribution of
Ordered (T+1) Series

$$P(Y_i, Y_{i+1}, \ldots, Y_{i+T})$$

Time Series Modeling Intuition

t=0 t=1억 time

T+1

sample 1

sample 2

.
.
.

sample N

Distribution of
Ordered (T+1) Series

$$P(Y_i, Y_{i+1}, \ldots, Y_{i+T})$$

Time Series Modeling Intuition

t=0 t=1억 time

T+1

sample 1

sample 2

.
.
.

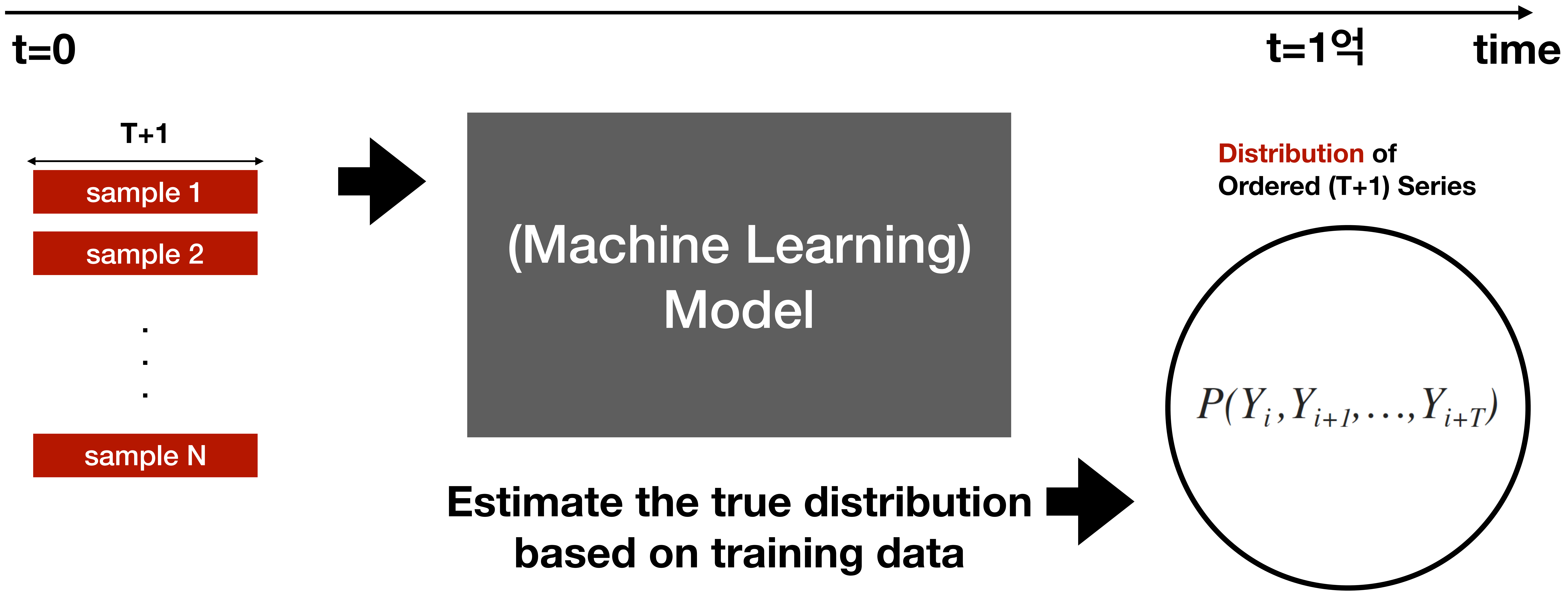
sample N

(Machine Learning) Model

Distribution of Ordered (T+1) Series

$$P(Y_i, Y_{i+1}, \ldots, Y_{i+T})$$

Time Series Modeling Intuition



ARMA (Whittle 1951; Box&Jenkins, 1971)

- **ARMA** (**A**uto**R**egressive **M**oving **A**verage) is a typical model for time series.
- ARMA(p,q): a **generative** linear model that combines **AR(p)** and **MA(q)**

$$\forall t, Y_t = \underbrace{\sum_{i=1}^{p} a_i Y_{t-i}}_{\text{AR(p)}} + \epsilon_t + \underbrace{\sum_{j=1}^{q} b_j \epsilon_{t-j}}_{\text{MA(q)}}.$$

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

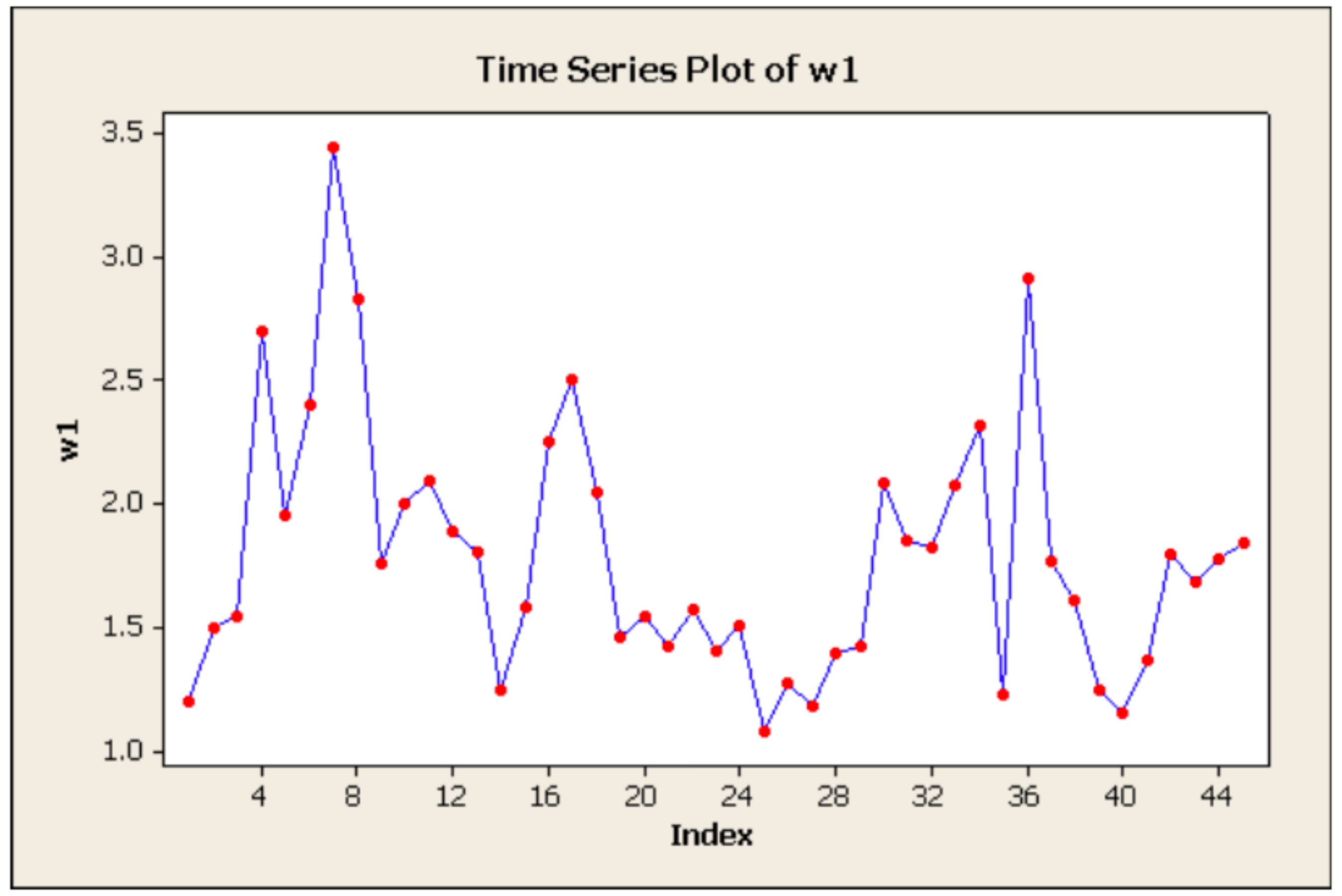
Modeling Process

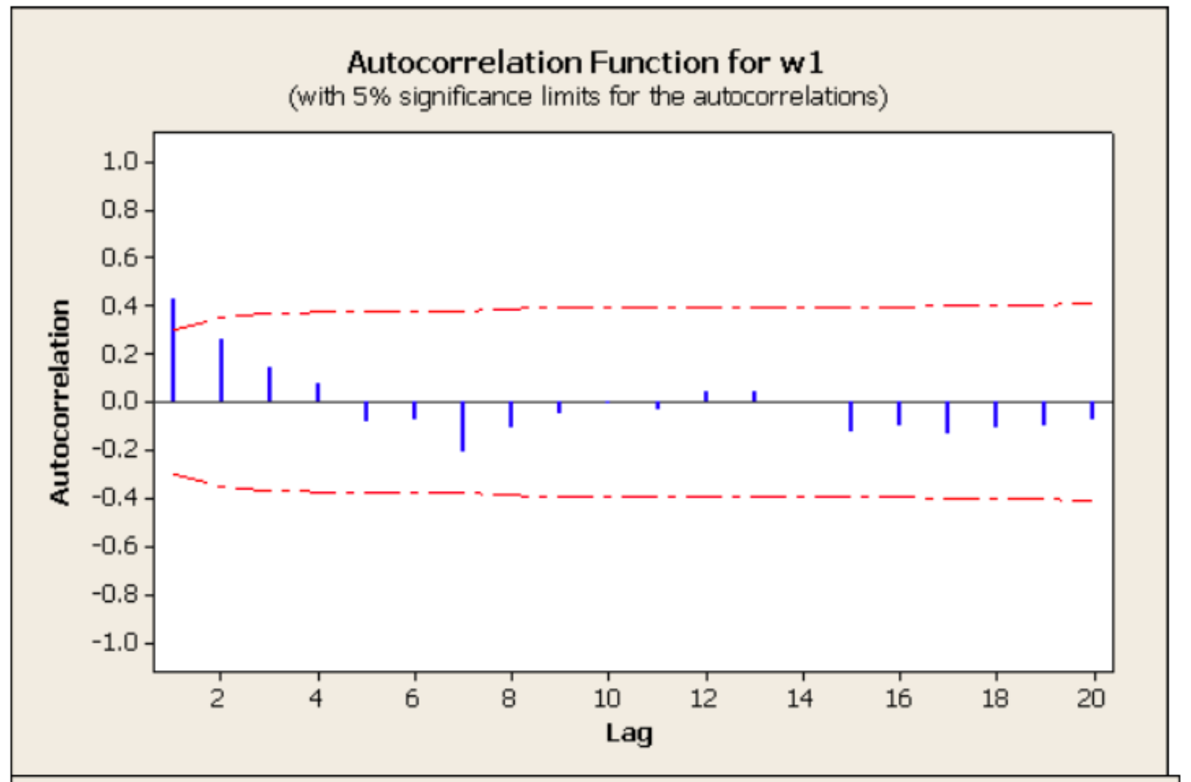
Model Identification	Determine suitable p and q of ARMA(p,q)
Parameter Estimation	Estimate coefficients of ARMA model
Diagnostic Checking	Check if the residuals are white noises
Model Selection	Choose an adequate model from alternatives

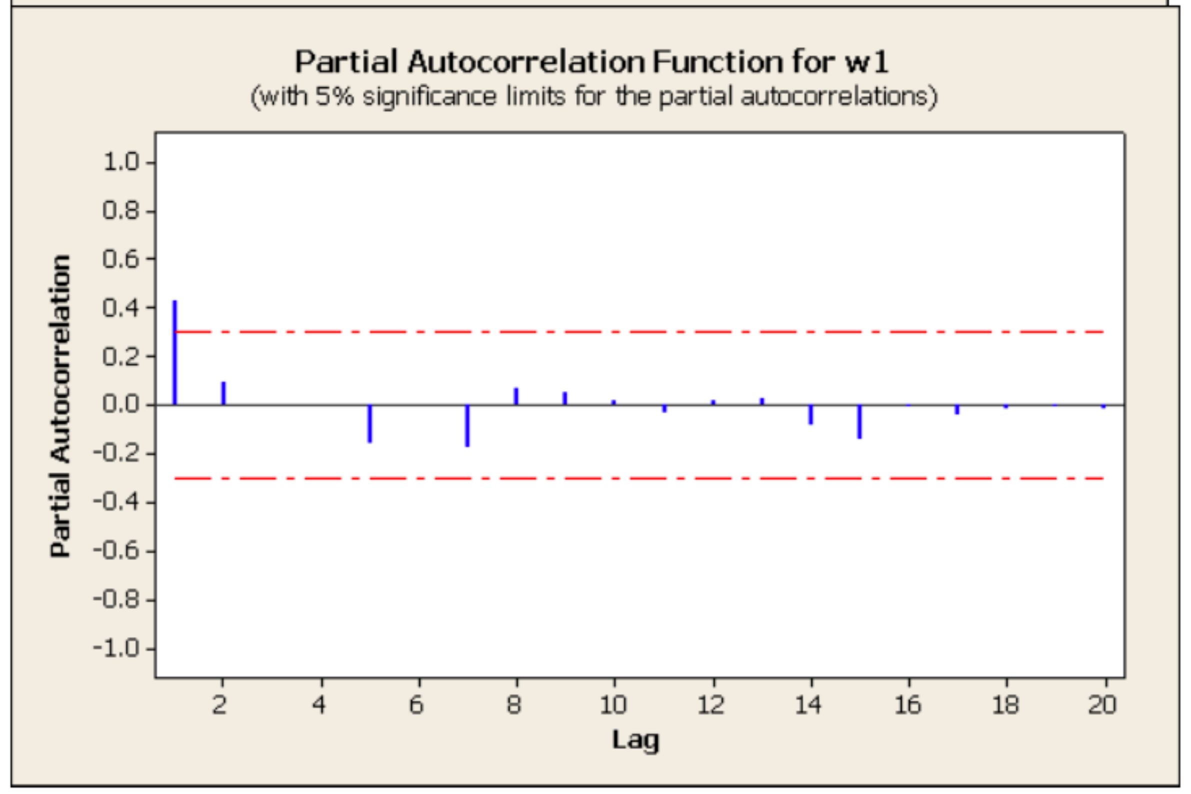
Modeling Process

Model Identification

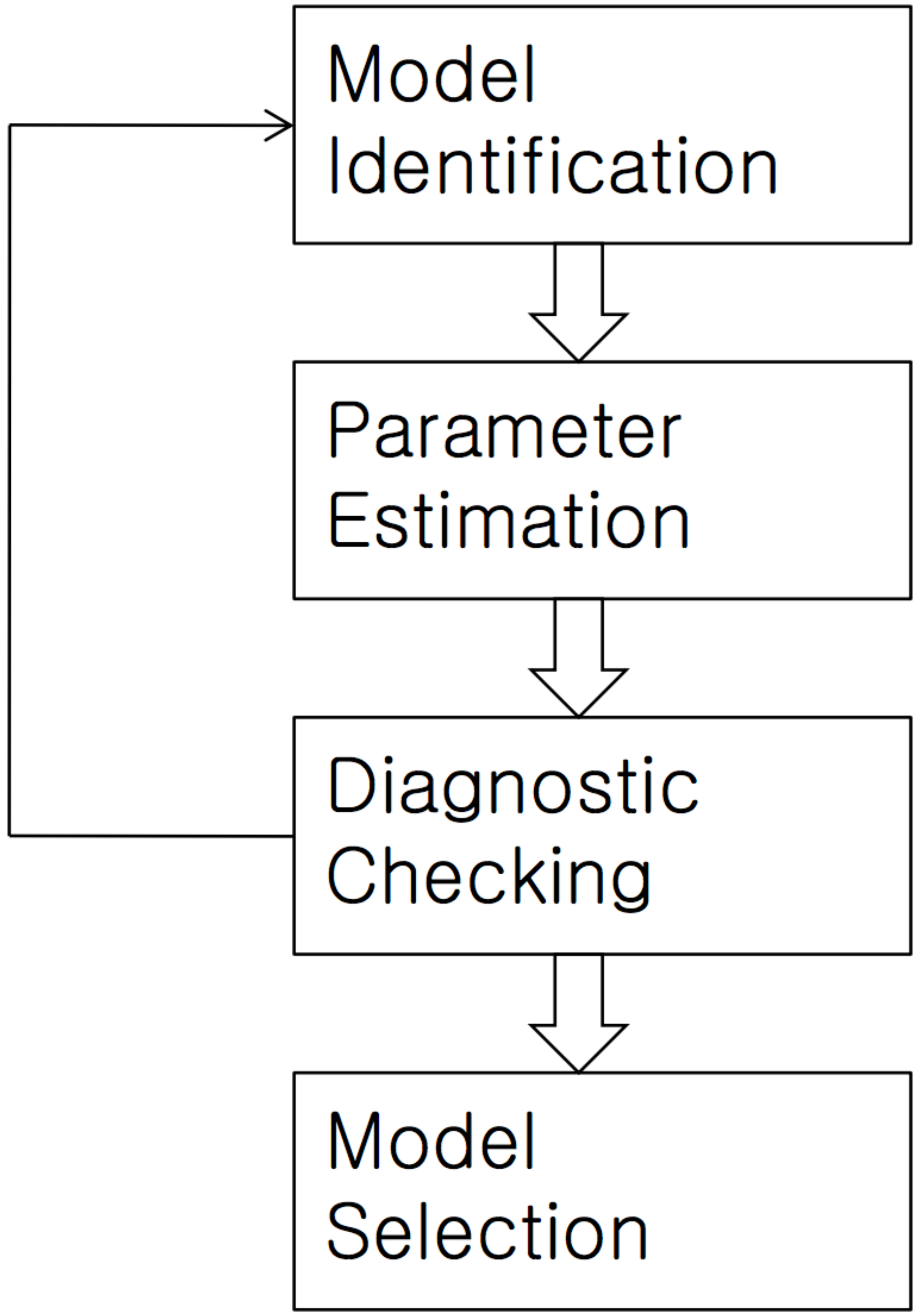
Determine suitable p and q of ARMA(p,q)







Modeling Process



Determine suitable p and q of ARMA(p,q)

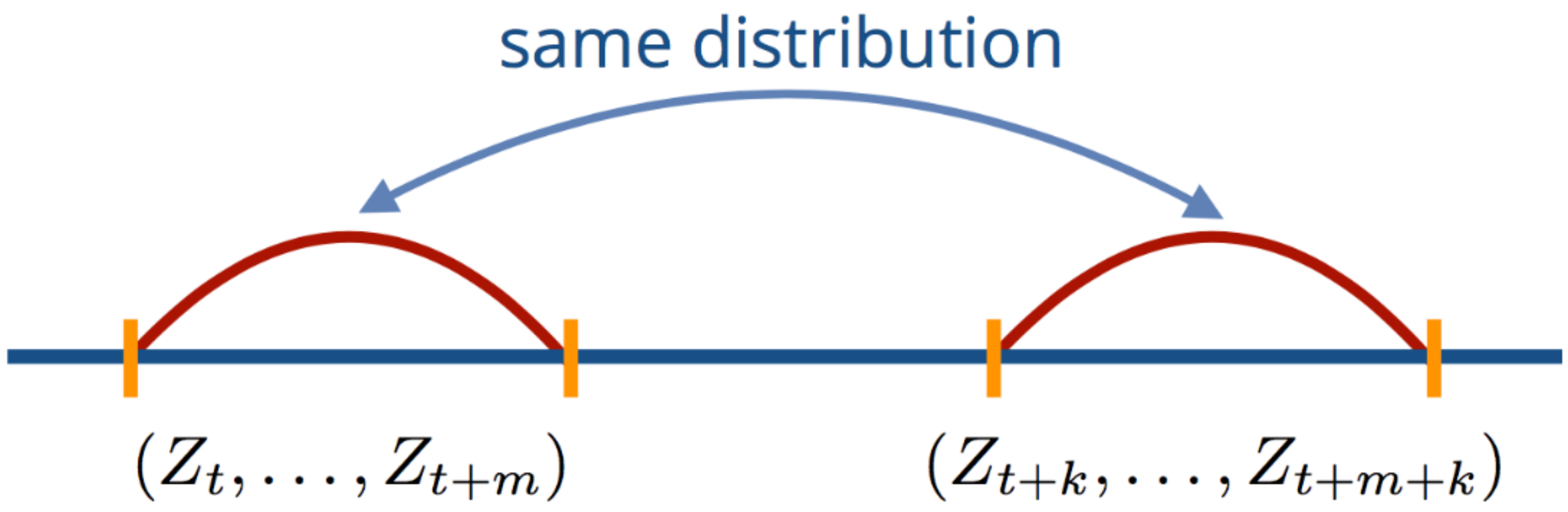
Estimate coefficients of ARMA model

Check if the residuals are white noises

Choose an adequate model from alternatives

Stationarity vs. Non-Stationarity

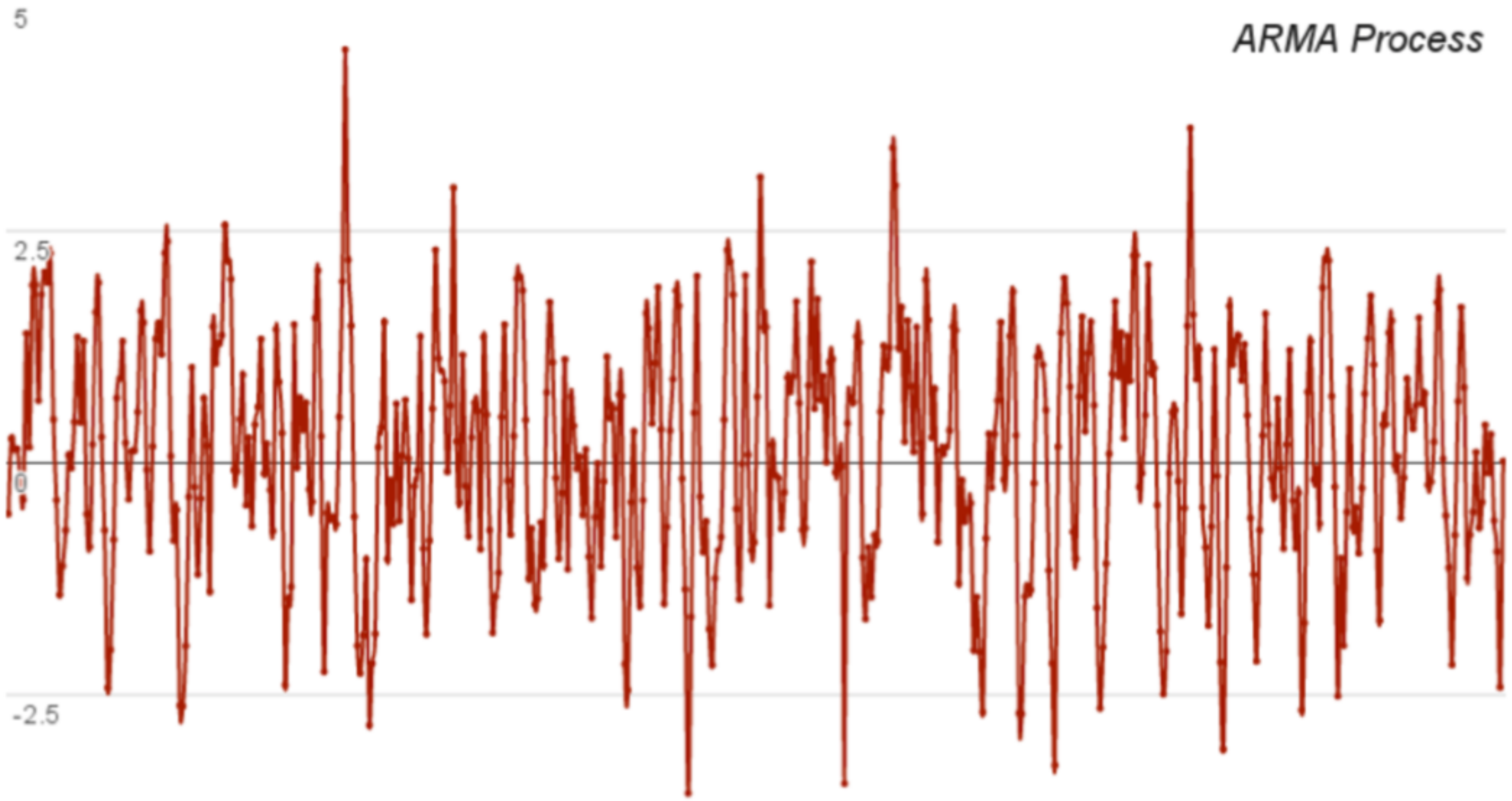
- Stationary Time Series
  - 기간 (Period)에 관계 없이 데이터의 확률 분포가 변하지 않은 시계열 데이터



$$F_{Z_{t_1},\ldots,Z_{t_n}}(x_1,\ldots,x_n) = F_{Z_{t_1+k},\ldots,Z_{t_n+k}}(x_1,\ldots,x_n)$$

Stationarity vs. Non-Stationarity

- Stationary Time Series
  - 기간 (Period)에 관계 없이 데이터의 확률 분포가 변하지 않은 시계열 데이터



Stationarity vs. Non-Stationarity

- Nonstationary Time Series
  - 기간 (Period)에 따라 데이터의 확률 분포가 변하는 시계열 데이터

$(Z_t, \ldots, Z_{t+m})$ $\qquad$ $(Z_{t+k}, \ldots, Z_{t+m+k})$

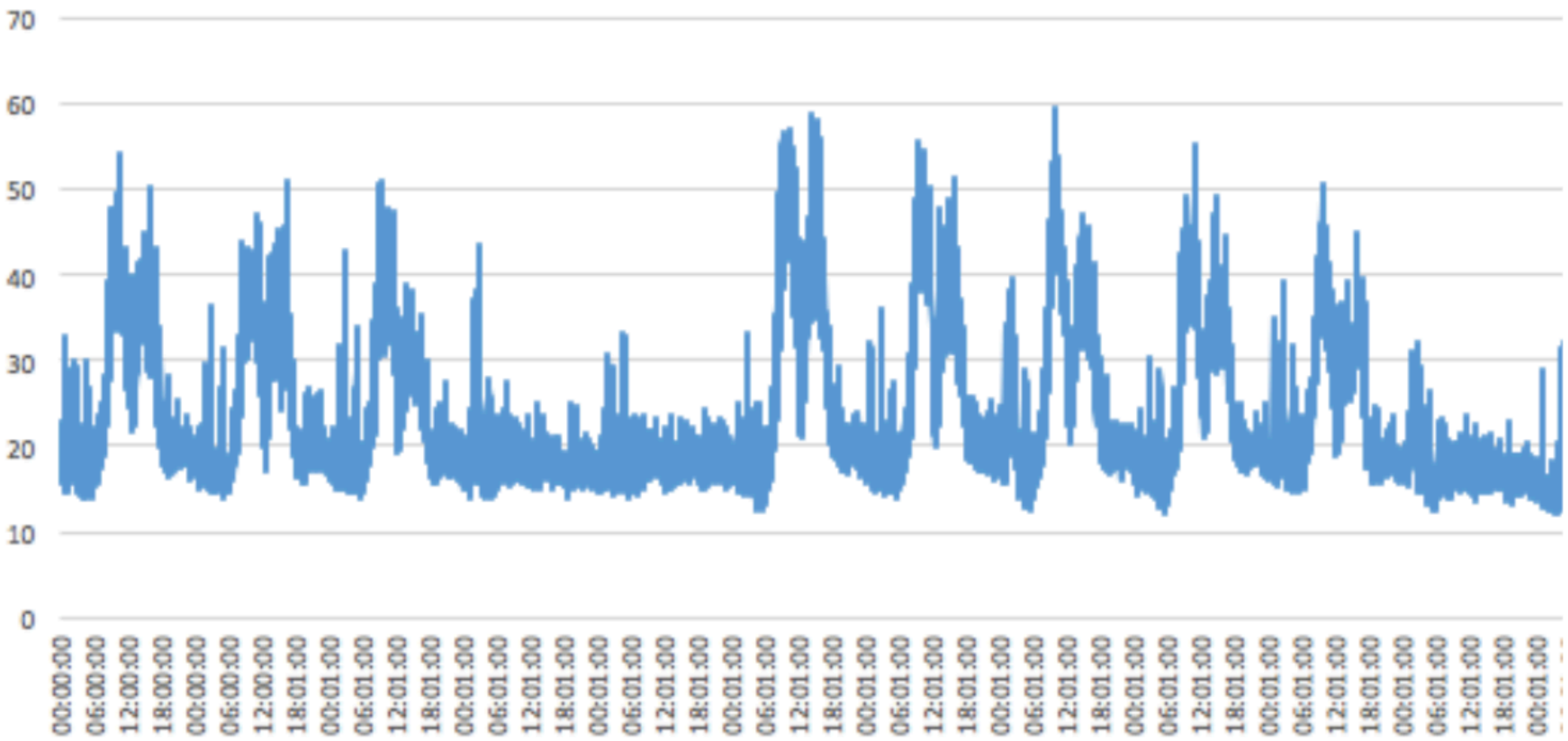
$$F_{Z_{t_1},\ldots,Z_{t_n}}(x_1, \ldots, x_n) \neq F_{Z_{t_1+k},\ldots,Z_{t_n+k}}(x_1, \ldots, x_n)$$

Stationarity vs. Non-Stationarity

- Nonstationary Time Series
  - 기간 (Period)에 따라 데이터의 확률 분포가 변하는 시계열 데이터

Stationarity vs. Non-Stationarity

- Nonstationary Time Series
  - 기간 (Period)에 따라 데이터의 확률 분포가 변하는 시계열 데이터



Stationarity vs. Non-Stationarity

- Nonstationary Time Series
  - 기간 (Period)에 따라 데이터의 확률 분포가 변하는 시계열 데이터

- Typical Examples:

  - Trends

  - Seasonality

  - Changes of Variance

Solutions for Nonstationarity

- Differencing
- Data Transformation
- Seasonal-Trend Decomposition
- (Deep) Neural Networks

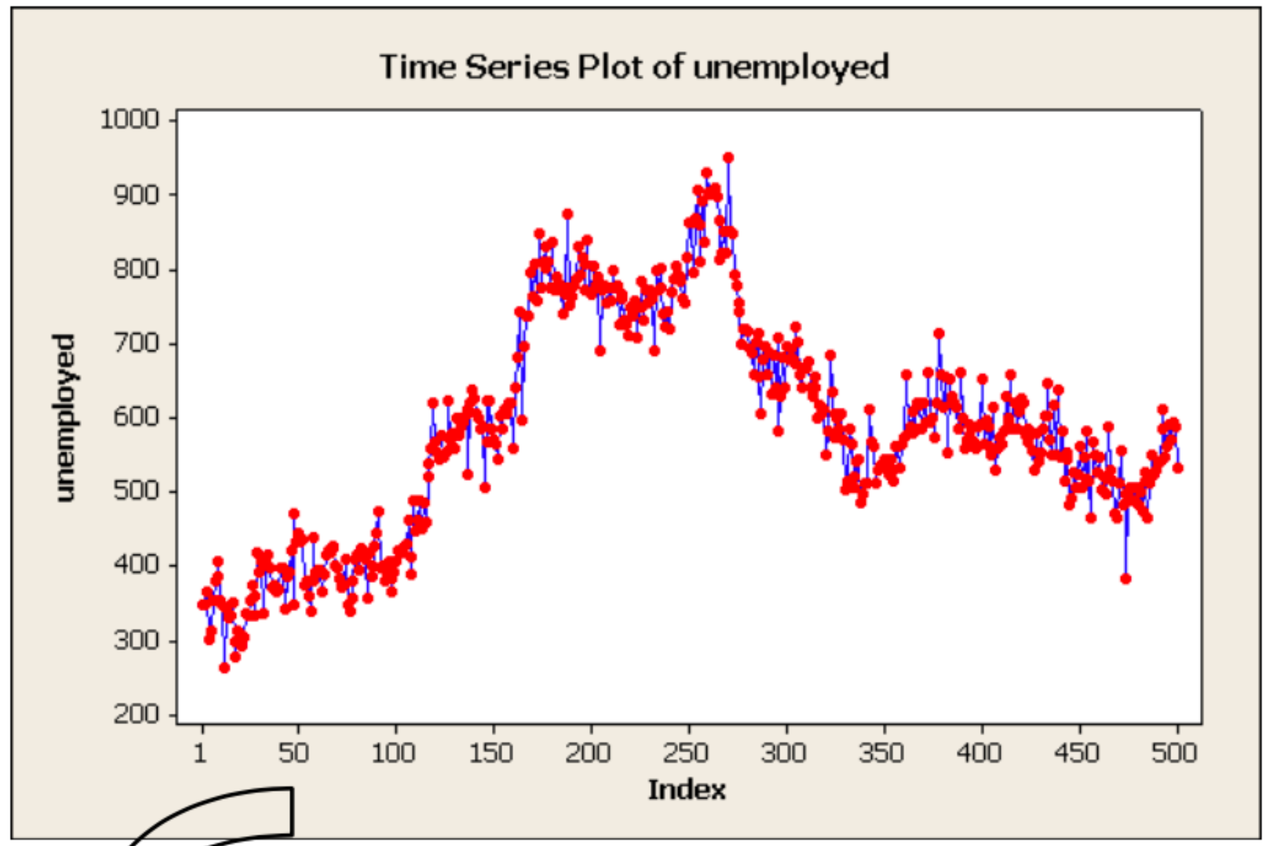
Solutions for Nonstationarity

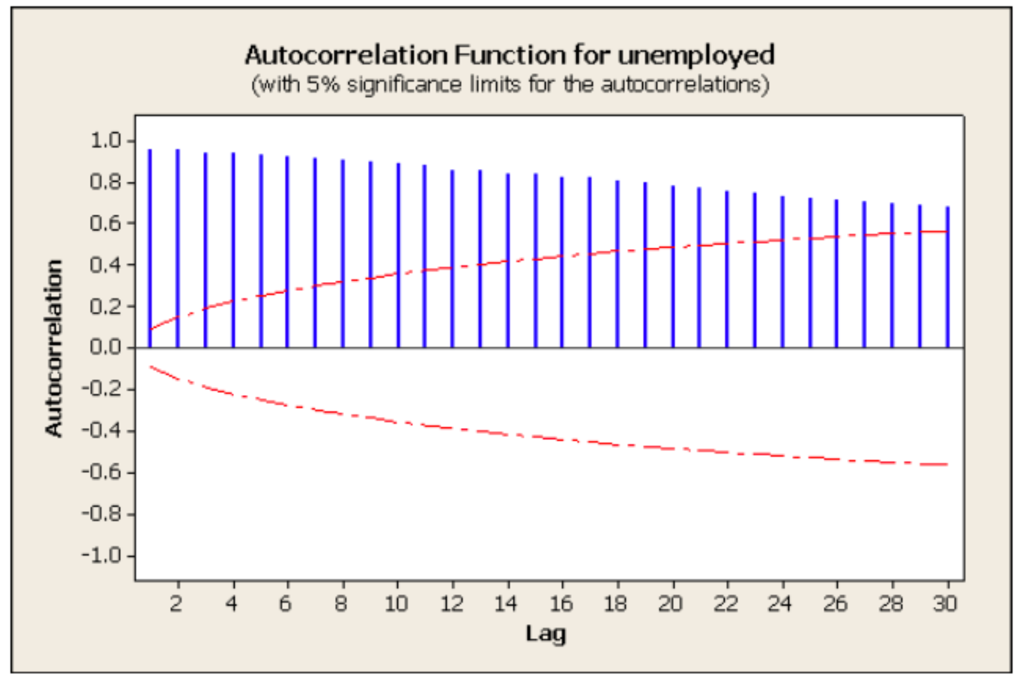
- **Differencing**

- Data Transforn

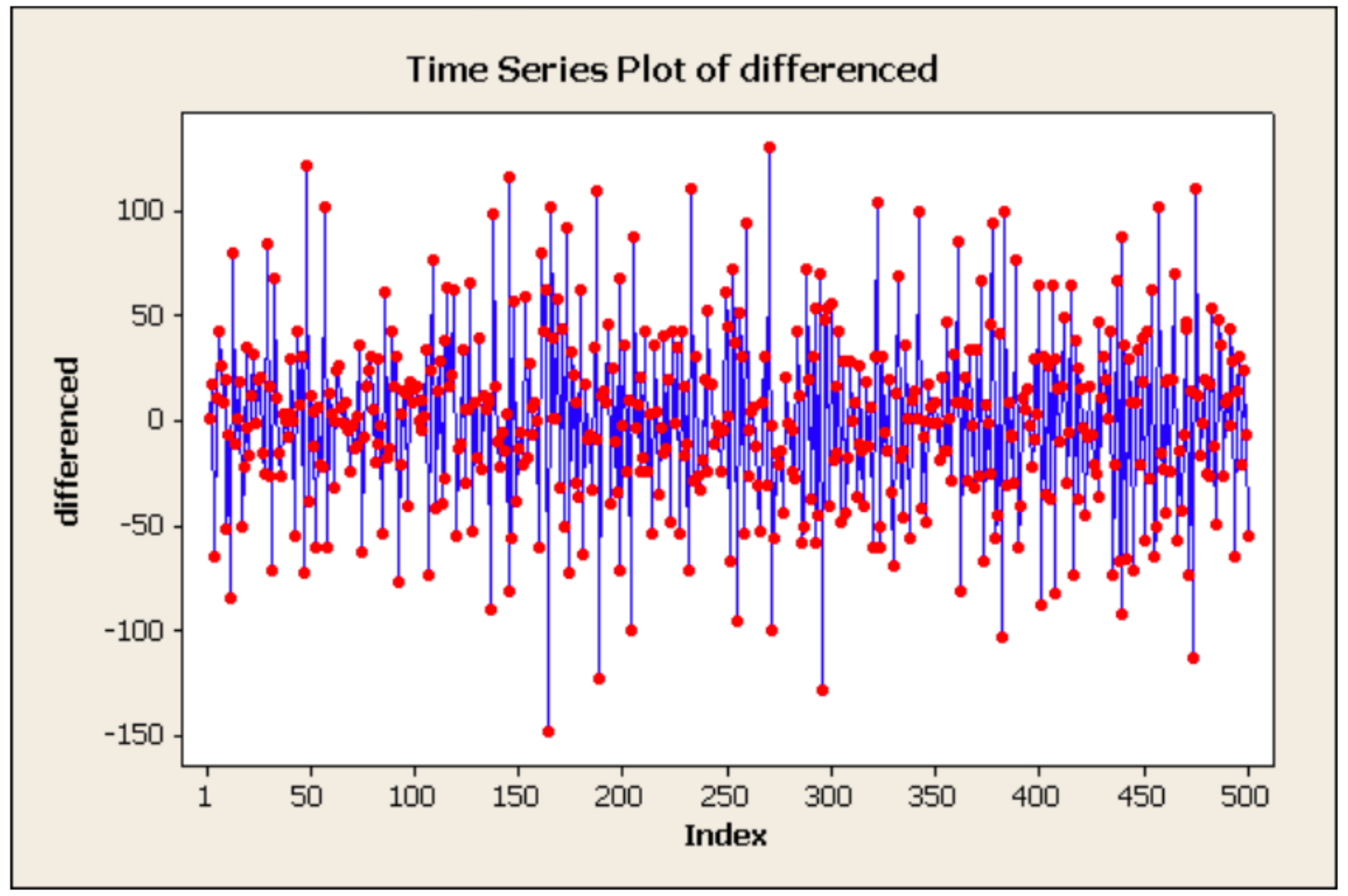
- Seasonal-Tren

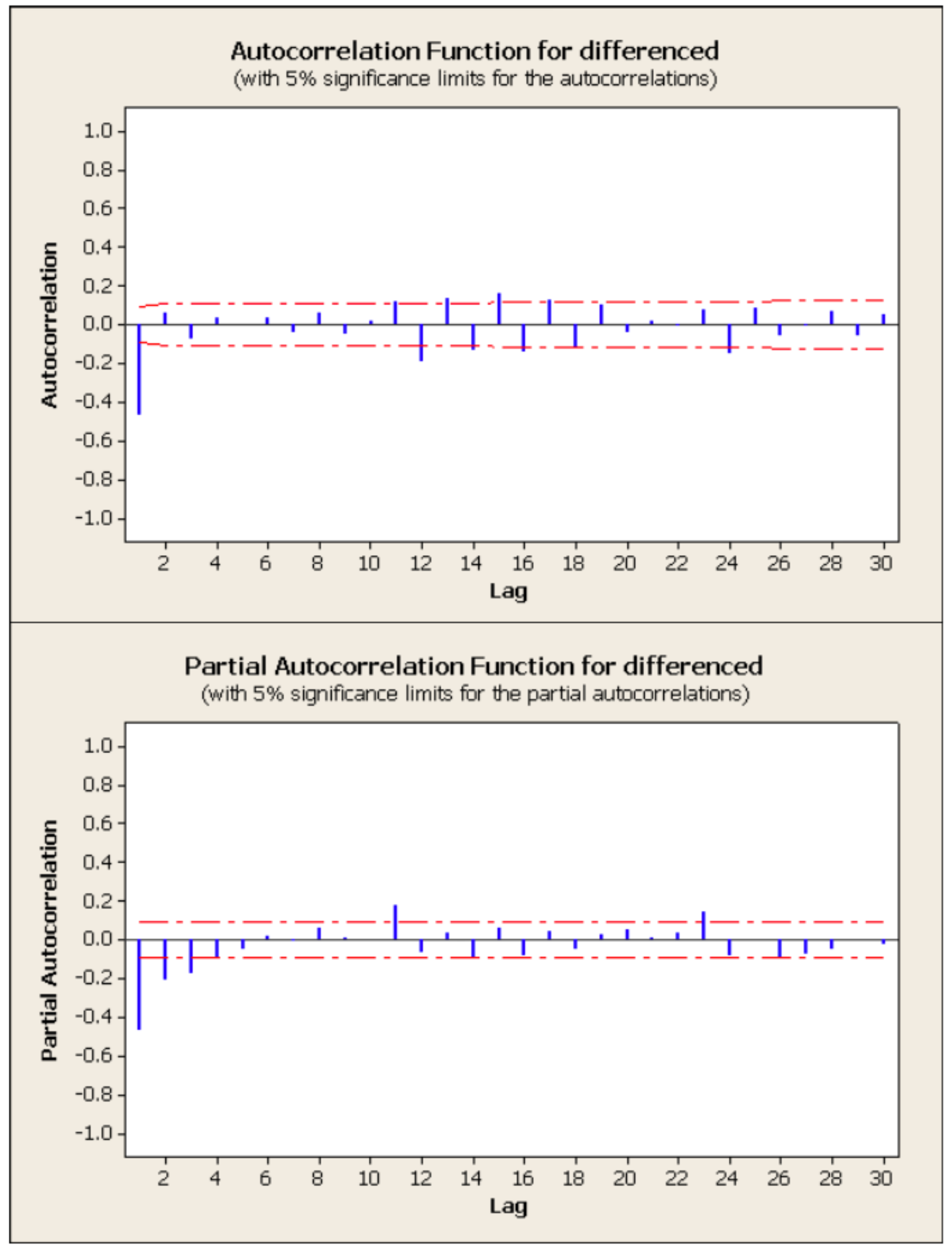
- (Deep) Neural





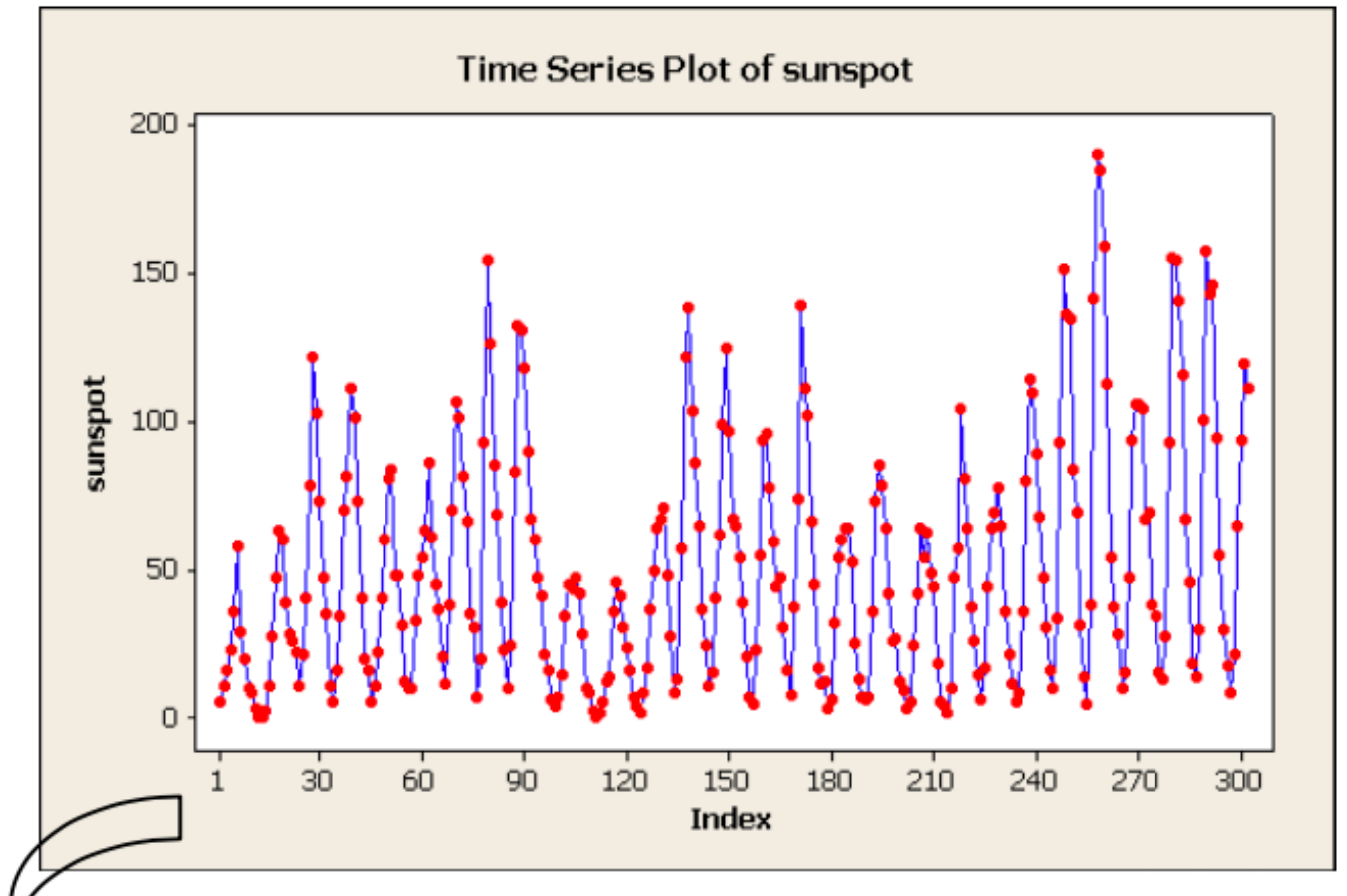
Differen
–cing



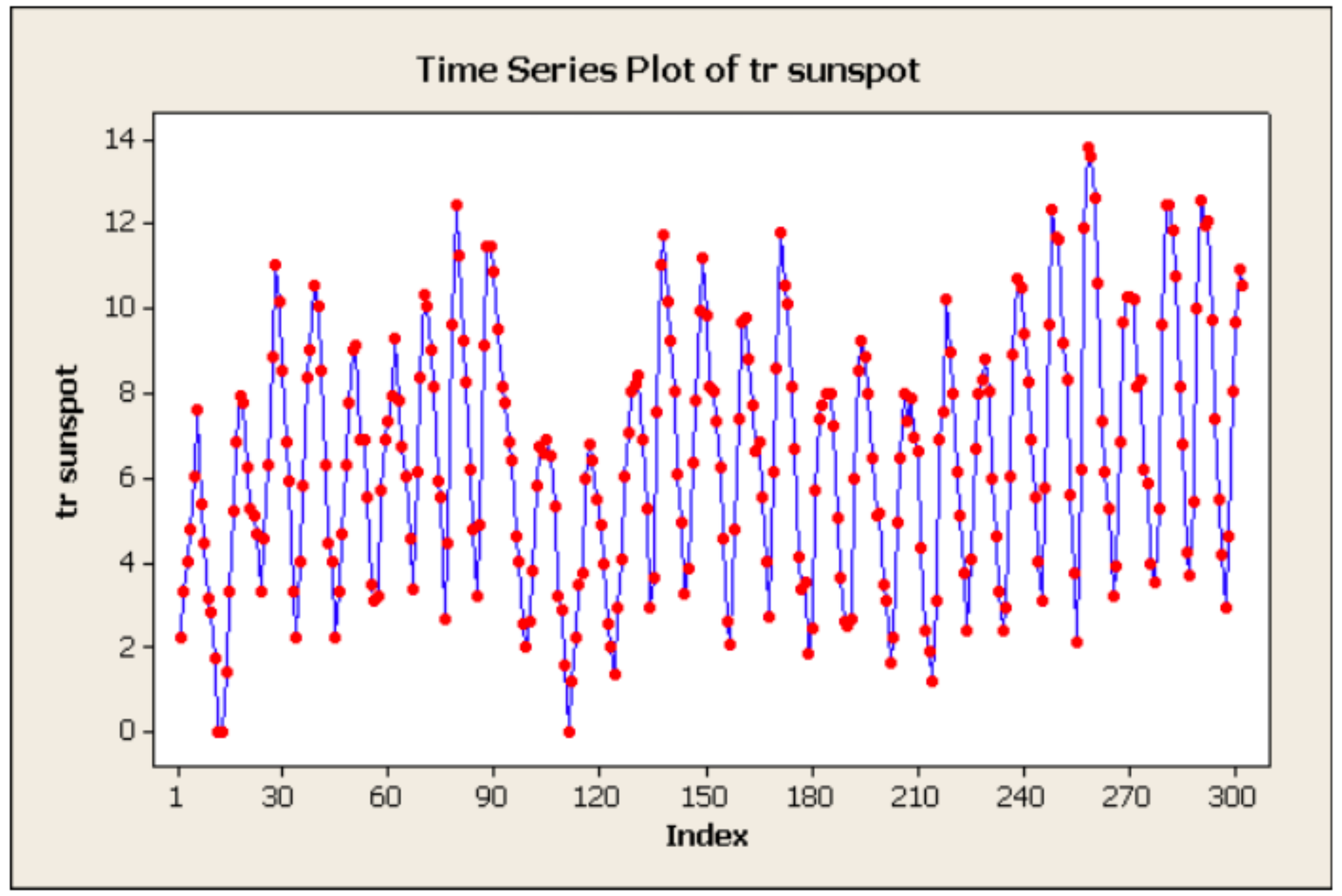


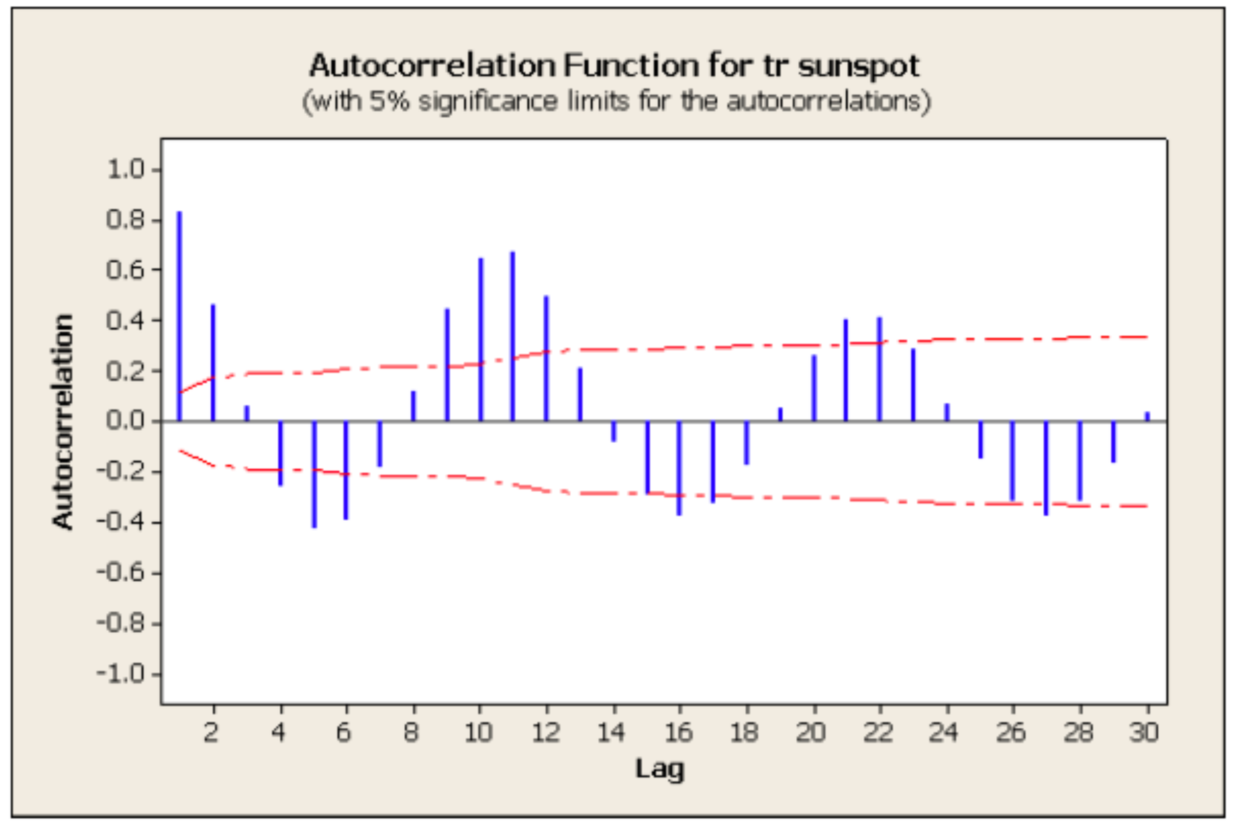
Solutions for Nonstationarity

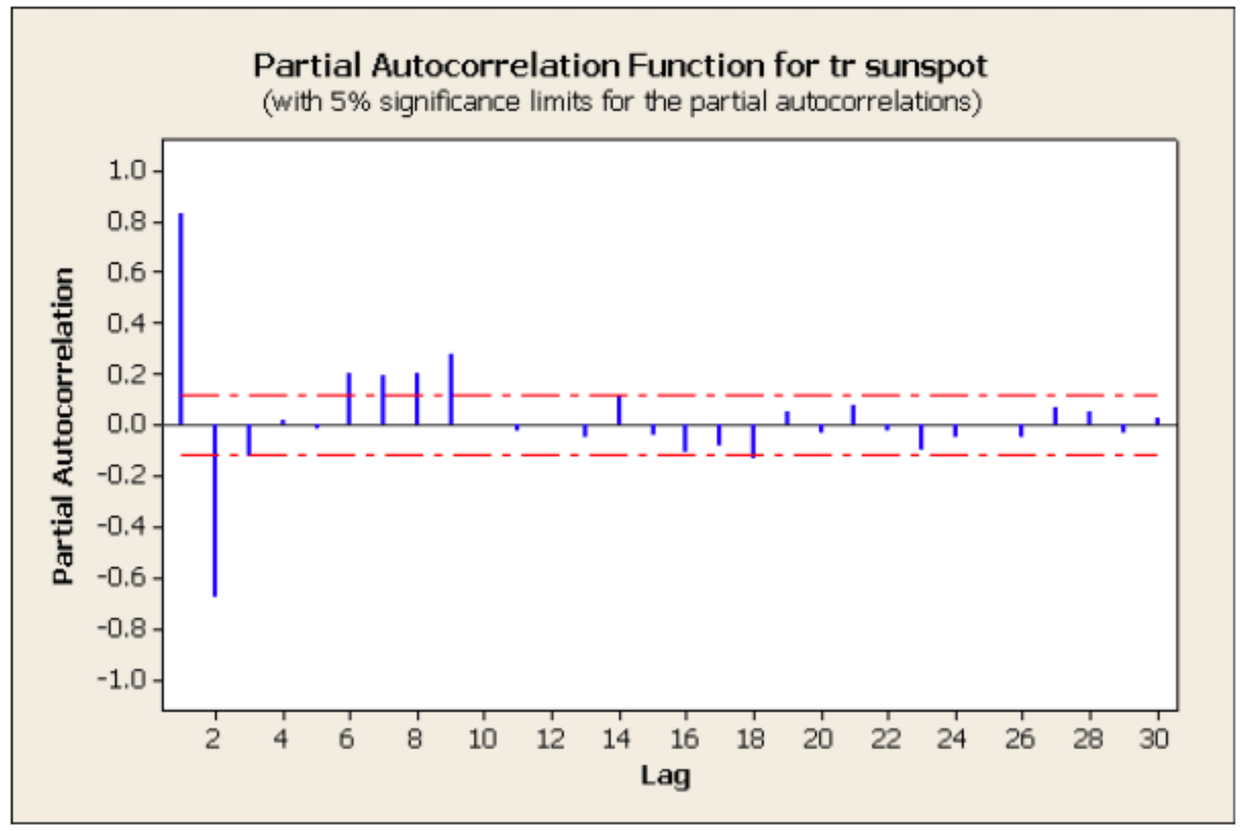
- Differencing
- **Data Transformation**
- Seasonal-Trend Decon
- (Deep) Neural Network



Sq.
root

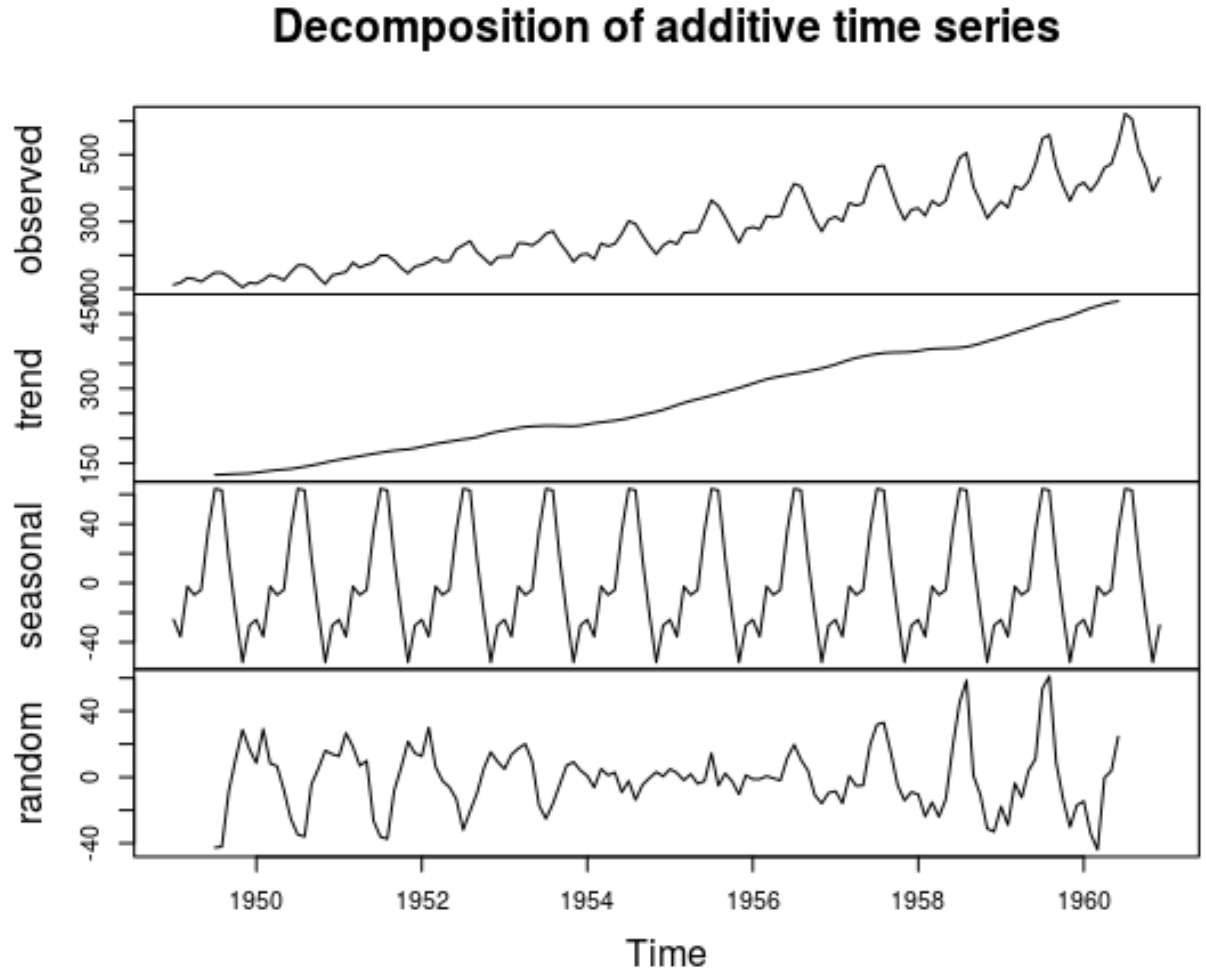






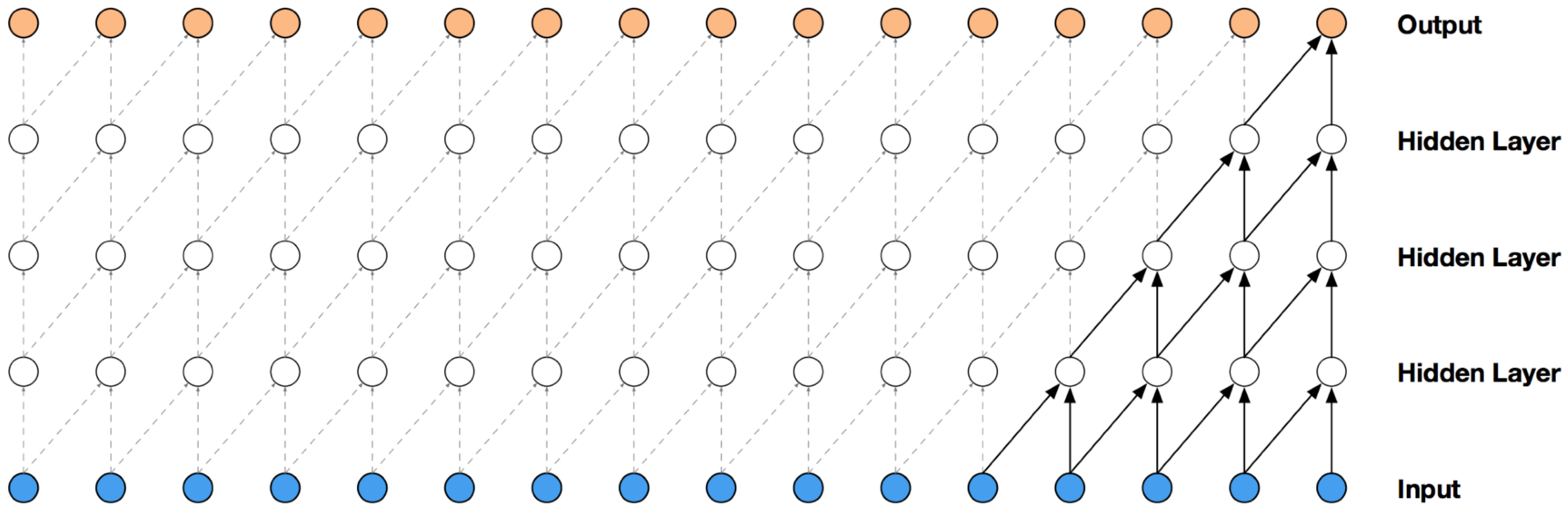
Solutions for Nonstationarity

- Differencing
- Data Transformation
- **Seasonal-Trend Decomposition**
- (Deep) Neural Networks



Solutions for Nonstationarity

- Differencing
- Data Transformation
- Seasonal-Trend Decomposition
- **(Deep) Neural Networks**



https://deepmind.com/blog/wavenet-generative-model-raw-audio/

RobustSTL: A Robust Seasonal-Trend Decomposition Algorithm for Long Time Series

Qingsong Wen, Jingkun Gao, Xiamin Song, Liang Sun, Huan Xu, Shenghuo Zhu
Alibaba
AAAI 2019 paper

Summary

- Decomposing complex time series into **trend**, **seasonality**, and **remainder** components is an important task to facilitate time series anomaly detection and forecasting.

Summary

- Decomposing complex time series into **trend**, **seasonality**, and **remainder** components is an important task to facilitate time series anomaly detection and forecasting.

- **Limitation of previous researches**
  1) Ability to handle seasonality fluctuation and shift, and abrupt changes in trend and reminder
  2) **robustness** of data with anomalies
  3) applicability on time series with **long seasonality** period.

Introduction

- ST Decomposition can reveal the underlying insights of a time series and can be useful in further analysis such as AD and forecasting.

Introduction

- ST Decomposition can reveal the underlying insights of a time series and can be useful in further analysis such as AD and forecasting.

- Without decomposition, it would be missed as its value is still much lower than the unusually high values **during a busy period.**

Introduction

- ST Decomposition can reveal the underlying insights of a time series and can be useful in further analysis such as AD and forecasting.

- Without decomposition, it would be missed as its value is still much lower than the unusually high values **during a busy period.**

- Spike & dip anomalies correspond to abrupt change of remainder and the change of mean anomaly corresponds to abrupt change of trend.

Introduction

- ST Decomposition can reveal the underlying insights of a time series and can be useful in further analysis such as AD and forecasting.

- Without decomposition, it would be missed as its value is still much lower than the unusually high values **during a busy period.**

- Spike & dip anomalies correspond to abrupt change of remainder and the change of mean anomaly corresponds to abrupt change of trend.

- Previous approaches still suffer from less flexibility when seasonality period is long and high noises are observed. Or not feasible on large-size data.

ST decomposition on Real-world

- **3 characteristics of real-world time series**

  1) **Seasonality fluctuation and shift** are quite common in real-world time series.

  2) Most algorithms can't handle **the abrupt change of trend and remainder.**

  3) Most methods are not applicable to **time series with long seasonality period** and some of them can only handle quarterly or monthly data.

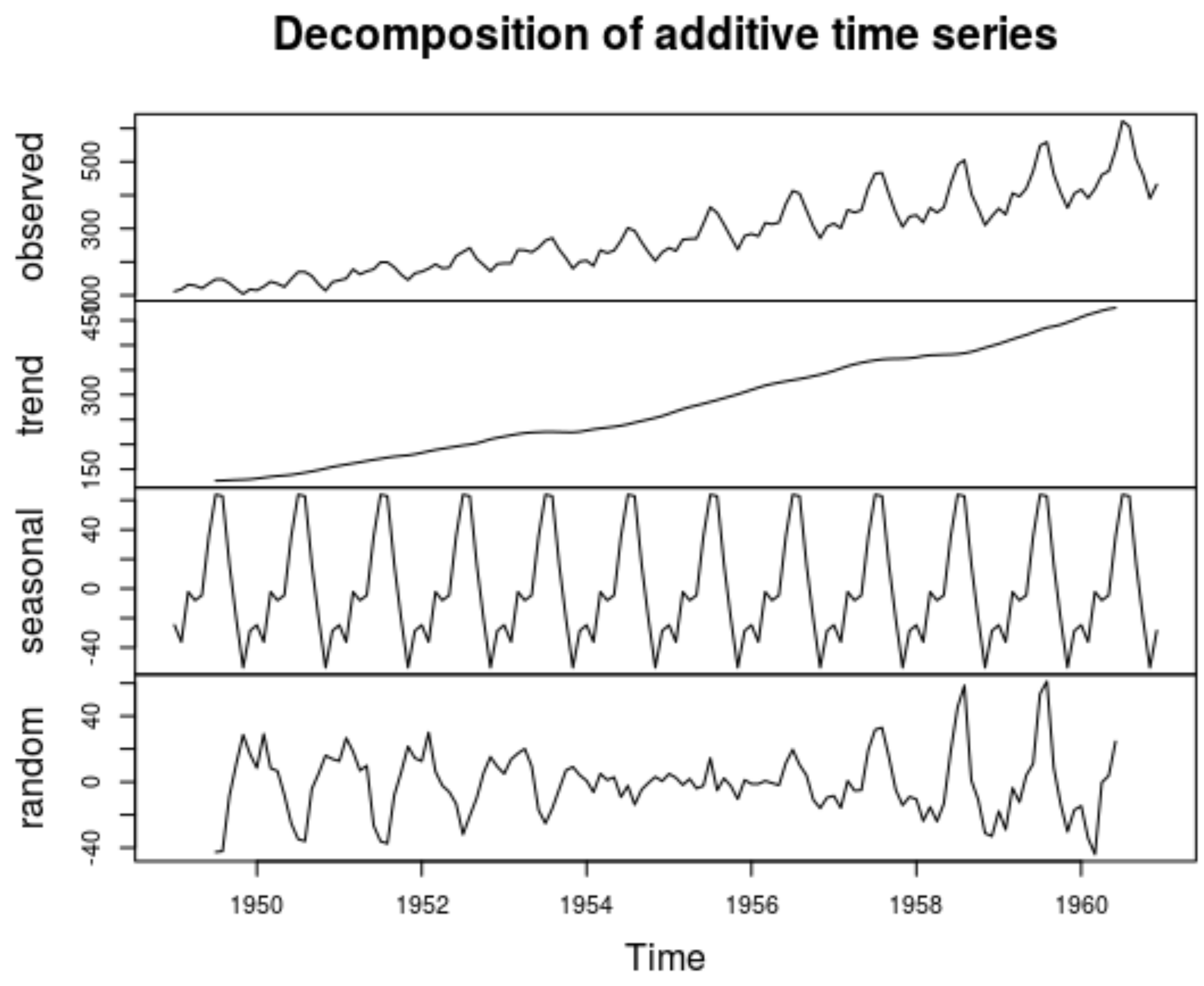
Previous approaches Summary

Table 1: Comparison of different time series decomposition algorithms (Y: Yes / N: No)

Algorithm	Outlier Robustness	Seasonality Shift	Long Period	Abrupt Trend Change
Classical	N	N	N	N
ARIMA/SEATS	N	N	N	N
STL	N	N	Y	N
TBATS	N	N	N	Y
STR	Y	Y	N	N
SSA	N	N	N	N
Our RobustSTL	Y	Y	Y	Y

Robust STL Model Overview

- What we want to do



Robust STL Model Overview

- A value on time t decomposes into (trend, seasonality, and remainder)
  - **Seasonality**: related pattern which changes slowly or even status constant over time.
  - **Trend**: change faster than seasonality.
  - **Remainder**: it consists of anomalies(spikes and dips) and white noise.

$$y_t = \tau_t + s_t + r_t, \quad t = 1, 2, ...N \tag{1}$$

$$r_t = a_t + n_t, \tag{2}$$

RobustSTL algorithm

RobustSTL algorithm

- S1. **Denoise** time series by applying bilateral filtering

RobustSTL algorithm

- S1. **Denoise** time series by applying bilateral filtering
- S2. **Extract trend** robustly by solving a LAD regression with sparse regularizations

RobustSTL algorithm

- S1. **Denoise** time series by applying bilateral filtering
- S2. **Extract trend** robustly by solving a LAD regression with sparse regularizations
- S3. **Calculate the seasonality** component by applying a non-local seasonal filtering to overcome seasonality fluctuation and shift

RobustSTL algorithm

- S1. **Denoise** time series by applying bilateral filtering
- S2. **Extract trend** robustly by solving a LAD regression with sparse regularizations
- S3. **Calculate the seasonality** component by applying a non-local seasonal filtering to overcome seasonality fluctuation and shift
- S4. **Adjust** extracted components (repeat S2 and S3)

S1. Noise Removal

- In real-world applications when time series are collected,
  the observations may be contaminated by carious types of errors and noises.

- Noise removal is indispensable for trend and seasonality decomposition, robustly.

- Many approaches: low-pass filtering, moving/median average, Gaussian filter.

- **The noise removal process "should not" destruct some underlying structuring in trend and seasonal components.**

S1. Noise Removal

- **Bilateral filtering**: cadge-preserving filter in image processing.
  Use neighbors with similar values to smooth the time series.
  The abrupt change of trend and spike and dip can be fully preserved.

S1. Noise Removal

where the $n_t = y_t - \hat{y}_t$ is the filtered noise.

S1. Noise Removal

- **Bilateral filtering**: cadge-preserving filter in image processing.
  Use neighbors with similar values to smooth the time series.
  The abrupt change of trend and spike and dip can be fully preserved.

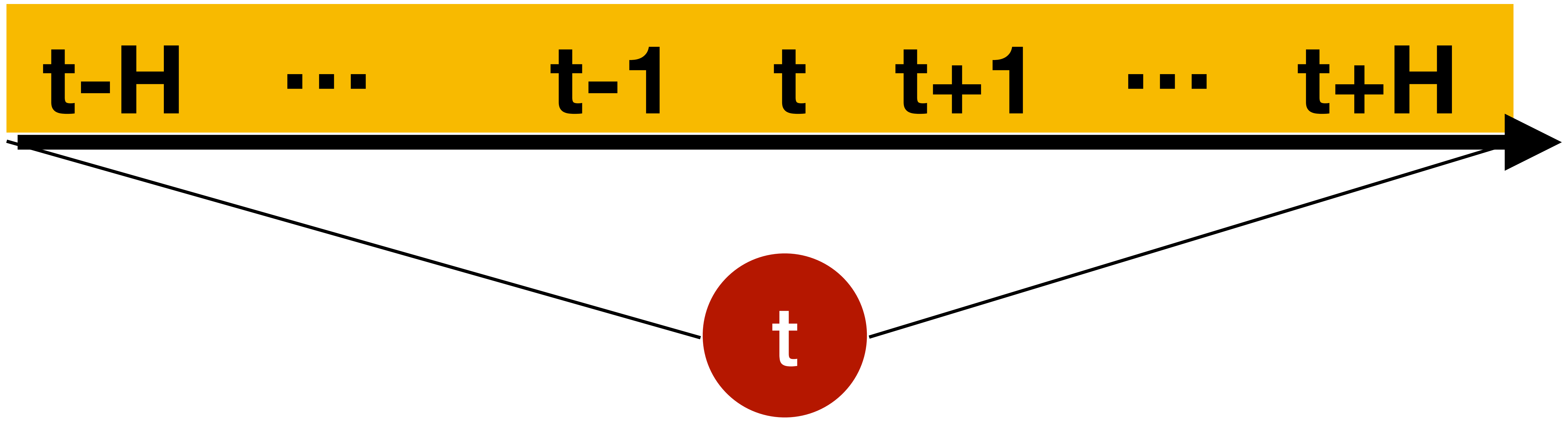
t-H ··· t-1 t t+1 ··· t+H →

S1. Noise Removal

- **Bilateral filtering**: cadge-preserving filter in image processing.
  Use neighbors with similar values to smooth the time series.
  The abrupt change of trend and spike and dip can be fully preserved.

S1. Noise Removal

- **Bilateral filtering**: cadge-preserving filter in image processing.
  Use neighbors with similar values to smooth the time series.
  The abrupt change of trend and spike and dip can be fully preserved.



S1. Noise Removal

- **Bilateral filtering**: cadge-preserving filter in image processing.
  Use neighbors with similar values to smooth the time series.
  The abrupt change of trend and spike and dip can be fully preserved.

S1. Noise Removal

- **Bilateral filtering**: cadge-preserving filter in image processing.
  Use neighbors with similar values to smooth the time series.
  The abrupt change of trend and spike and dip can be fully preserved.

$$y'_t = \sum_{j \in J} w_j^t y_j, \quad J = t, t \pm 1, \cdots, t \pm H \tag{3}$$

where J denotes the filter window with length $2H + 1$, and the filter weights are given by two Gaussian functions as

$$w_j^t = \frac{1}{z} e^{-\frac{|j-t|^2}{2\delta_d^2}} e^{-\frac{|y_j - y_t|^2}{2\delta_i^2}}, \tag{4}$$

S1. Noise Removal

- **Bilateral filtering**: cadge-preserving filter in image processing.
  Use neighbors with similar values to smooth the time series.
  The abrupt change of trend and spike and dip can be fully preserved.

$$y'_t = \sum_{j \in J} w_j^t y_j, \quad J = t, t \pm 1, \cdots, t \pm H \tag{3}$$

where J denotes the filter window with length $2H + 1$, and the filter weights are given by two Gaussian functions as

$$w_j^t = \frac{1}{z} e^{-\frac{|j-t|^2}{2\delta_d^2}} e^{-\frac{|y_j - y_t|^2}{2\delta_i^2}}, \tag{4}$$

$$y'_t = \tau_t + s_t + r'_t$$
$$r'_t = a_t + (n_t - \hat{n}_t)$$

where the $\hat{n}_t = y_t - y'_t$ is the filtered noise.

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

S2. Trend Extraction

- The joint learning of trend and seasonal components is challenging.

- As **the seasonality component is assumed to change slowly**,
  we first perform seasonal difference operation for the despised signal to mitigate the seasonal effects.

S2. Trend Extraction

- Then, the seasonal difference is dominated by trend difference because we assume seasonality and reminder difference are small.

$$
\begin{aligned}
g_t &= \nabla_T y'_t = y'_t - y'_{t-T} \\
&= \nabla_T \tau_t + \nabla_T s_t + \nabla_T r'_t \\
&= \sum_{i=0}^{T-1} \nabla \tau_{t-i} + (\nabla_T s_t + \nabla_T r'_t),
\end{aligned} \tag{7}
$$

S2. Trend Extraction

- Thus, the objective function of trend extraction is to recover the first order difference of trend signal: LAD (robust to outliers)

$$\sum_{t=T+1}^{N} |g_t - \sum_{i=0}^{T-1} \nabla \tau_{t-i}| + \lambda_1 \underbrace{\sum_{t=2}^{N} |\nabla \tau_t|}_{\text{Trend change unit}} + \lambda_2 \underbrace{\sum_{t=3}^{N} |\nabla^2 \tau_t|}_{\text{Smoothness}}, \quad (8)$$

S2. Trend Extraction

- Thus, the objective function of trend extraction is to recover the first order difference of trend signal: LAD (robust to outliers)

$$\sum_{t=T+1}^{N} \left| g_t - \sum_{i=0}^{T-1} \nabla \tau_{t-i} \right| + \lambda_1 \underbrace{\sum_{t=2}^{N} |\nabla \tau_t|}_{\textbf{Trend change unit}} + \lambda_2 \underbrace{\sum_{t=3}^{N} |\nabla^2 \tau_t|}_{\textbf{Smoothness}}, \quad (8)$$

- Second term assumes that the trend difference usually changes slowly but can also exhibit some abrupt level shifts.

- Third term assumes that the trends are smooth and piecewise linear such that sparsity.

S2. Trend Extraction

- Objective with matrix form.

$$||\mathbf{g} - \mathbf{M}\nabla\boldsymbol{\tau}||_1 + \lambda_1||\nabla\boldsymbol{\tau}||_1 + \lambda_2||\mathbf{D}\nabla\boldsymbol{\tau}||_1, \qquad (9)$$

$$\mathbf{g} = [g_{T+1}, g_{T+2}, \cdots, g_N]^T,$$

$$\nabla\boldsymbol{\tau} = [\nabla\tau_2, \nabla\tau_3, \cdots, \nabla\tau_N]^T,$$

$\mathbf{M}$ and $\mathbf{D}$ are $(N-T)\times(N-1)$ and $(N-2)\times(N-1)$ Toeplitz matrix, respectively, with the following forms

$$\mathbf{M} = \begin{bmatrix} \overbrace{1 \quad \cdots \quad 1}^{T \text{ ones}} & & & \\ & 1 & \cdots & 1 & \\ & & \ddots & & \\ & & 1 & \cdots & 1 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} 1 & -1 & & \\ & 1 & -1 & \\ & & \ddots & \\ & & 1 & -1 \end{bmatrix}.$$

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

S2. Trend Extraction

- The optimization problem is equivalent to below.

To facilitate the process of solving the above optimization problem, we further formulate the three ℓ_1-norms in Eq. (9) as a single ℓ_1-norm, i.e.,

$$||\mathbf{P}\nabla\boldsymbol{\tau} - \mathbf{q}||_1, \tag{12}$$

where the matrix $\mathbf{P}$ and vector $\mathbf{q}$ are

$$\mathbf{P} = \begin{bmatrix} \mathbf{M}_{(N-T)\times(N-1)} \\ \lambda_1 \mathbf{I}_{(N-1)\times(N-1)} \\ \lambda_2 \mathbf{D}_{(N-2)\times(N-1)} \end{bmatrix}, \mathbf{q} = \begin{bmatrix} \mathbf{g}_{(N-T)\times 1} \\ \mathbf{0}_{(2N-3)\times 1} \end{bmatrix}.$$

S2. Trend Extraction

- The optimization problem is equivalent to below.

$$\min \quad \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix}^T \begin{bmatrix} \nabla \boldsymbol{\tau} \\ \mathbf{v} \end{bmatrix} \tag{13}$$

$$\text{s.t.} \quad \begin{bmatrix} \mathbf{P} & -\mathbf{I} \\ -\mathbf{P} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \nabla \boldsymbol{\tau} \\ \mathbf{v} \end{bmatrix} \leq \begin{bmatrix} \mathbf{q} \\ -\mathbf{q} \end{bmatrix},$$

$$y''_t = y'_t - \tilde{\tau}^r_t = s_t + \tau_1 + r''_t, \tag{15}$$

$$r''_t = a_t + (n_t - \hat{n}_t) + (\tau_t - \tilde{\tau}_t). \tag{16}$$

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

(Reminder) RobustSTL algorithm

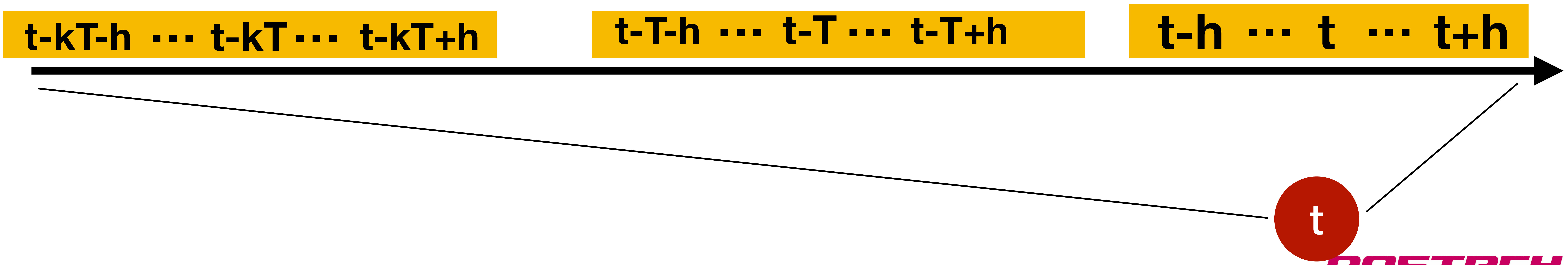
- S1. **Denoise** time series by applying bilateral filtering
- S2. **Extract trend** robustly by solving a LAD regression with sparse regularizations
- S3. **Calculate the seasonality** component by applying a non-local seasonal filtering to overcome seasonality fluctuation and shift
- S4. **Adjust** extracted components (repeat S2 and S3)

S3. Seasonality Extraction

- After de-trending, it can be considered as a contaminated seasonality.
- To consider seasonality shift, **non-local seasonal filtering** is proposed and also consider K neighborhoods centered at seasonal parts.
- In this way, the points with most similar seasonality are automatically found and the seasonality shift problem is solved.

S3. Seasonality Extraction

- After de-trending, it can be considered as a contaminated seasonality.
- To consider seasonality shift, **non-local seasonal filtering** is proposed and also consider K neighborhoods centered at seasonal parts.
- In this way, the points with most similar seasonality are automatically found and the seasonality shift problem is solved.



S3. Seasonality Extraction

- **Non-local Seasonal Filtering**

$$\tilde{s}_t = \sum_{(t',j)\in\Omega} w^t_{(t',j)} y''_j \qquad (17)$$

where the $w^t_{(t',j)}$ and Ω are defined as

$$w^t_{(t',j)} = \frac{1}{z} \boxed{e^{-\frac{|j-t'|^2}{2\delta_d^2}} e^{-\frac{|y''_j - y''_{t'}|^2}{2\delta_i^2}}} \qquad (18)$$

I think the notation has to be t, not t prime. And it's right !

$$\Omega = \{(t',j)|(t' = t - k \times T, j = t' \pm h)\}$$
$$k = 1, 2, \cdots, K; \quad h = 0, 1, \cdots, H$$

S3. Seasonality Extraction

- **Non-local Seasonal Filtering**

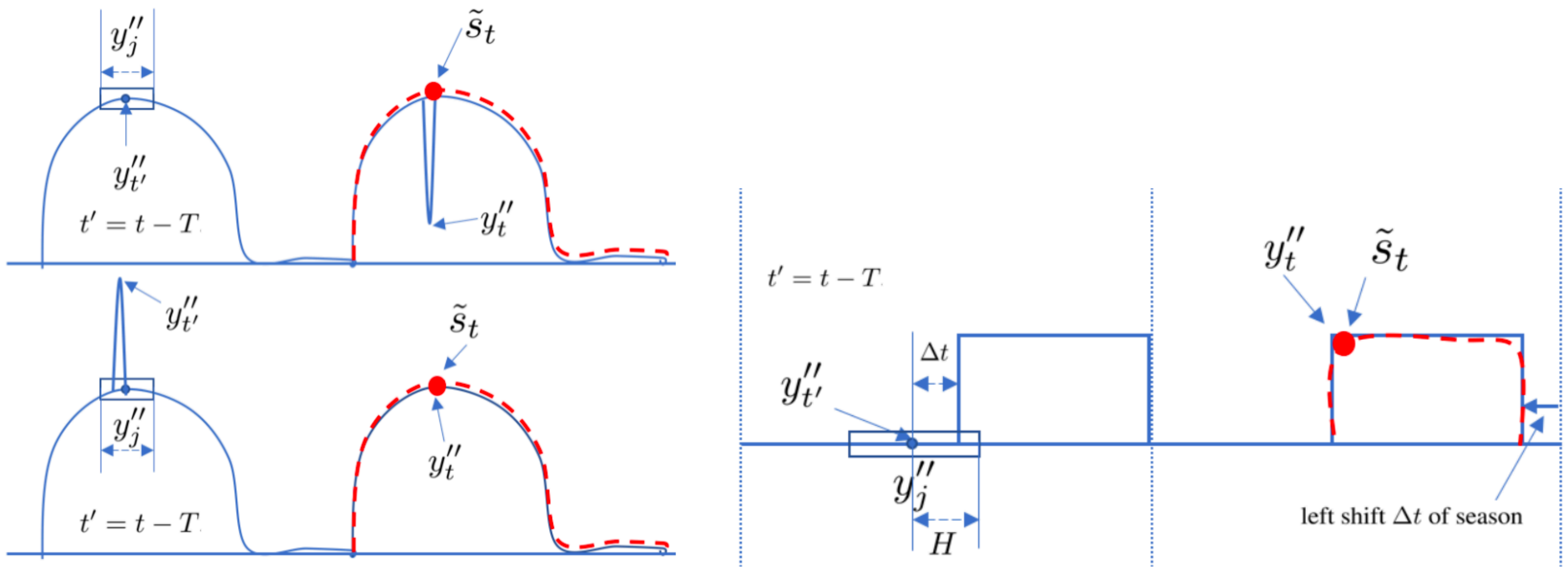
$$\tilde{s}_t = \sum_{(t',j)\in\Omega} w^t_{(t',j)} y''_j \tag{17}$$

where the $w^t_{(t',j)}$ and Ω are defined as

$$\begin{aligned} w^t_{(t',j)} &= \frac{1}{z} e^{-\frac{|j-t\ |^2}{2\delta_d^2}} e^{-\frac{|y''_j - y''_{t\cdot}|^2}{2\delta_i^2}} \\ \Omega &= \{(t',j)|(t' = t - k \times T, j = t' \pm h)\} \\ k &= 1, 2, \cdots, K; \quad h = 0, 1, \cdots, H \end{aligned} \tag{18}$$

S3. Seasonality Extraction

- Robustness of non-local seasonal filtering to outliers.



(a) Outlier robustness (b) Season shift adaptation

Figure 1: Robust and adaptive properties of the non-local seasonal filtering (red curve denotes the extracted seasonal signal).

S4. Final Adjustment

- To make seasonal-trend decomposition unique, RobustSTL makes **the sum of seasonality components become zero**, using mean shift.

$$\hat{\tau}_1 = \frac{1}{T\lfloor N/T \rfloor} \sum_{t=1}^{T\lfloor N/T \rfloor} \tilde{s}_t.$$

$$\hat{s}_t = \tilde{s}_t - \hat{\tau}_1,$$

$$\hat{\tau}_t = \tilde{\tau}_t^r + \hat{\tau}_1.$$

$$\hat{r}_t = r_t''' + \hat{n}_t \quad \text{or} \quad \hat{r}_t = y_t - \hat{s}_t - \hat{\tau}_t.$$

$$\hat{r}_t = \begin{cases} a_t + n_t + (s_t - \hat{s}_t) + (\tau_1 - \hat{\tau}_1), & t = 1 \\ a_t + n_t + (s_t - \hat{s}_t) + (\tau_t - \tilde{\tau}_t), & t \geq 2 \end{cases}$$

Algorithm Summary

Algorithm 1 RobustSTL Algorithm Summary

Input: y_t, parameter configurations.
Output: $\hat{\tau}_t, \hat{s}_t, \hat{r}_t$

Step 1: Denoise input signal using bilateral filter

$$w_j^t = \frac{1}{z} e^{-\frac{|j-t|^2}{2\delta_d^2}} e^{-\frac{|y_j - y_t|^2}{2\delta_i^2}}, \quad y_t' = \sum_{j \in J} w_j^t y_j$$

Step 2: Obtain relative trend from ℓ_1 sparse model

$$\nabla\tilde{\boldsymbol{\tau}} = \arg\min_{\nabla\boldsymbol{\tau}} ||\mathbf{P}\nabla\boldsymbol{\tau} - \mathbf{q}||_1 \text{(see Eq. (8), (9), (12))}$$

$$\tilde{\tau}_t^r = \begin{cases} 0, & t = 1 \\ \sum_{i=2}^{t} \nabla\tilde{\tau}_i, & t \geq 2 \end{cases},$$

$$y_t'' = y_t' - \tilde{\tau}_t^r$$

Step 3: Obtain season using non-local seasonal filtering

$$w_{(t',j)}^t = \frac{1}{z} e^{-\frac{|j-t'|^2}{2\delta_d^2}} e^{-\frac{|y_j'' - y_{t'}''|^2}{2\delta_i^2}}$$

$$\tilde{s}_t = \sum_{(t',j)\in\Omega} w_{(t',j)}^t y_j''$$

Step 4: Adjust trend and season

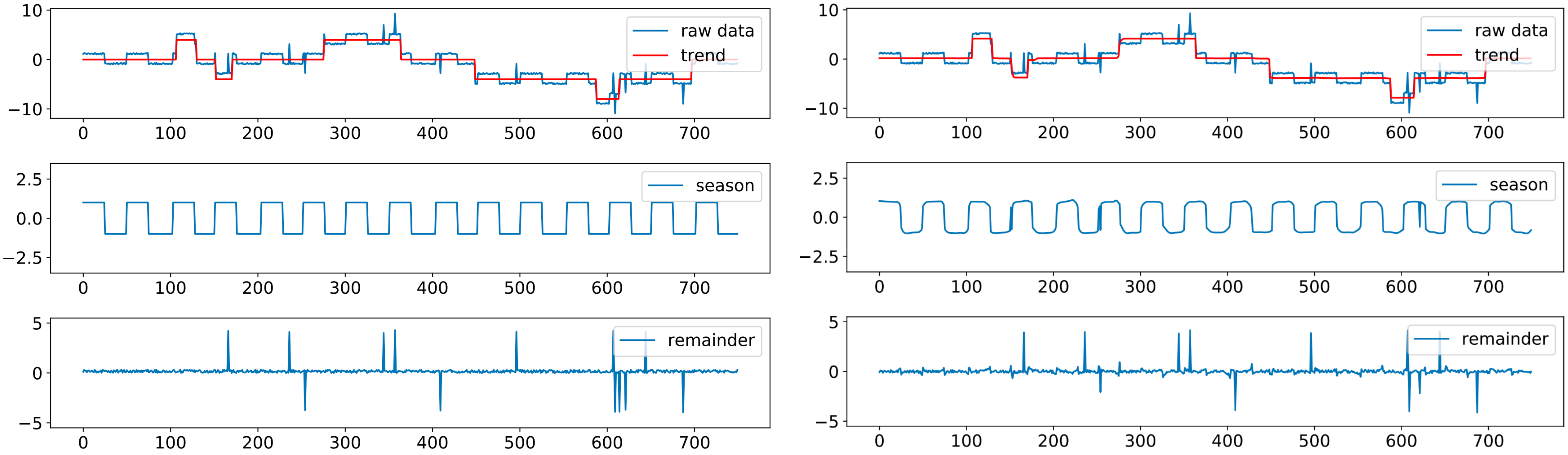
$$\hat{\tau}_1 = \frac{1}{T\lfloor N/T \rfloor} \sum_{t=1}^{T\lfloor N/T \rfloor} \tilde{s}_t$$

$$\hat{\tau}_t = \tilde{\tau}_t^r + \hat{\tau}_1, \; \hat{s}_t = \tilde{s}_t - \hat{\tau}_1, \; \hat{r}_t = y_t - \hat{s}_t - \hat{\tau}_t$$

Step 5: Repeat Steps 1-4 for $\hat{r}_t$ until convergence

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

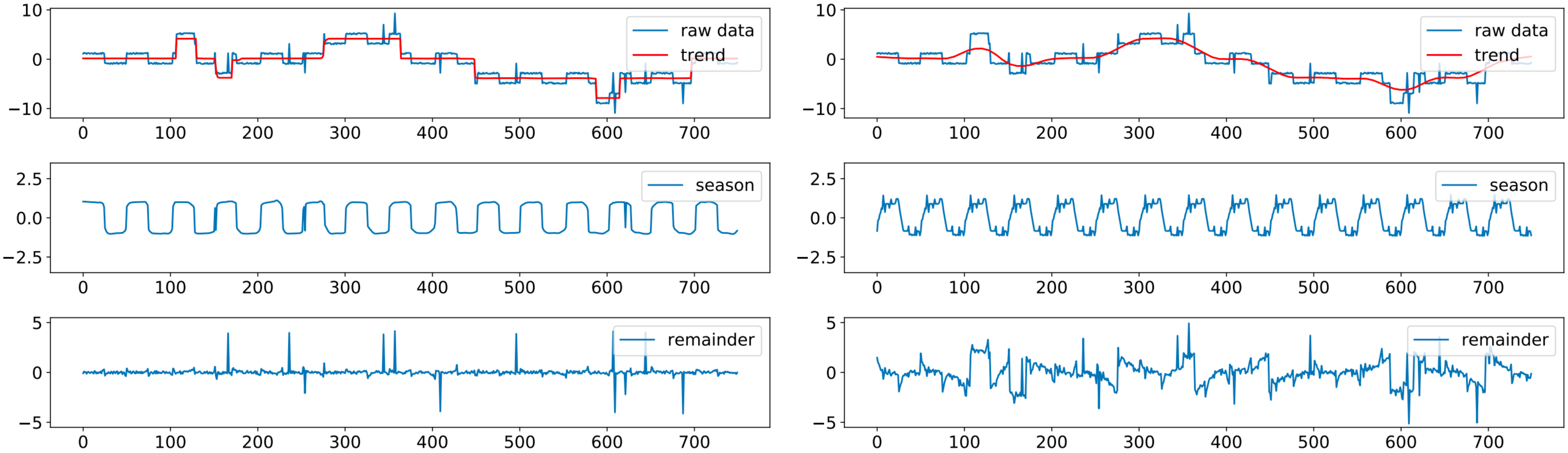
Results - Synthetic Data



(a) RobustSTL

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

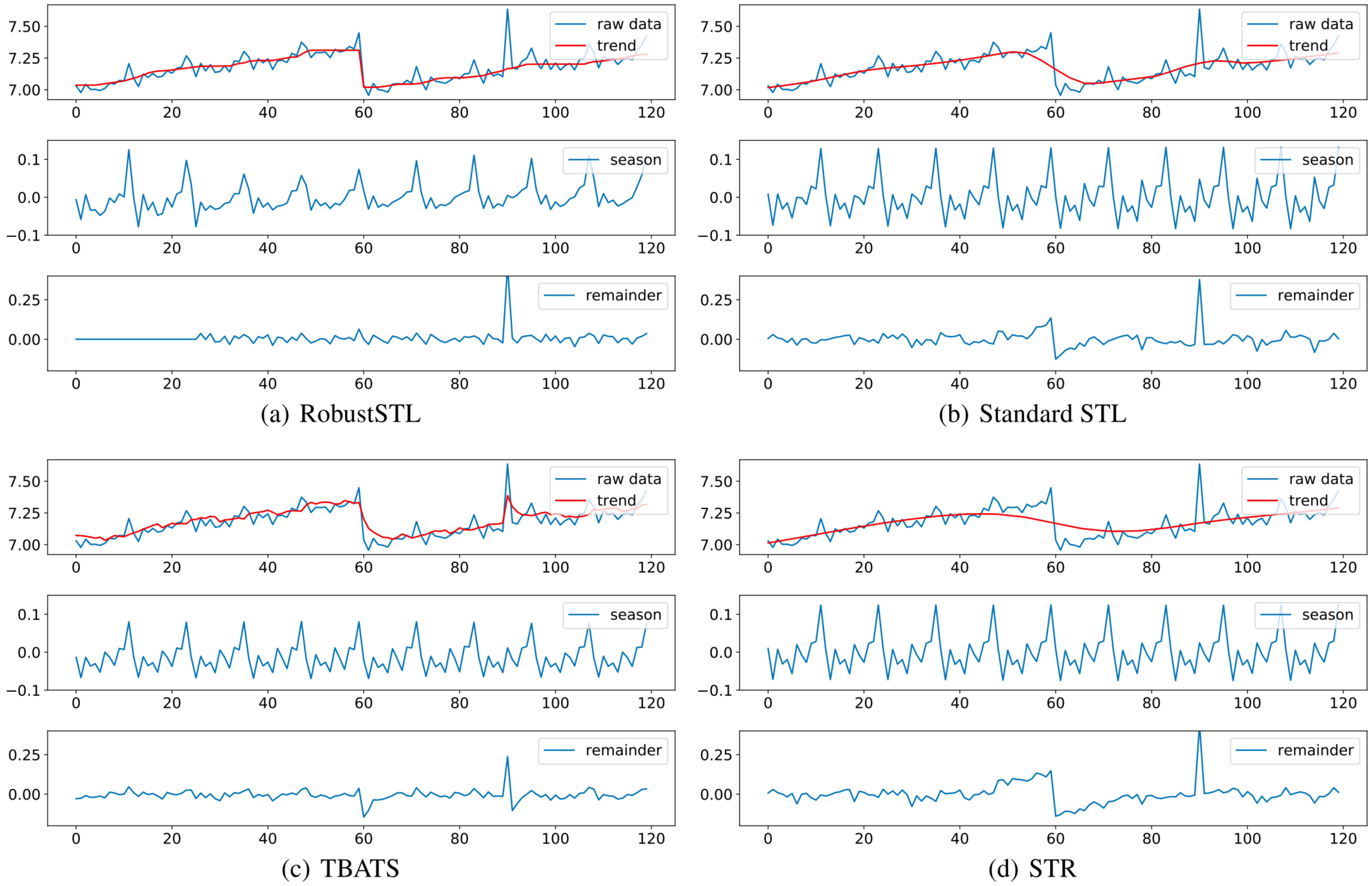
Results - Synthetic Data



(a) RobustSTL (b) Standard STL

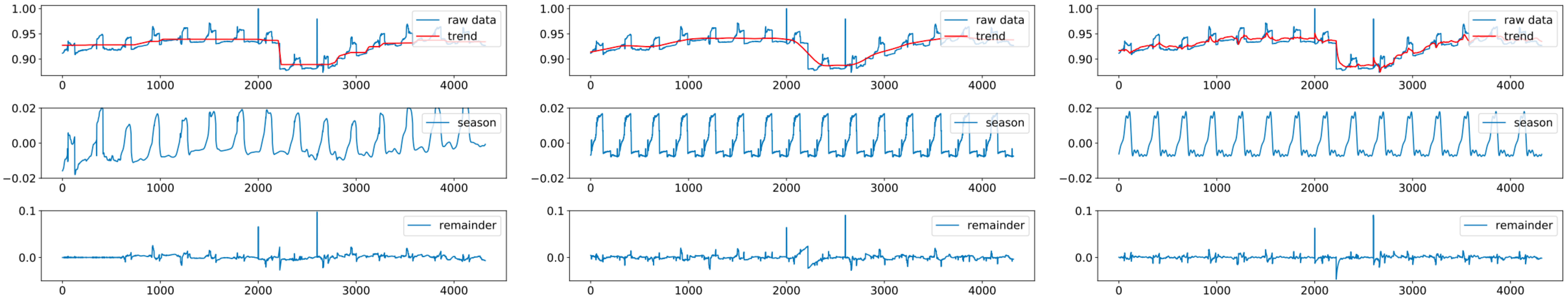
POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Results - Real-world Data 1



POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Results - Real-world Data 2



(a) RobustSTL (b) Standard STL (c) TBATS

Implementations

- codes: https://www.github.com/LeeDoYup/RobustSTL

RobustSTL: A Robust Seasonal-Trend Decomposition Algorithm for Long Time Series (AAAI 2019)

This repository contains python (3.5.2) implementation of RobustSTL (paper) .

Decomposing complex time series into trend, seasonality, and remainder components is an important task to facilitate time series anomaly detection and forecasting.
RobustSTL extract trend using LAD loss with sparse regularization and non-local seasonal filtering.
Compared to previous approaches (such as traditional STL), RobustSTL has advantages on

1. Ability to handle seasonality fluctuation and shift, and abrupt change in trend and reminder
2. robustness of data with anomalies
3. applicability on time series with long seasonality period.

Requirments & Run

First, install some required libraries using pip.

```
pip3 install -r requirments.txt
python3 main.py
```

Sample Results

We generate a synthetic sample (sample_generator.py) and decompose it into `trend`, `seasonality`, and `remainder`. In `run_example.ipynb`, I attach the example codes to use RobustSTL and the outputs.

Some Open Questions

- Real Time applications ?
- Neural Networks ?
- Higher frequency time series ?
- Does really the algorithm work well? (Some problems yet..)

Thank you.