A gentle introduction to

Time Series

Wize

DML

Speakers

Parsa Haghighi

M.Sc Student
Researcher in DML lab

haghighi@sharif.edu

Armin Behnamnia

Ph.D Student
Researcher in DML lab
Data Scientist at Wize Analytics
arminbehnamnia@gmail.com

Outline: First half

- Introduction
- Some Applications
- Common Patterns
- Evaluation Metrics
- Stationary Time Series
    - Definition
    - Statistical Tests
    - Converting Non-stationary to Stationary

Introduction

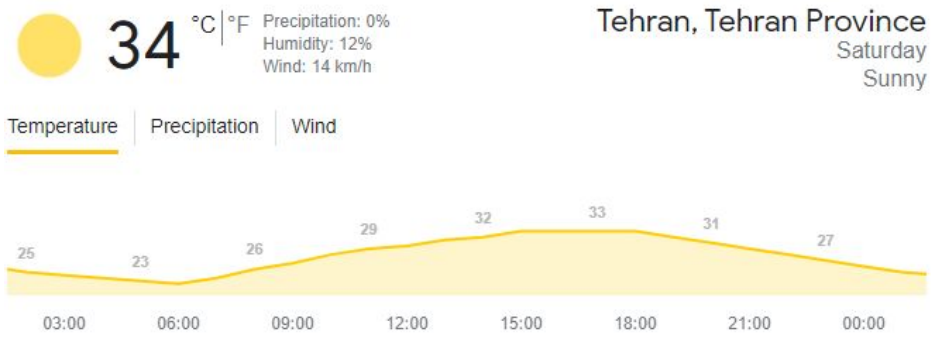
- Definition:

  An ordered sequence of values of a variable at equally spaced time intervals.

- Goal:
  - Obtain an understanding of the underlying structure
  - Fit a model and proceed to forecasting
  - Controlling future events via intervention

Data like

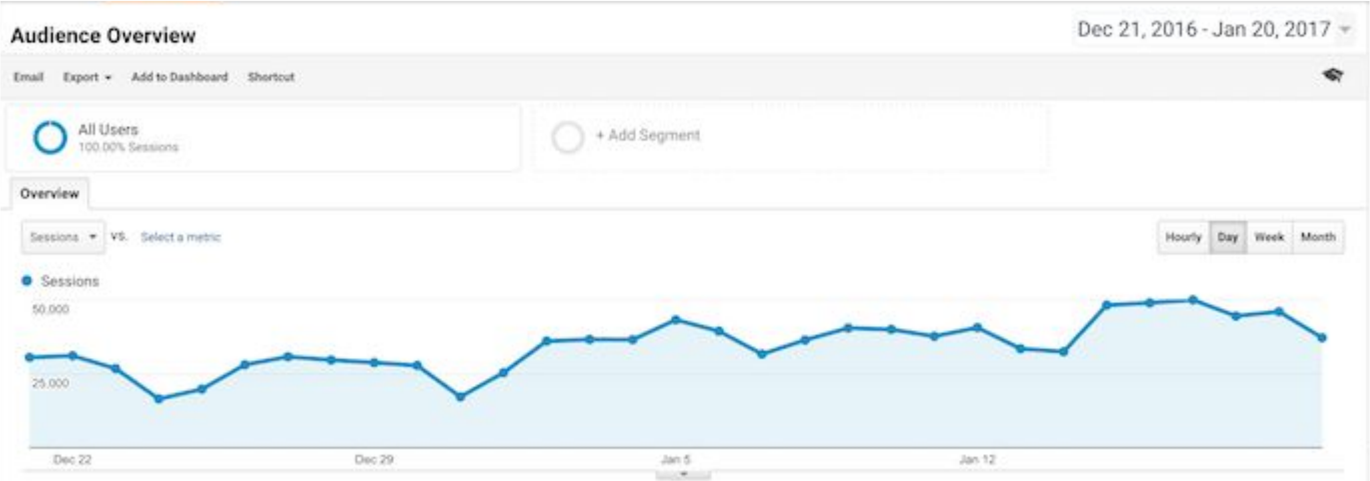
- Temperature



Time	Temperature
5:00 am	59 °F
6:00 am	59 °F
7:00 am	58 °F
8:00 am	58 °F
9:00 am	60 °F
10:00 am	62 °F
11:00 am	64 °F
12:00 pm	66 °F
1:00 pm	67 °F
2:00 pm	69 °F
3:00 pm	71 °F
4:00 pm	71 °F
5:00 pm	71 °F
6:00 pm	69 °F

Data like

- Google Analytics

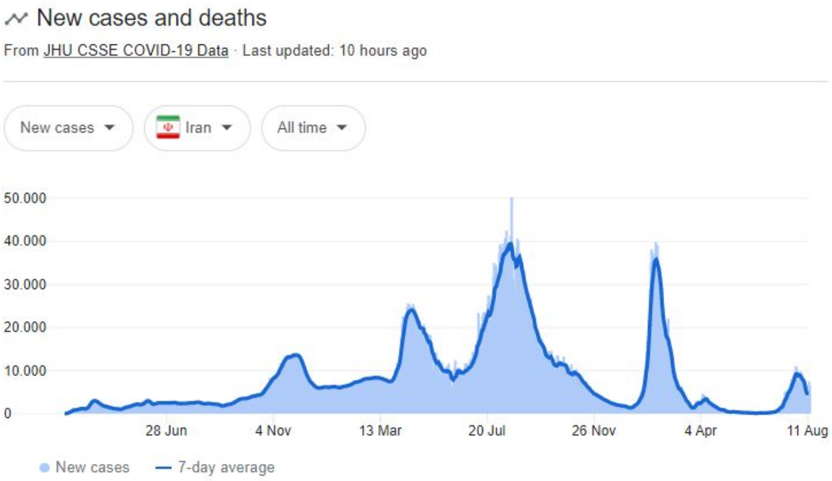


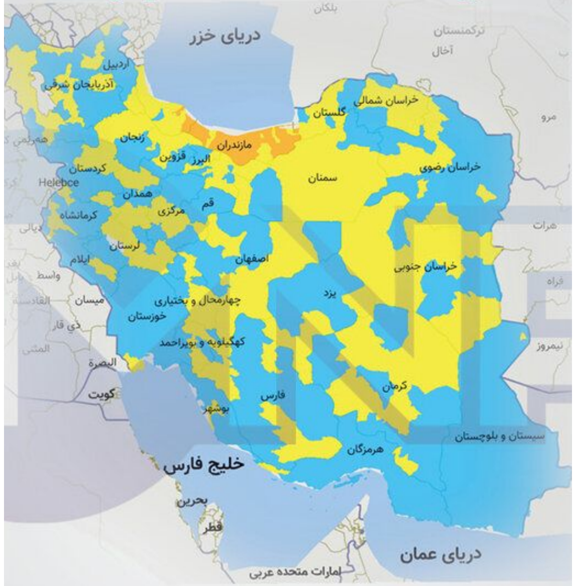
Data like

- Finance Time Series

Data like

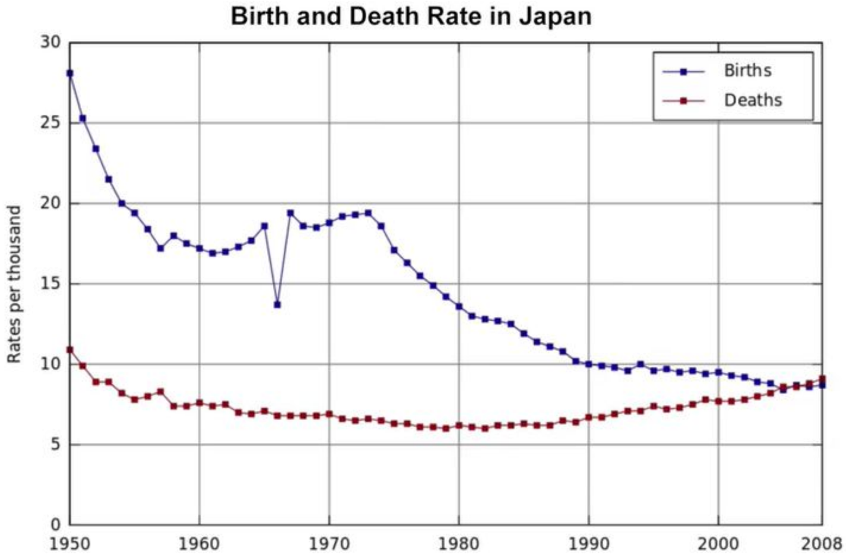
- Corona Stats

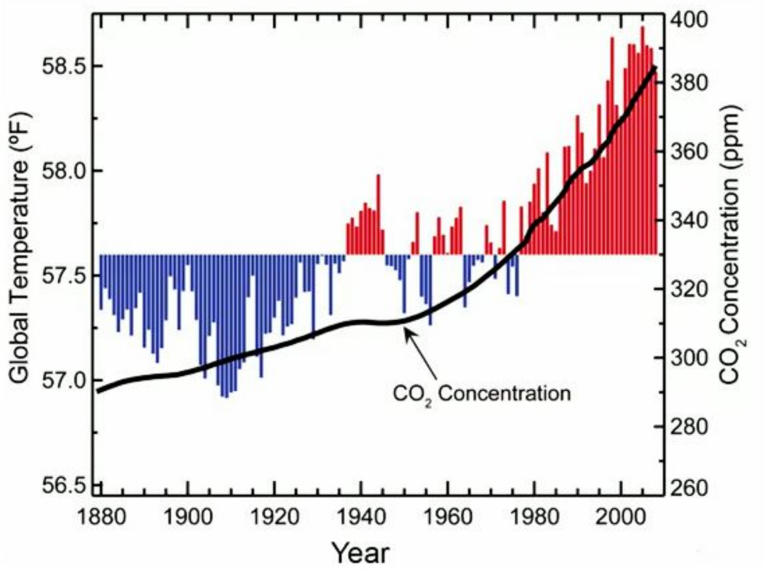




Data like

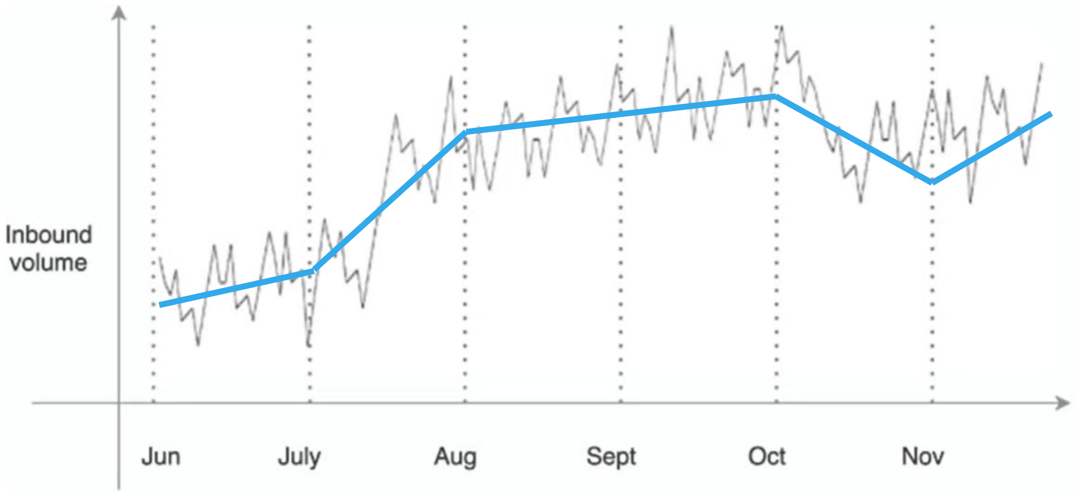
- Multivariate Time Series





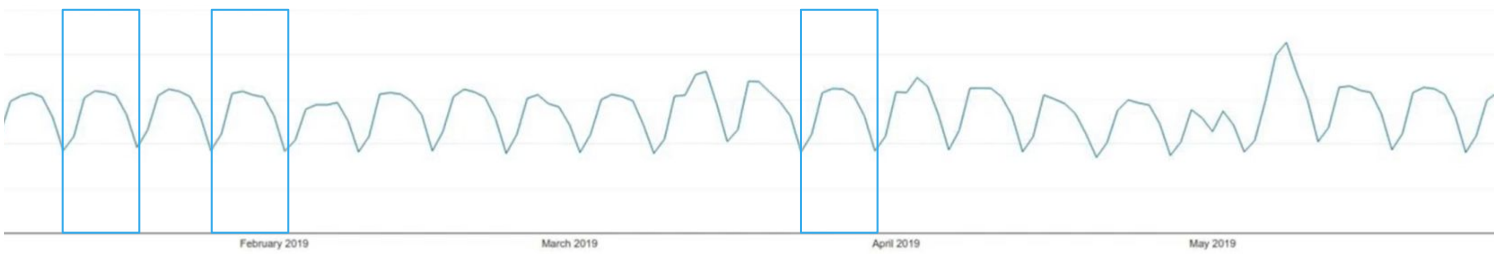
Common Patterns in Time Series

- Trend



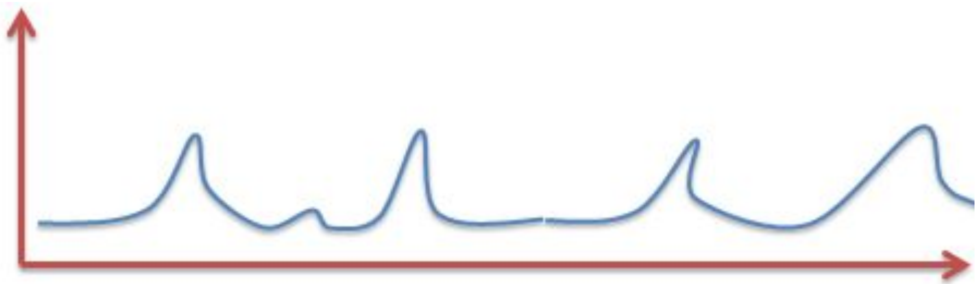
Common Patterns

- Seasonality



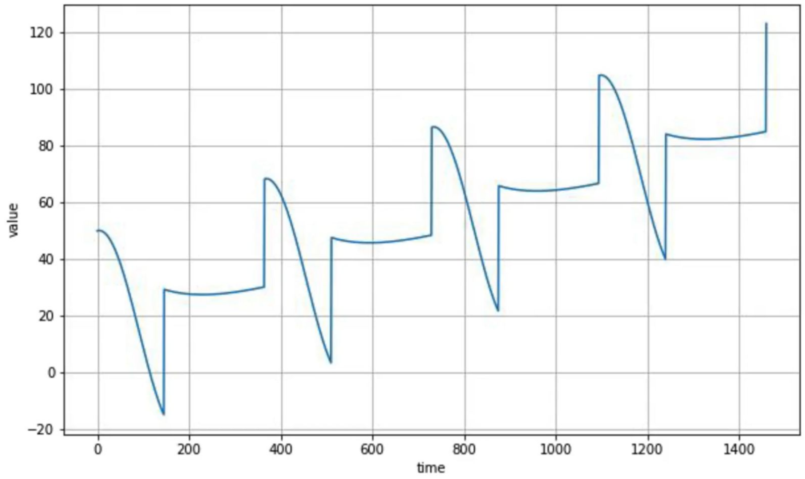
Common Patterns

- Cyclical



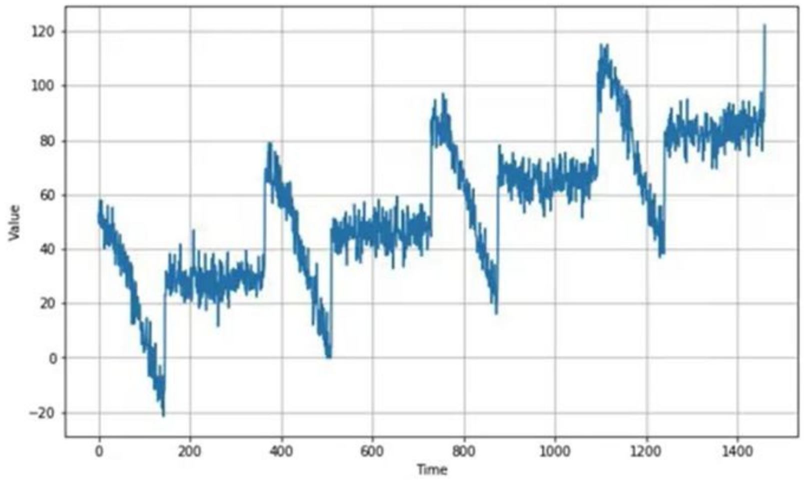
Common Patterns

- Trend + Seasonality



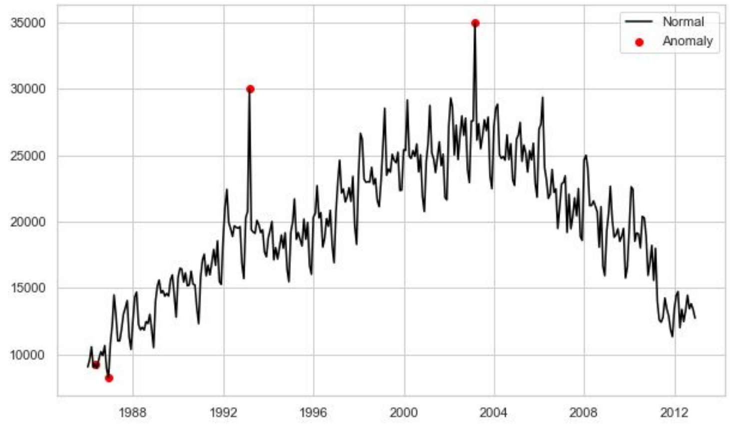
Common Patterns

- Trend + Seasonality + Noise

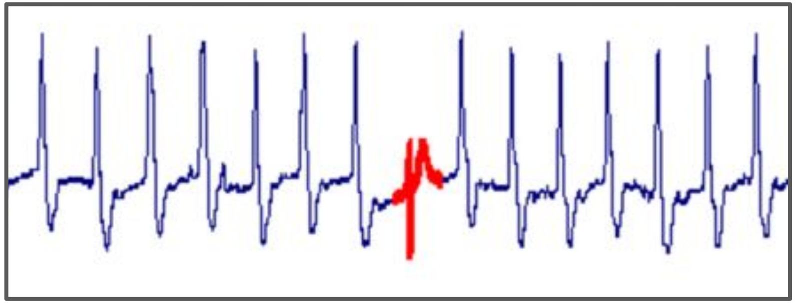


Common Patterns

- Anomaly



Pointwise anomaly

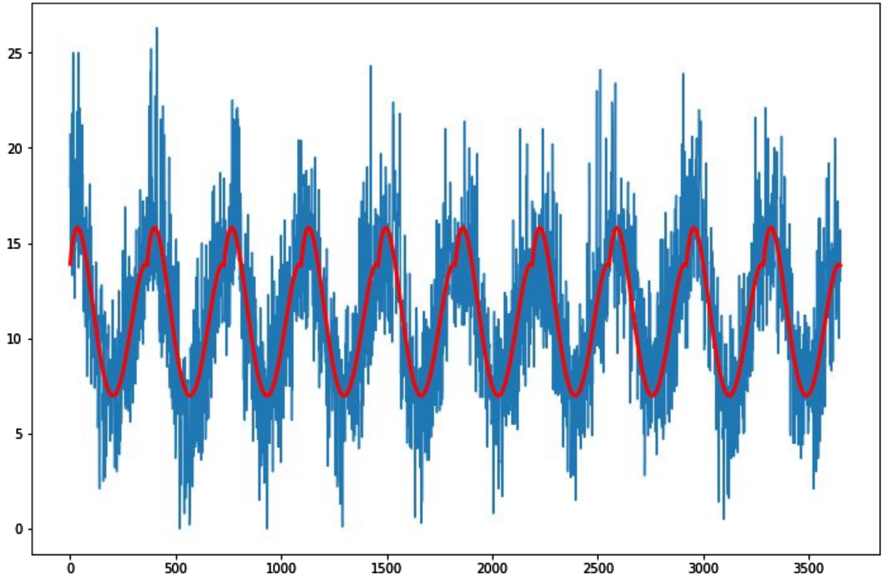


Collective anomaly

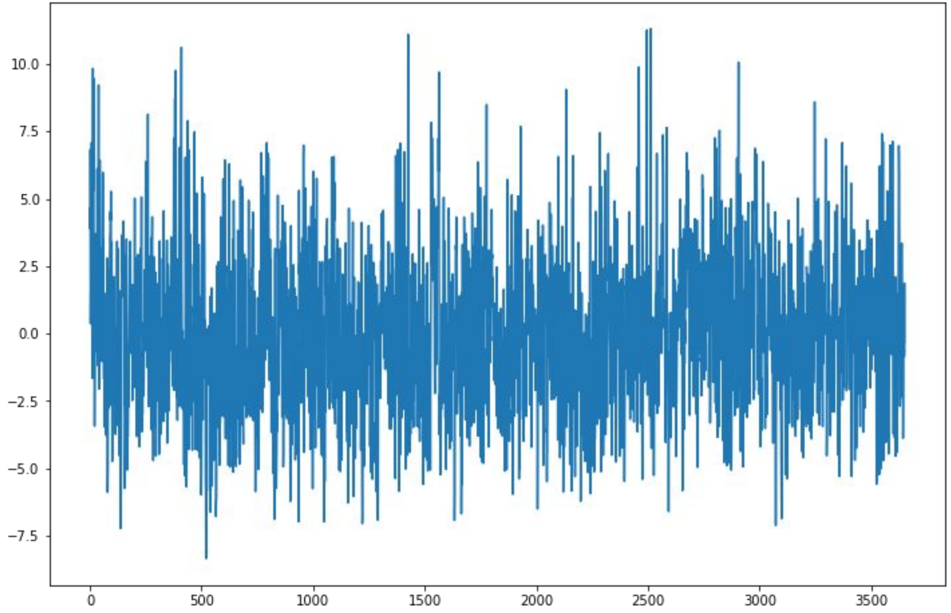
Removing Seasonality

We can model the seasonal component directly, then subtract it from the observations.

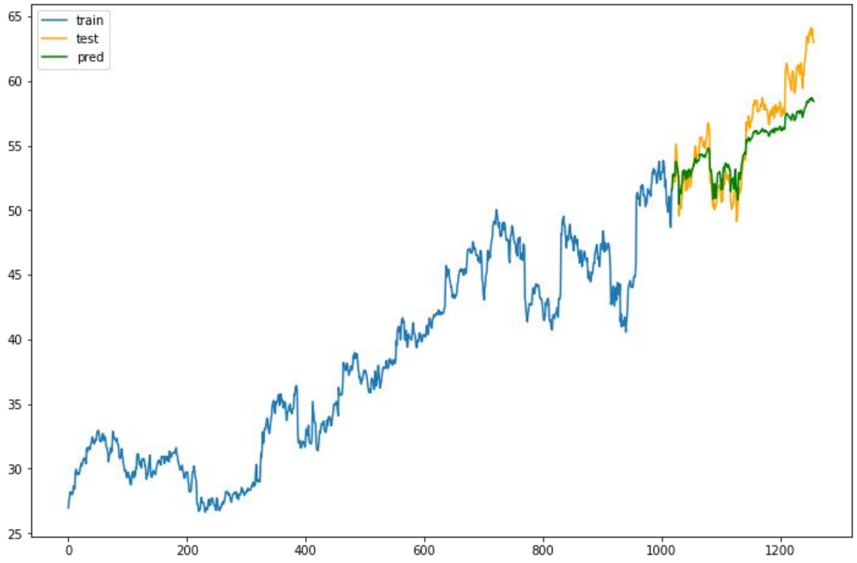
```
from numpy import polyfit

X = np.array([i%365 for i in range(0, len(series))])
y = series

degree = 4
coef = polyfit(X, y, degree)

curve = list()
for i in range(len(X)):
  value = coef[-1]
  for d in range(degree):
    value += X[i]**(degree-d) * coef[d]
  curve.append(value)
```

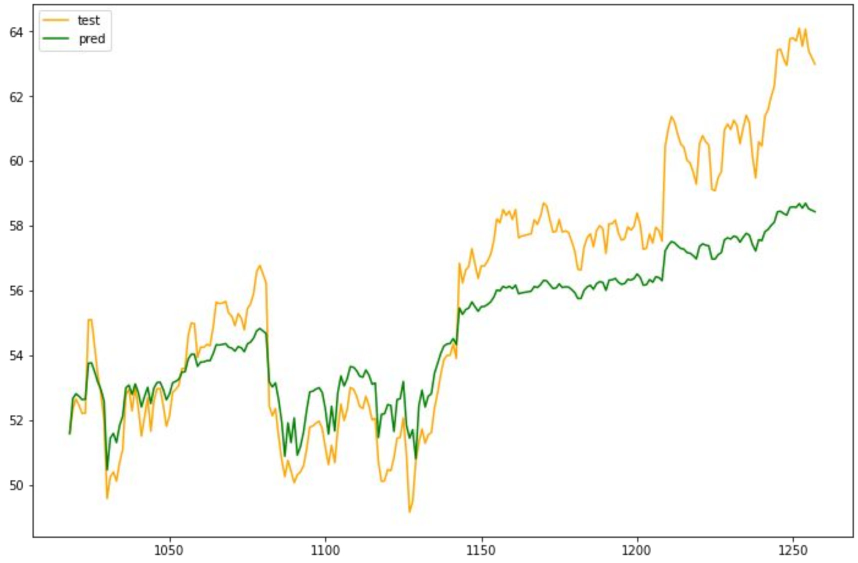


Removing Seasonality



How good is a model?





Evaluation Metrics

- Mean Squared Error (MSE)

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

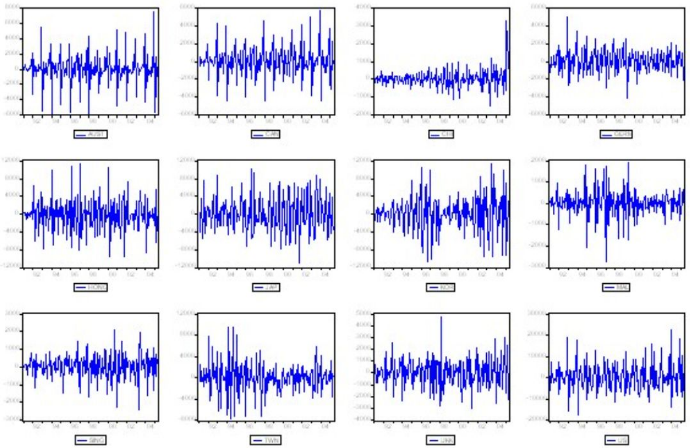
- Root Mean Square (RMS)

```python
from sklearn.metrics import mean_squared_error
from math import sqrt

rms = sqrt(mean_squared_error(preds,test))
```

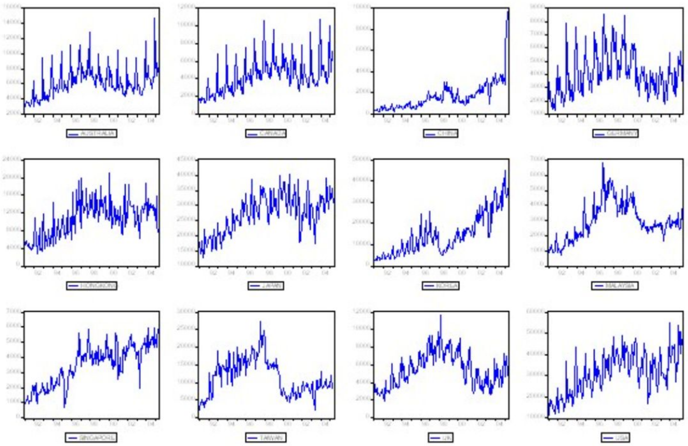
Key Assumption: Stationarity

- Using non-stationary time series data produces unreliable and spurious results and leads to poor understanding and forecasting
- Stationary means statistical properties (mean, variance, and covariance) of the time series doesn’t change as time goes on.

Stationary vs Non-stationary



Stationary



Non-stationary

Methods to check Stationarity

- There are two Statistical hypothesis testing for checking stationary.
  - p-value >0.05 Fail to reject (H0)
  - p-value <= 0.05 Accept (H1)

- Augmented Dickey-Fuller (ADF) test:
  - Null Hypothesis (H0): series has unit root
  - Alternative Hypothesis: series is trend-stationary.

- Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test
  - They were intended to complement unit root tests, such as the ADF test.
  - To figure out a time series is stationary around a mean or linear trend, or is non-stationary due to a unit root

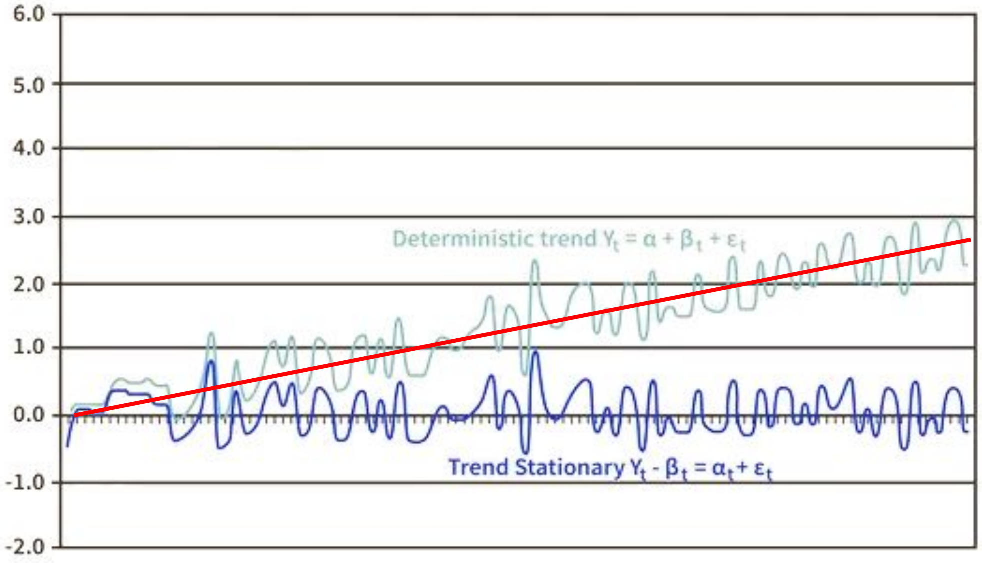
Converting Non-stationary into stationary

- Detrending
  - Linear Regression
  - Differencing

- Transformation
  - Log transfer
  - Square root
  - Box-Cox Transform

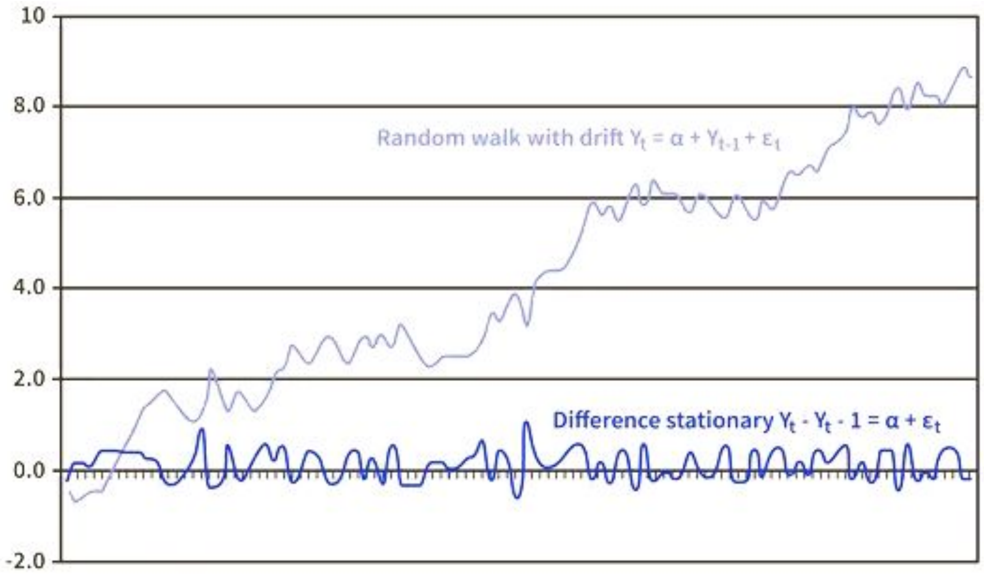
Detrending: by Linear Regression

```python
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X, y)

trend = model.predict(X)
```



Detrending: by Differencing

```
value(t) = observation(t) - observation(t-1)
```



Transformation

Transformation

- Box-Cox Transform

The resulting series may be more linear and the resulting distribution more Gaussian or Uniform, depending on the underlying process that generated it.

$$y_i^{(\lambda)} = \begin{cases} \dfrac{y_i^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0, \\ \ln y_i & \text{if } \lambda = 0, \end{cases}$$

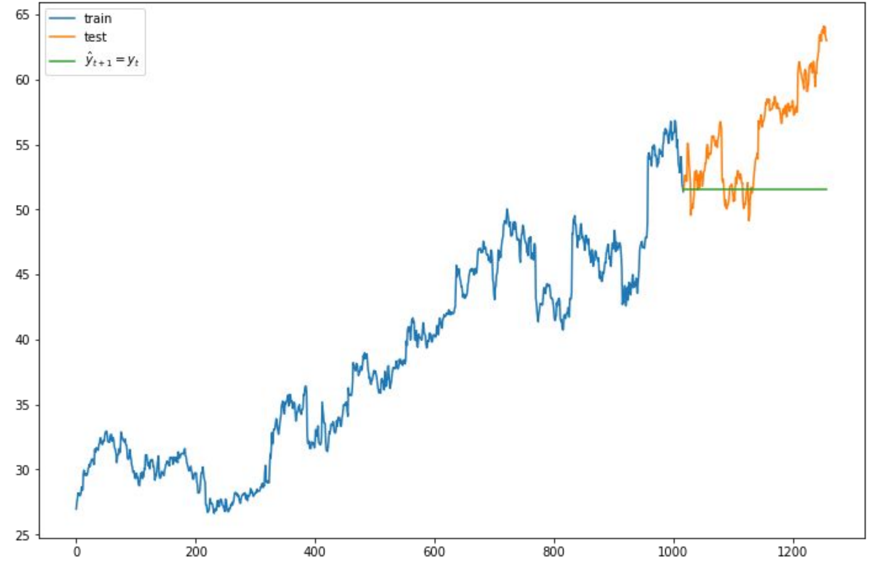
λ	Transformed Data
-2	y^{-2}
-1	y^{-1}
-0.5	$1/\sqrt{y}$
0	ln(y)
0.5	$\sqrt{y}$
1	y
2	y^2

```
from scipy.stats import boxcox

new_series = boxcox(series, lmbda=0.0)
```

Some Statistical methods

- Naive Approach

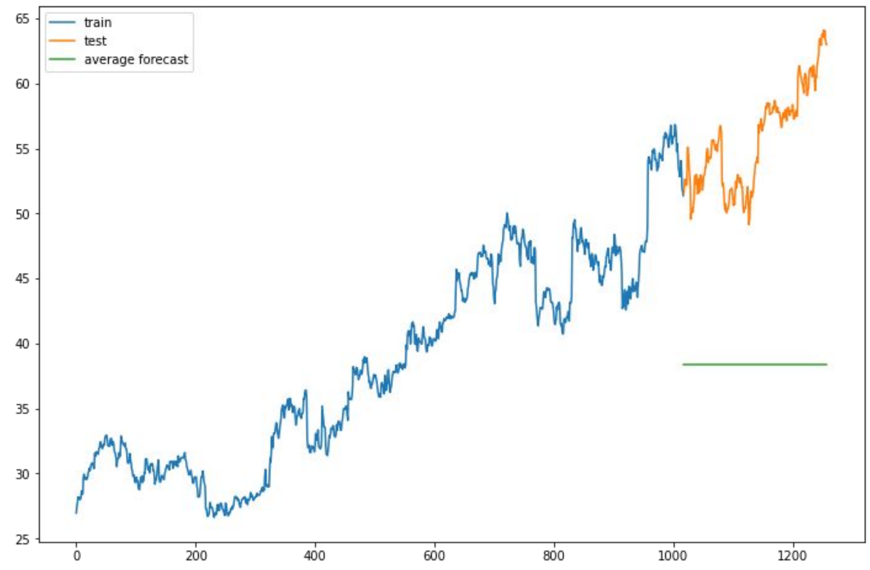
RMS = 5.696



Some Statistical methods

- Simple Average

RMS = 17.841



Some Statistical methods

- Moving Average

RMS = 3.934

Some Statistical methods

- Weighted Moving Average

$$\hat{y}_t = \frac{1}{T}(w_1 y_{t-1} + w_2 y_{t-2} + \ldots + w_T y_{t-T})$$

Some Statistical methods

- Simple Exponential Smoothing

$$\hat{y}_{t+1} = \alpha y_t + \alpha(1-\alpha)y_{t-1} + \alpha(1-\alpha)^2 y_{t-1} + \ldots$$

```
from statsmodels.tsa.api import SimpleExpSmoothing

fit2 = SimpleExpSmoothing(train).fit(smoothing_level = 0.7,optimized = False)
pred = fit2.forecast()
```

Analysis Methods

Outline: Second half

- Introduction
- White noise
- Autocorrelation and Partial Autocorrelation
- Parametric Linear Models
- Parametric Nonlinear Models
- Nonparametric Models
- Spectral Analysis
- Deep Neural Network Models

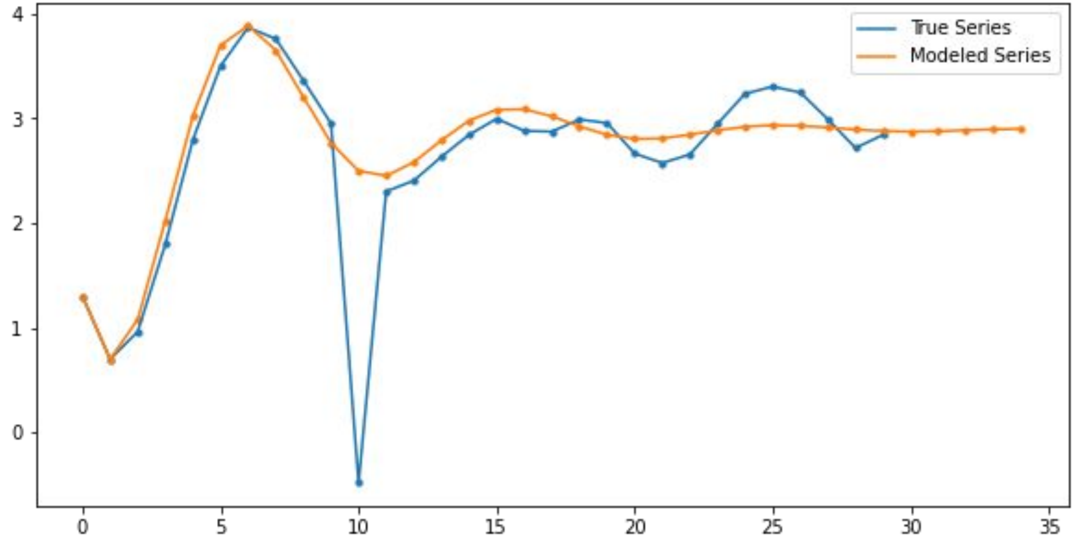
Introduction

What is a time series analysis model?

Why we create a time series model?

Forecasting

Anomaly detection



White Noise

Process $\{\epsilon_t\}_t$ is called white noise if:

1. $\forall t : \mathbb{E}[\epsilon_t] = 0$
2. $\forall t : Var(\epsilon_t) = \sigma^2$
3. $\forall t \neq s : Cov(\epsilon_t, \epsilon_s) = 0$

A white noise series is a realization (sample) of white noise process.

Autocorrelation and Partial Autocorrelation

Autocorrelation:

Correlation of the signal with its lagged (shifted) version.

$$\rho(k) = \frac{\frac{1}{n-k}\sum_{t=k+1}^{n}(x_t - \bar{x})(x_{t-k} - \bar{x})}{\sqrt{\frac{1}{n}\sum_{t=1}^{n}(x_t - \bar{x})^2}\sqrt{\frac{1}{n-k}\sum_{t=k+1}^{n}(x_{t-k} - \bar{x})^2}}$$

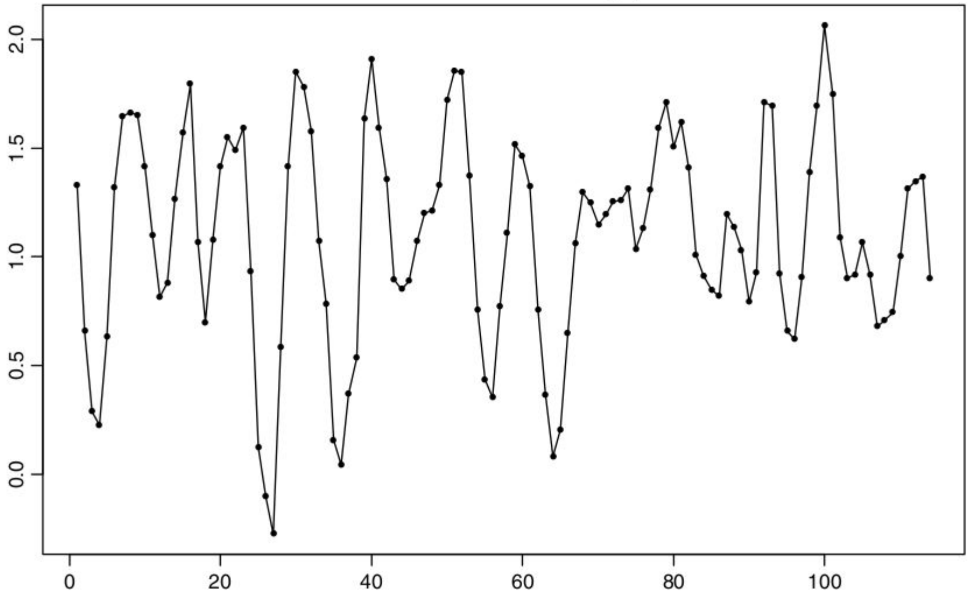
Partial Autocorrelation:

Autocorrelation of the signal with k-shifted version of signal after removing linear independence of signal points with previous k-1 points.

Linear Parametric: Autoregressive Model (AR)

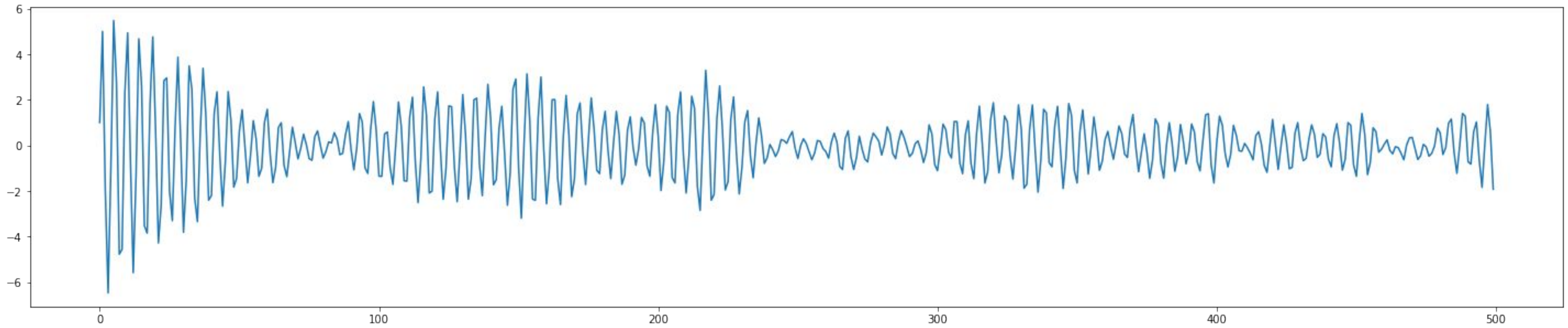
$$X_t = b_1 X_{t-1} + b_2 X_{t_2} + \ldots + b_p X_{t-p} + \epsilon_t$$
$$\epsilon_t \sim \mathcal{N}(\mu, \sigma^2)$$

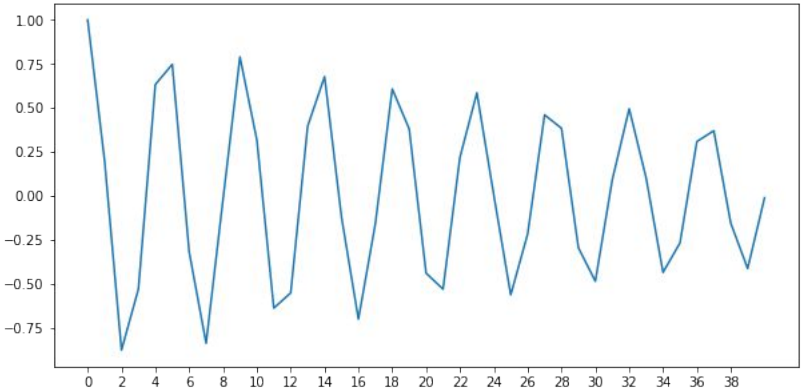
$$X_t = 1.35 X_{t-1} - 0.72 X_{t-2} + \epsilon$$
$$\epsilon \sim \mathcal{N}(1.07, 0.24^2)$$

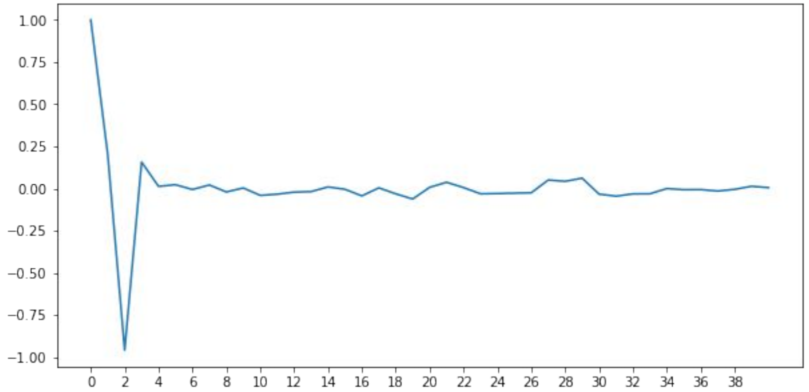


AR Model: Example

$$X_t = 0.75X_{t-1} - 1.1X_{t_2} + 0.35X_{t-3} + \epsilon_t \qquad \epsilon_t \sim \mathcal{N}(0, 0.3^2)$$







Linear Parametric: Moving Average Model (MA)

$$X_t = \epsilon_t + a_1 \epsilon_{t-1} + a_2 \epsilon_{t-2} + \ldots + a_p \epsilon_{t-p}$$

$\{\epsilon_t\}$ is white noise

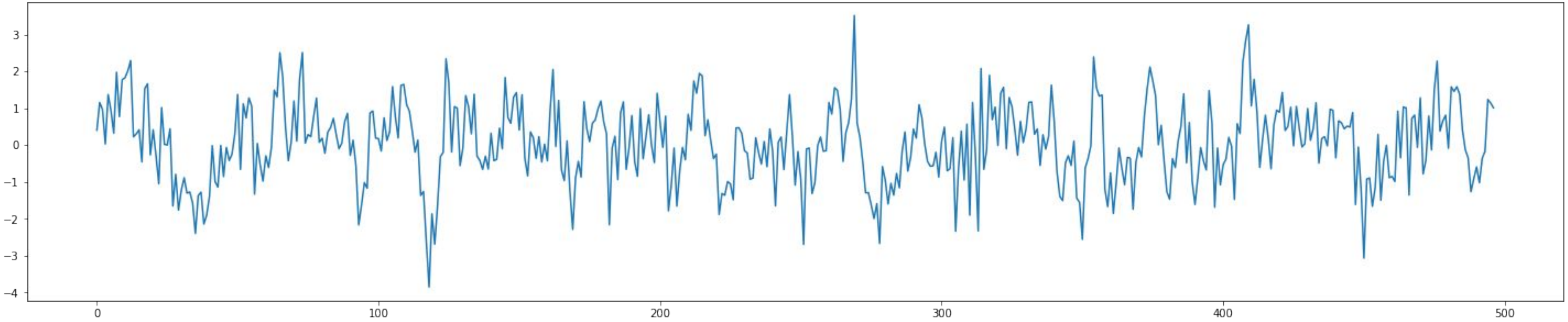
1. For h > p:

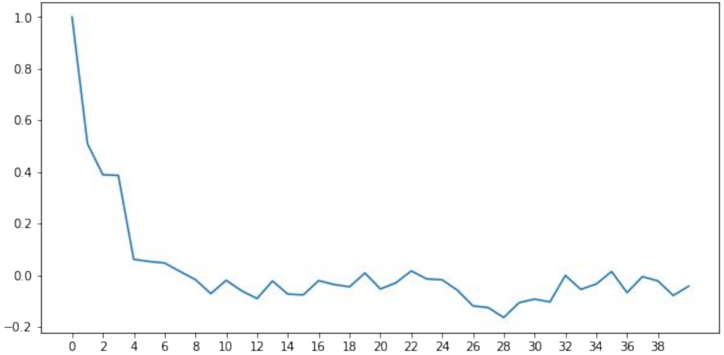
   X_t uncorrelated of X_(t-h)
2. Straightforward first and second order statistics.
3. Hard to implement

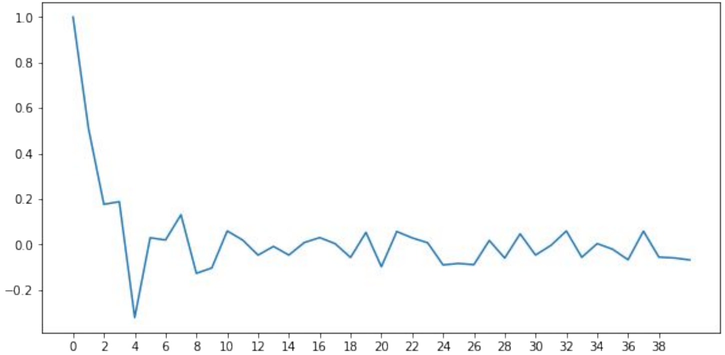
MA Model: Example

$$X_t = \epsilon_t + 0.7 \times \epsilon_{t-1} + 0.9 \times \epsilon_{t-2} + 1.7 \times \epsilon_{t-3}$$

$\{\epsilon_n\}_n$ is white noise







Linear Parametric: ARIMA Model (AR + I + MA)

ARIMA(p, d, q): Following model on the data detrended by differencing:

$$X_t = b_1 X_{t-1} + b_2 X_{t-2} + \ldots + b_p X_{t-p} + \epsilon_t + a_1 \epsilon_{t-1} + a_2 \epsilon_{t-2} + \ldots + a_q \epsilon_{t-q}$$

$$\epsilon_t \sim N(\mu, \sigma^2)$$

$\{\epsilon_t\}$ is white noise

ARIMA Model Algorithm

1. Remove trend by differencing
2. Check for autocorrelation of different orders -> MA model
3. Check for partial autocorrelation of different orders -> AR model
4. Add any of AR or MA if applicable
5. Solve the model

Nonlinear Parametric Models

- Threshold Autoregressive Model (TAR):

$$X_t = \sum_{i=1}^{m} (b_{i0} + b_{i1} X_{t-1} + b_{i2} X_{t-2} + \ldots + b_{ip} X_{t-p} + \sigma_i \epsilon_t) I(X_{t-d} \in A_i)$$

- Autoregressive Conditional Heteroscedastic Model (ARCH):

$$X_t = \sigma_t \epsilon_t \quad : \quad \sigma_t^2 = b_0 + b_1 X_{t-1}^2 + b_2 X_{t-2}^2 + \ldots + b_p X_{t-p}^2$$

Nonparametric Models

No parameters are learned. Instead function values are estimated for each real input.

Functional-Coefficient Autoregressive Model (FAR):

$$X_t = a_1(X_{t-d})X_{t-1} + a_2(X_{t-d})X_{t-2} + \ldots + a_p(X_{t-d})X_{t-p} + \sigma(X_{t-d})\epsilon_t$$

Extension of TAR model.

Additive Autoregressive Model (AAR):

$$X_t = f_1(X_{t-1}) + \ldots + f_p(X_{t-p}) + \epsilon_t$$

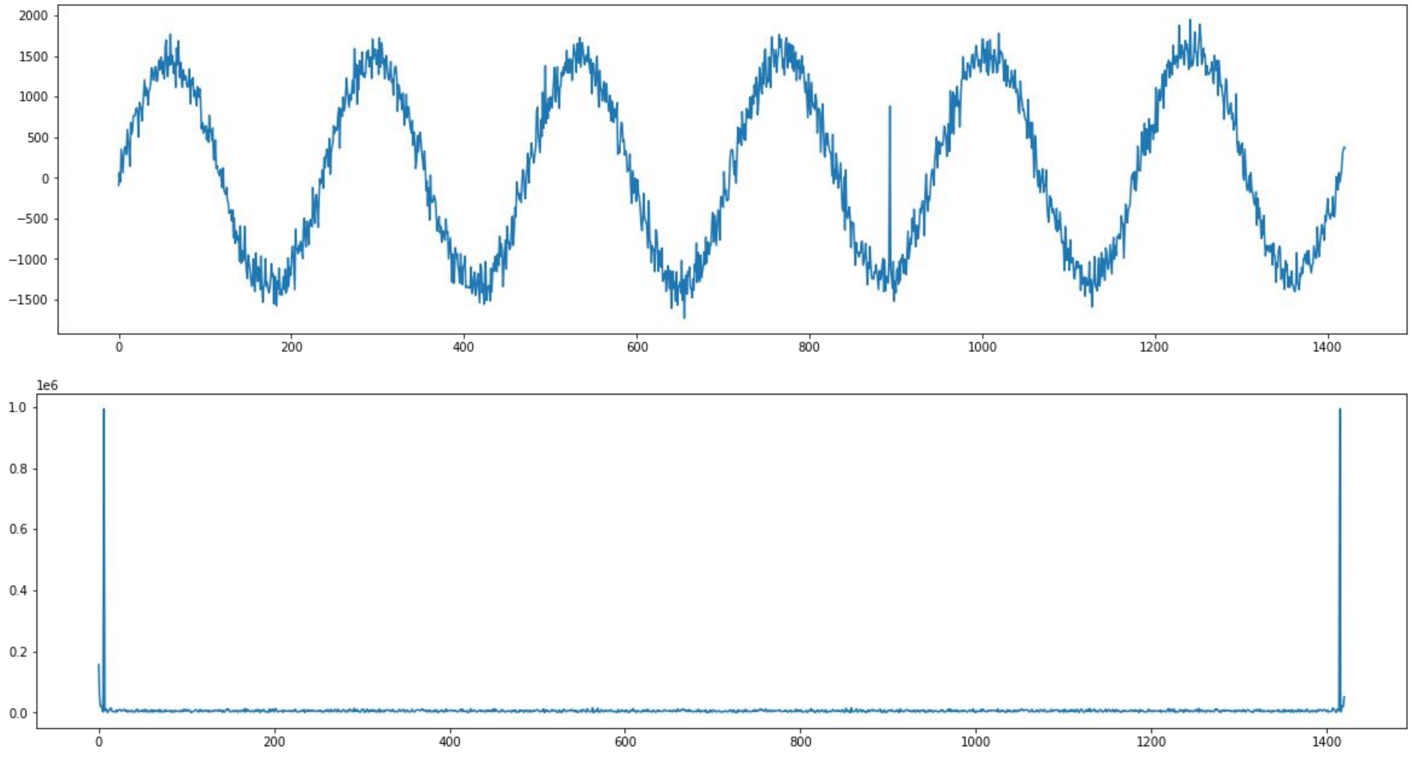
Spectral Analysis

For many applications signals show dominant features in frequency domain that are not apparent in time domain.

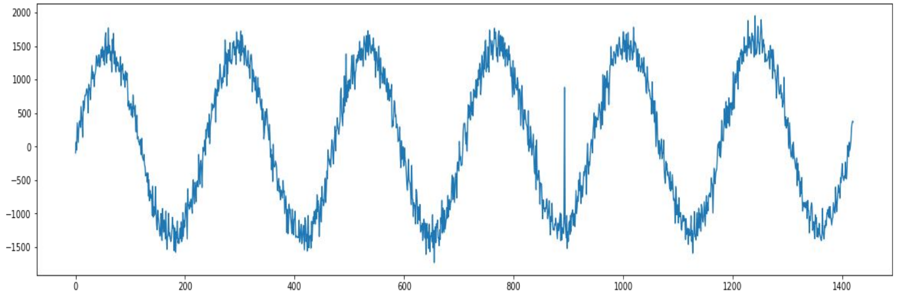
Periodic signals

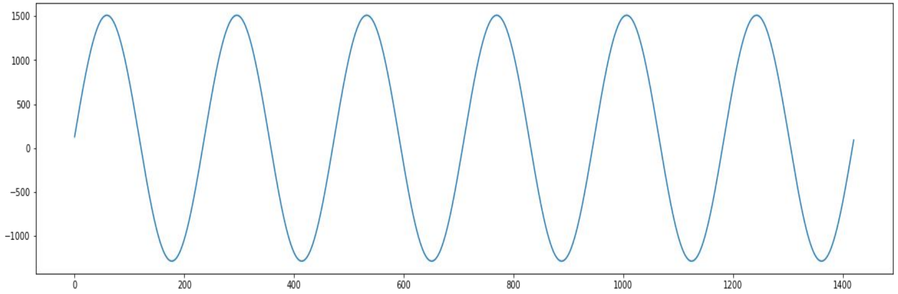
Observing signal in Frequency domain can help decompose signal into simple Sine functions and a residual series.

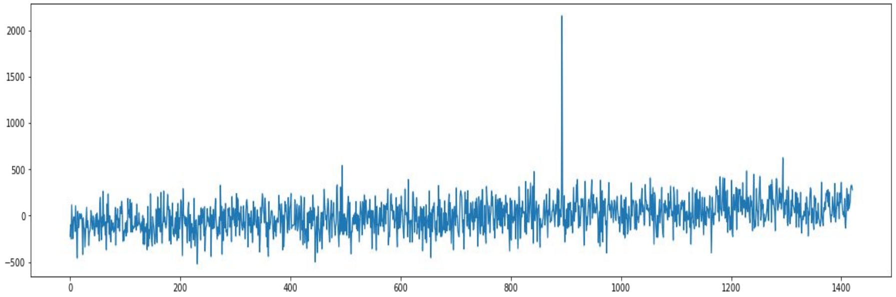
Spectral Analysis



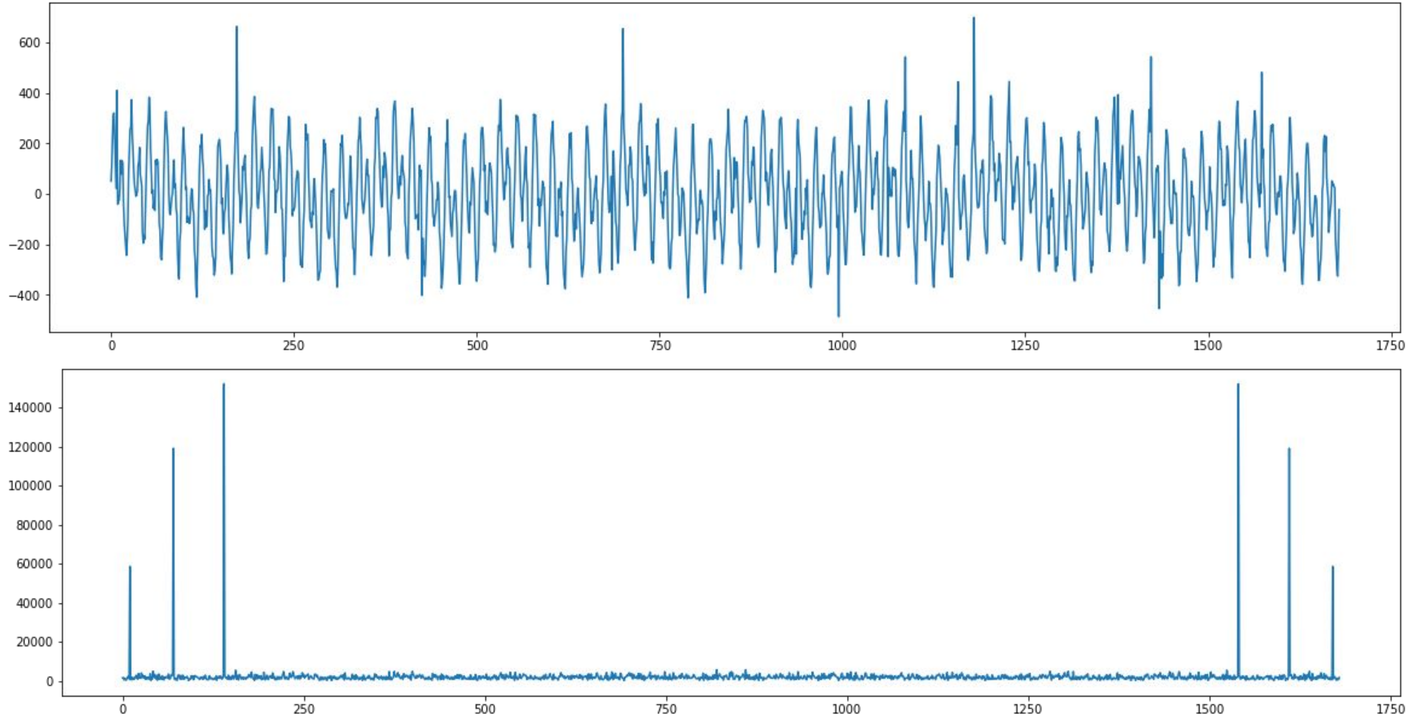
Spectral Analysis



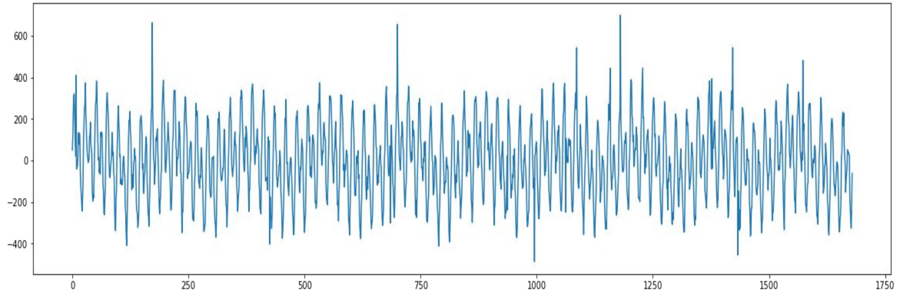


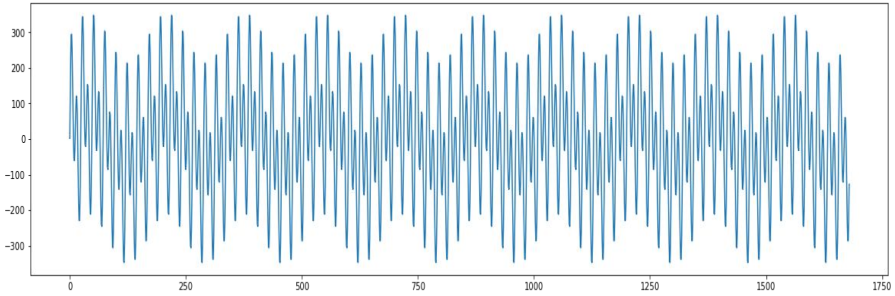


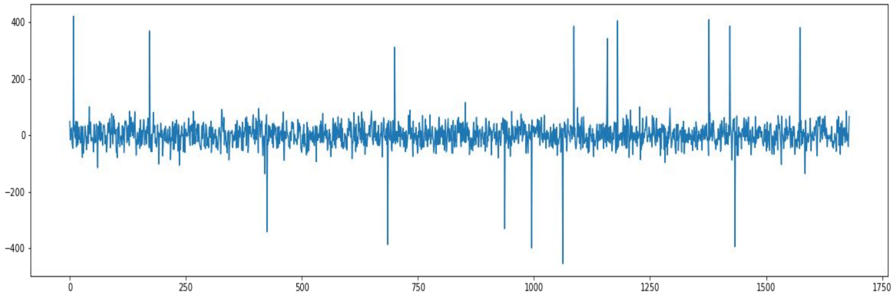
Spectral Analysis



Spectral Analysis

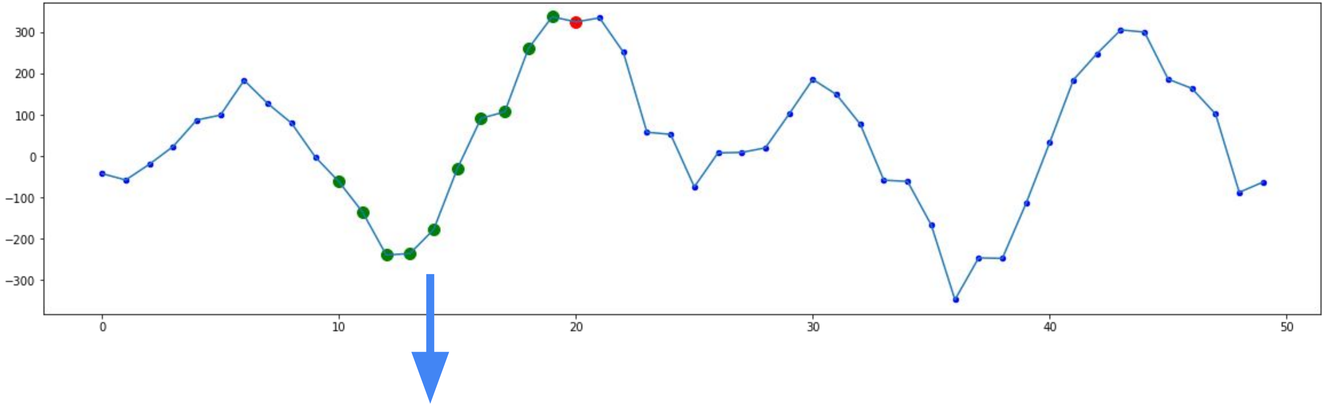






Deep Neural Network Models

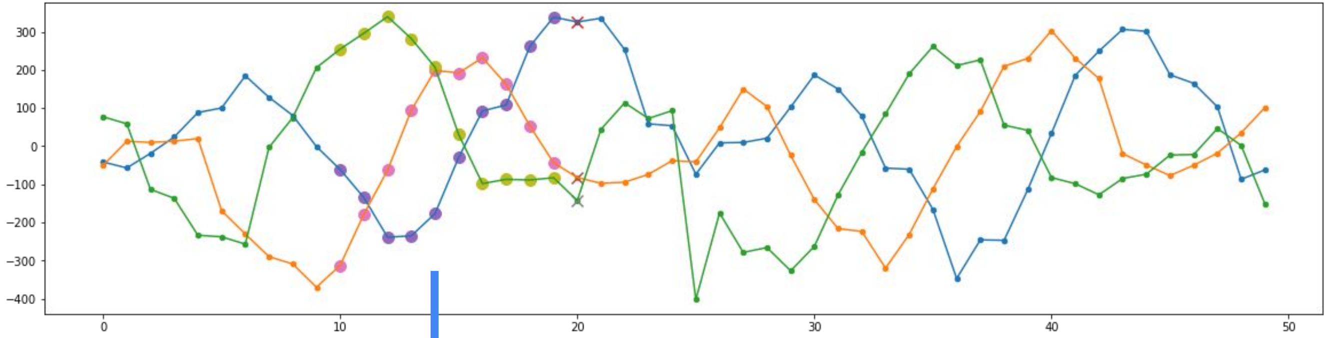
MLP (Multi-Layer Perceptron) Model: Multi-layer fully-connected neural network



[-61.1 ,-135.0 ,-238.8 ,-235.2 ,-176.9 , -29.1 , 92.3 , 108.0 , 261.5 , 338.5] , 335.6

Deep Neural Network Models

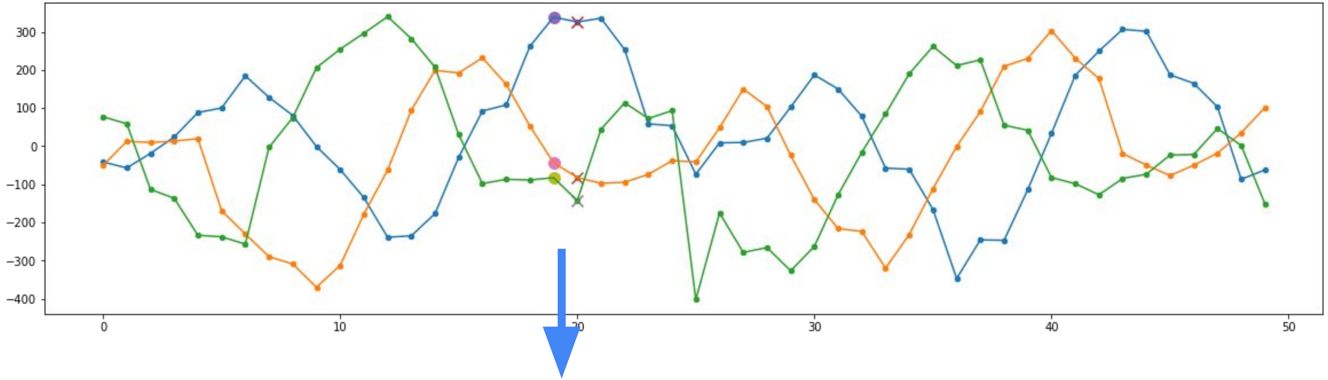
CNN (Convolutional Neural Network) Model



[-61.1	-135.0	-238.8	-235.2	-176.9	-29.1	92.3	108.0	261.5	338.5]	325.8
[-312.7	-178.5	-62.8	94.2	198.7	192.0	231.8	162.4	52.9	-43.3]	-82.3
[254.5	296.2	339.6	282.1	207.2	30.0	-98.2	-86.9	-88.7	-82.7]	-142.7

Deep Neural Network Models

LSTM (Long short-term memory) Model



[338.5] 325.8

[-43.3] -82.3

[-82.7] -142.7

DNN Models: Comparison

MLP:

1. Easy to implement
2. Fast training
3. Mostly used for univariate signals
4. Only short-time relations can be modeled efficiently

CNN:

1. Fast training
2. Efficient number of parameters (allows longer relations)
3. Used both for univariate and multivariate signals
4. Considers locality

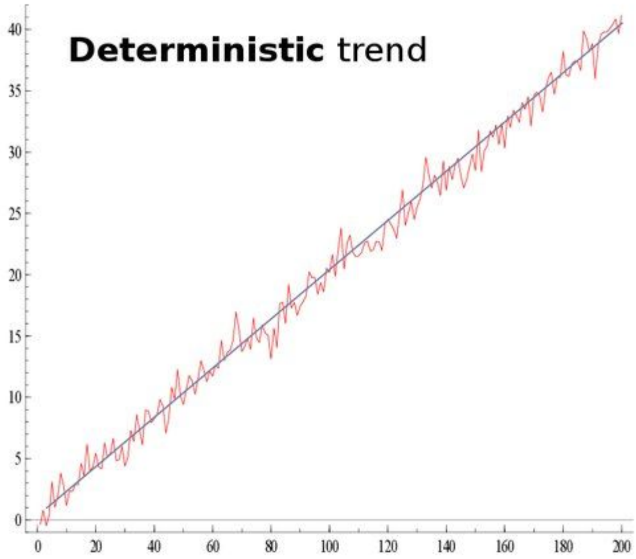
DNN Models: Comparison

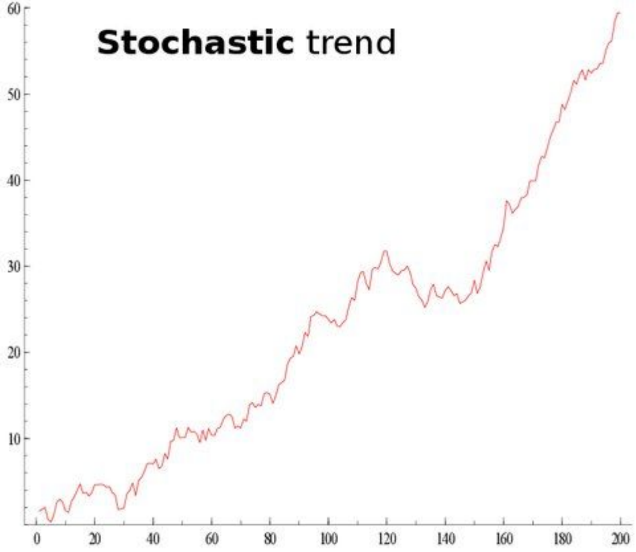
LSTM:

1. Slow training
2. Used for multivariate signals
3. Used for single-variate signals with additional features
4. Models long-time relations (theoretically, arbitrary history length)
5. Can model more complex relations/structures
6. Fast inference time

Any Question?

What is trend deterministic?





What is root of a series?!

Corresponds to ADF test, Unit Root

an autoregressive process of order p: $y_t = a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_p y_{t-p} + \varepsilon_t.$

characteristic equation: $m^p - m^{p-1} a_1 - m^{p-2} a_2 - \cdots - a_p = 0$

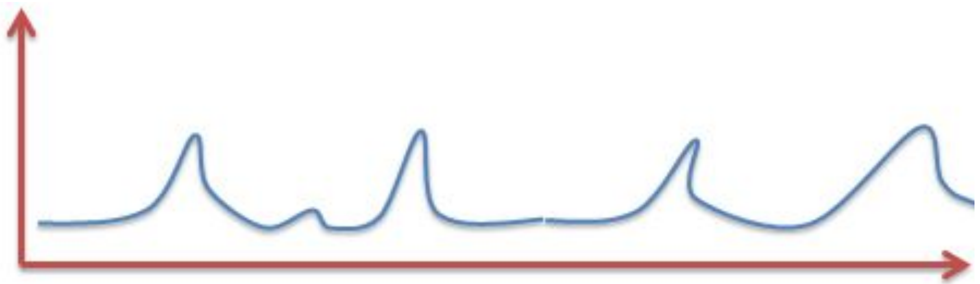
The difference between ADF test and p-value?

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \delta_1 \Delta y_{t-1} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \varepsilon_t,$$

- Null hypothesis: $\gamma = 0$
- Alternative hypothesis: $\gamma < 0.$

- A value for the test statistic : $\mathrm{DF}_\tau = \frac{\hat{\gamma}}{\mathrm{SE}(\hat{\gamma})}$

What about Uncertain periods?



Why are we changing our series?