Time Series Forecasting - 101

PURDUE UNIVERSITY

WHAT IS A TIME SERIES?

- A time series is a series of data points indexed in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time.
- TS data are collected and used in every type of businesses.
- Example of time series:
- Monthly sales
  Hourly stock closing prices
  Quarterly unemployment rate
  Annual GDP
  Daily airline filled seats

OUR DATASET

	Month	Air RPM (000s)	Rail PM	VMT (billions)
0	Jan-90	35153577	454115779	163.28
1	Feb-90	32965187	435086002	153.25
2	Mar-90	39993913	568289732	178.42
3	Apr-90	37981886	568101697	178.68
4	May-90	38419672	539628385	188.88
...	...	...	...	...
167	Dec-03	57795908	489403554	237.60
168	Jan-04	53447972	410338691	217.30
169	Feb-04	52608801	389778365	210.40
170	Mar-04	63600019	453014590	247.50
171	Apr-04	61887720	471116666	245.40

- Number of miles travelled by air, rail and road since 1990 January
- Type of data – monthly

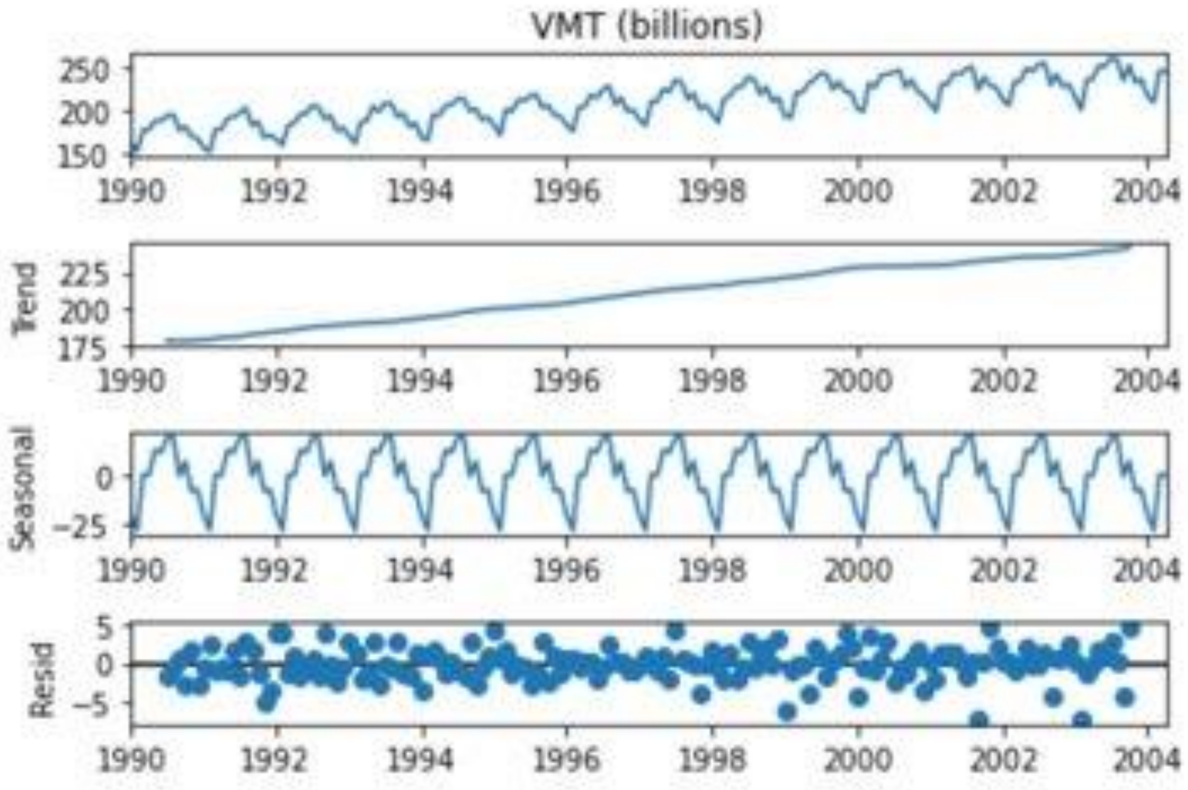
Pre-processing steps

- Remember, time-series data must have:
- an index with equal increments.
- must be of datetime type, when working with Python
- Daterange is a pandas function that is handy in redefining time series indexes.
- Check pandas.date_range — pandas 1.5.3 documentation (pydata.org)
- Here, we perform the following:

```
# Always convert the time column to a datetime format
df['Month'] = pd.date_range(start='1990/01/01', end='2004/04/01',freq='MS')
```

Time series components

A time series can be decomposed into the following components:



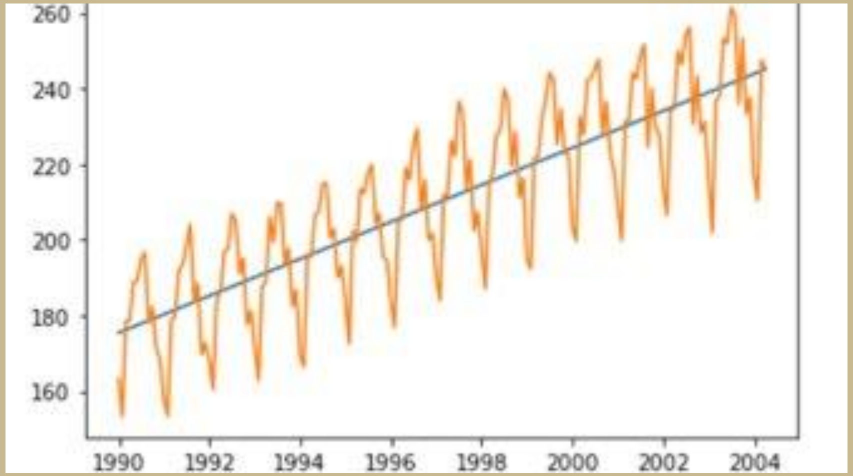
Level – the mean of all points

PURDUE UNIVERSITY

SIMPLE LINEAR REGRESSION FOR TIME SERIES

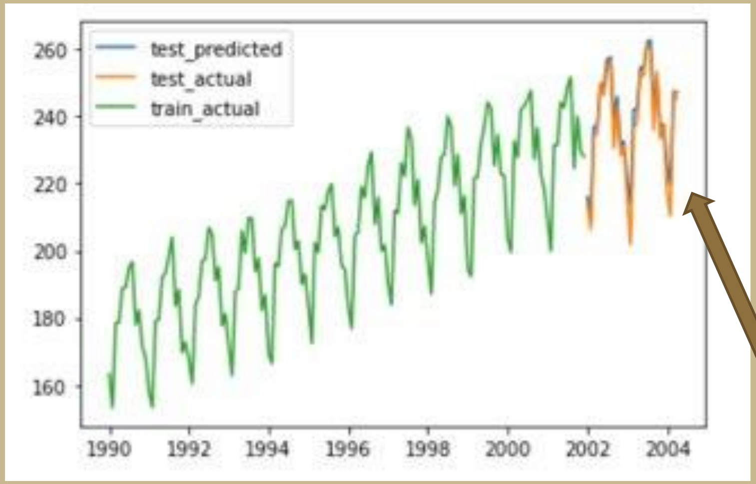
- Target variable – y(t)
- Predictor – indexed t (1,2,3....t)
- Y(t) is expected to function the trendline
- Evaluation of predictors, model like traditional linear regression
- Con – Does not capture seasonality

Dep. Variable:	VMT (billions)	R-squared:	0.662
Model:	OLS	Adj. R-squared:	0.660
Method:	Least Squares	F-statistic:	333.3
Date:	Fri, 10 Feb 2023	Prob (F-statistic):	6.46e-42
Time:	22:13:24	Log-Likelihood:	-703.55
No. Observations:	172	AIC:	1411.
Df Residuals:	170	BIC:	1417.
Df Model:	1		
Covariance Type:	nonrobust		



MULTIPLE LINEAR REGRESSION FOR TIME SERIES

	coef	std err	t	P>|t|	[0.025	0.975]
const	154.0111	0.842	182.993	0.000	152.346	155.676
2	-7.3464	1.078	-6.812	0.000	-9.480	-5.213
3	20.2113	1.079	18.740	0.000	18.078	22.345
4	19.7548	1.079	18.315	0.000	17.621	21.889
5	31.9059	1.079	29.579	0.000	29.772	34.040
6	30.9345	1.079	28.675	0.000	28.800	33.069
7	38.3655	1.079	35.558	0.000	36.231	40.500
8	38.7308	1.079	35.891	0.000	36.596	40.865
9	19.7352	1.079	18.285	0.000	17.600	21.870
10	26.0437	1.080	24.125	0.000	23.908	28.179
11	11.0023	1.080	10.189	0.000	8.866	13.138
12	11.9242	1.080	11.040	0.000	9.788	14.061
t	0.4273	0.005	80.401	0.000	0.417	0.438



2 – 12 Represent seasons taking binary values (0 or 1).

- Target – y(t)
- Predictors – indexed t, seasons (dummied), sometimes other exogenous variables.
  - Exogenous variables are independent predictors.
  - For example, if we use number of bananas sold at time t-1 to predict number of apples at time t, number of bananas becomes an exogenous variable
- Captures seasonality

INTRODUCTION TO ACF AND PACF

Lags are time series n times removed. For example, lag 1 (y(t-1)) for a time series 1,2,3,4 would be NaN, 1, 2 , 3

What are lags?

```
[88]: df_rail = pd.DataFrame()
      df_rail['y(t)'] = df['Rail PM']
      df_rail['y(t-1)'] = [float('nan')]+list(df_
      df_rail['y(t-2)'] = [float('nan')]+list(df_
```

```
[89]: df_rail
```

t[89]:

Month	y(t)	y(t-1)	y(t-2)
1990-01-01	454115779	NaN	NaN
1990-02-01	435086002	454115779.0	NaN
1990-03-01	568289732	435086002.0	454115779.0
1990-04-01	568101697	568289732.0	435086002.0
1990-05-01	539628385	568101697.0	568289732.0

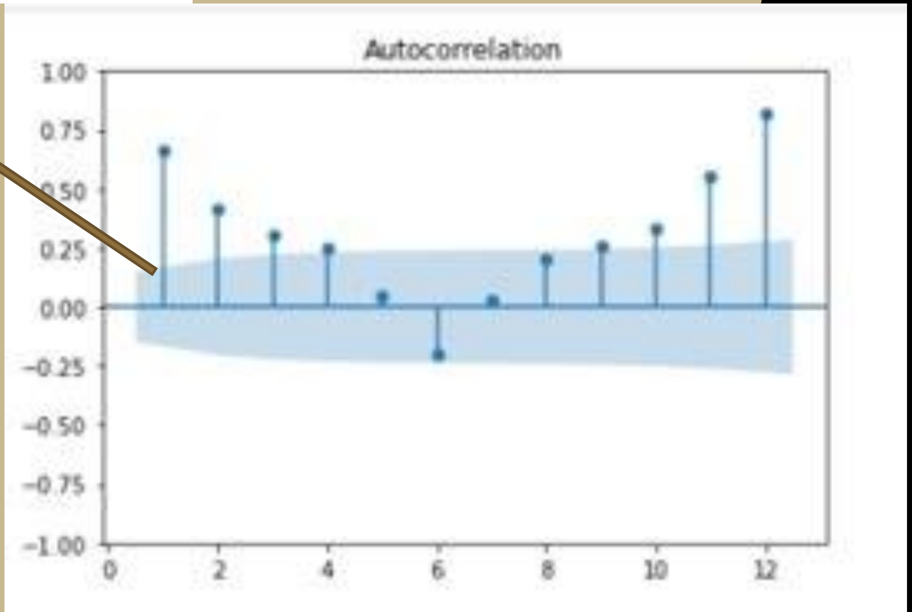
PURDUE UNIVERSITY®

ACF

Correlation between y(t) and its lags

```
In [90]: df_rail.corr()
Out[90]:
```

	y(t)	y(t-1)	y(t-2)
y(t)	1.000000	0.662715	0.411188
y(t-1)	0.662715	1.000000	0.662701
y(t-2)	0.411188	0.662701	1.000000



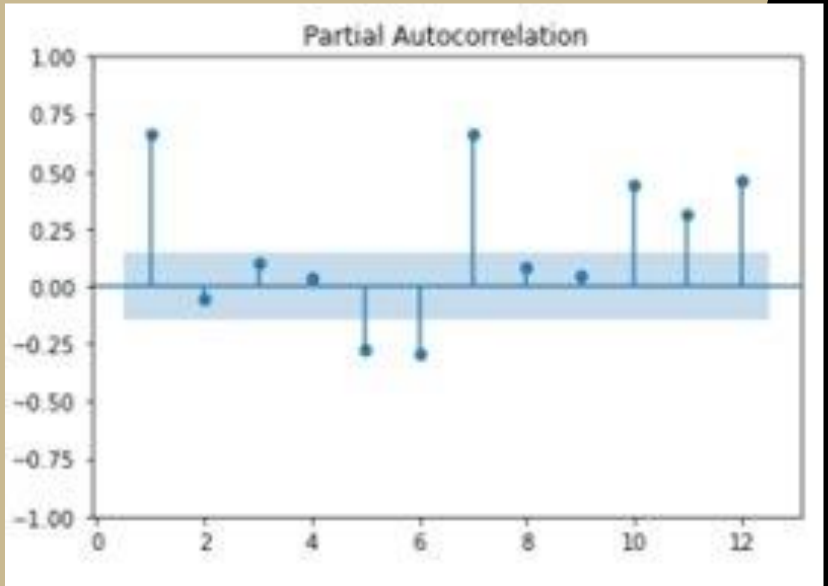
AC – Auto correlation, as its name suggests, represents correlation between lags. For example, it represents the correlation of the time series with itself in a way.

The ACF – autocorrelation function is a visual representation of the table above

PACF

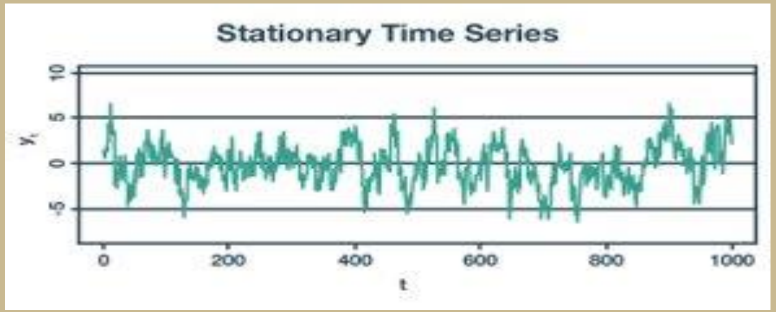
	y(t)	y(t-1)	y(t-2)
y(t)	1.000000	0.662715	0.411188
y(t-1)	0.662715	1.000000	0.662701
y(t-2)	0.411188	0.662701	1.000000

- Suppose the variable y(t) is correlated to both y(t-1) and y(t-2) (lag 1 and lag 2 series). This would mean that y(t-1) and y(t-2) would also be correlated. What if we want to know the true effect of y(t-2) on y(t), removing its relationship with y(t-1)?
- The PACF does exactly that! It indicates the "true correlation" between a series and its n-lag.



STATIONARITY

- A time series is said to be stationary if the statistical properties such as mean, variance, and auto-correlation do not change over time.

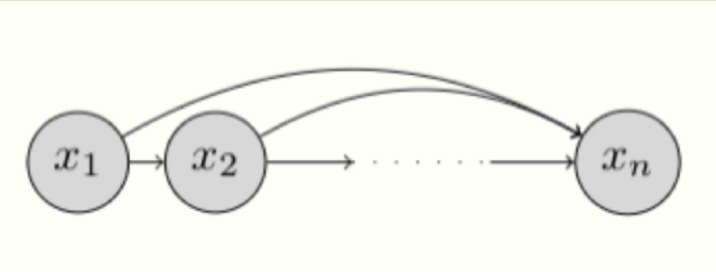


- ADF (Augmented Dickey-Fuller) test is a statistical significance test which means the test will give results in hypothesis tests with null and alternative hypotheses. As a result, we will have a p-value from which we will need to make inferences about the time series, whether it is stationary or not.

PURDUE UNIVERSITY®

AUTO-REGRESSIVE MODEL

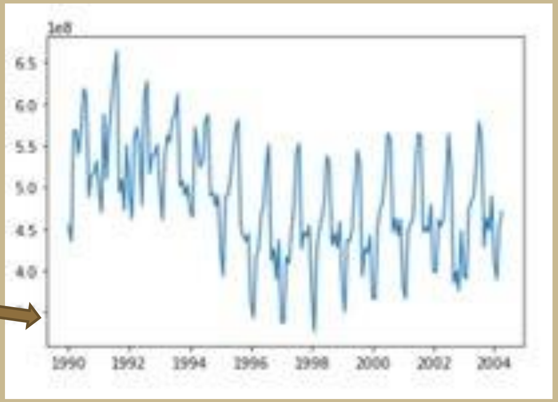
- The AR model, as its name suggests, is a regression model with significant lags acting as predictors of y(t).
- The conditions to fit an AR model are:
- The time series should be stationary. If not, it should be differences and made stationary
- The lag variables should be significant. The number of lags to be included in the model is picked using the PACF
- ACFs whose lags' significance reduces geometrically indicate that a time series is good to model with the AR model.

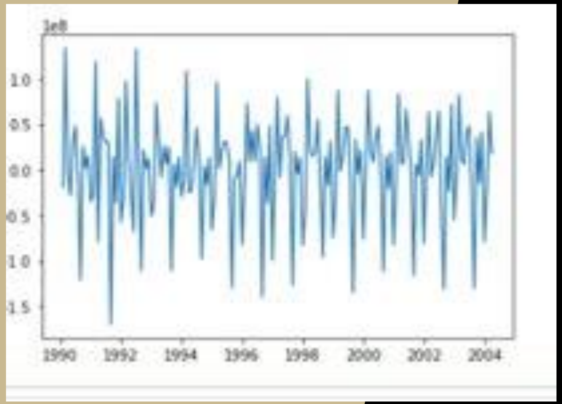


PURDUE UNIVERSITY

AR MODEL EXAMPLE

- Consider the railways data's original series, and its differenced plot.





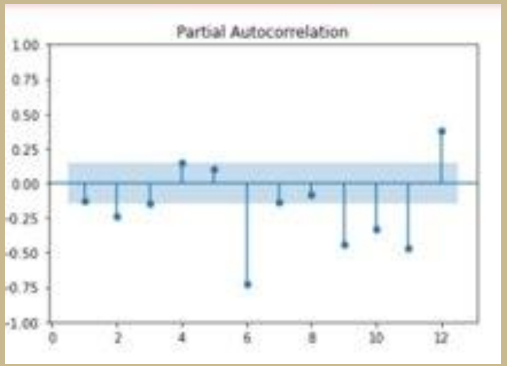
- Performing the ADF test on the differenced data

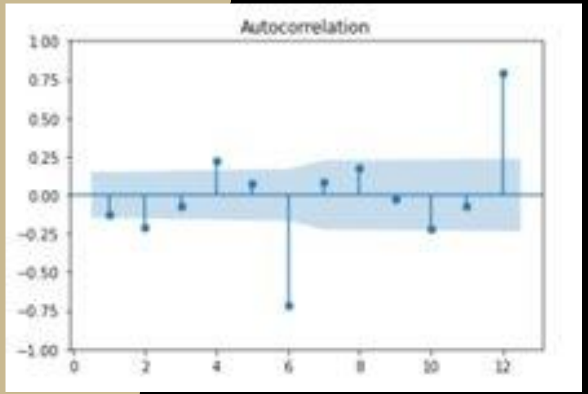
```
In [94]: from statsmodels.tsa.stattools import adfuller
         df_stationarityTest = adfuller(df_rail['y(t)'].diff().dropna())
         df_stationarityTest[1]

Out[94]: 1.781590161039216e-08
```

- Visualizing the acf and pacf of the differenced data

- There is no geometric decrease of lag-significance in the ACF. This is enough to assume that we won't get a great AR model

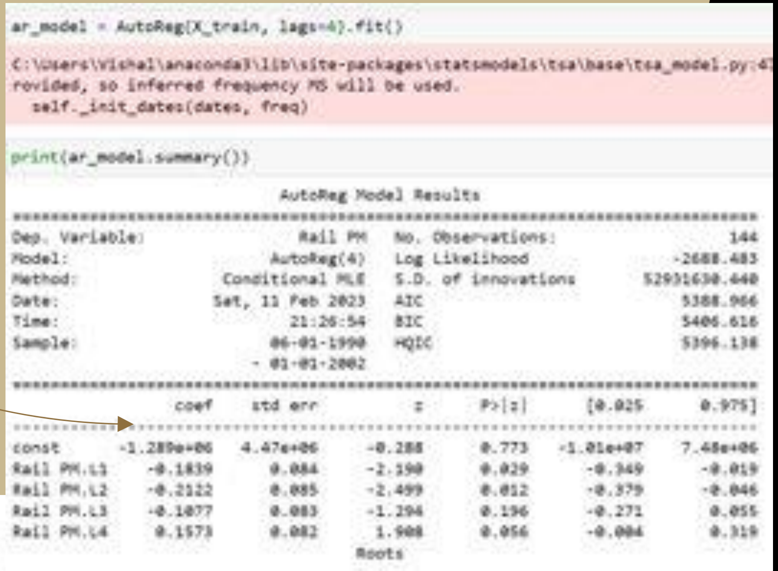




PURDUE UNIVERSITY®

AR MODEL – MODELLING AND PERFORMANCE

- Not all variables are significant
- Performance of the model not that great as expected

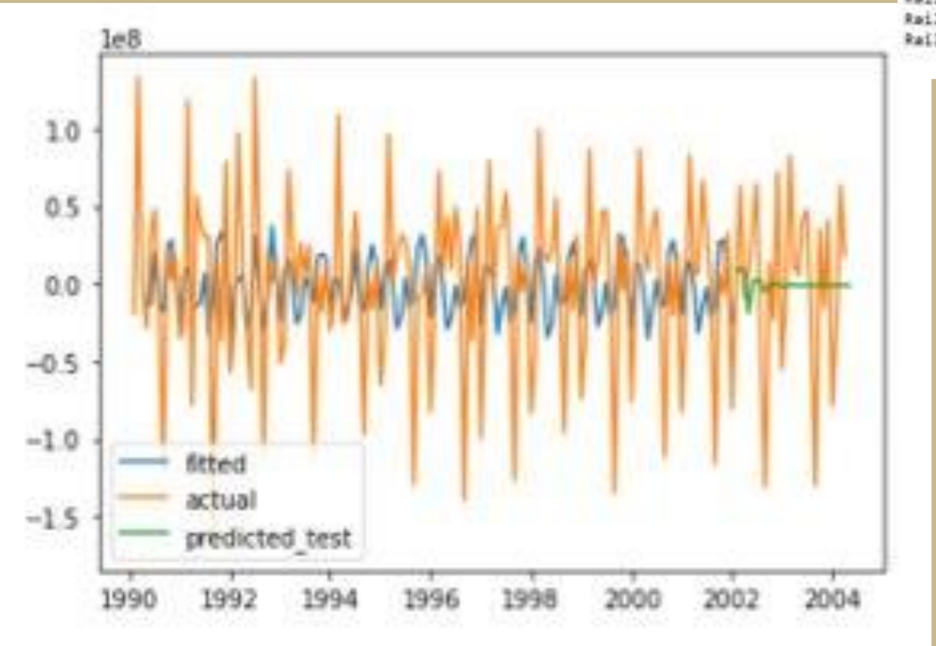


```
ar_model = AutoReg(X_train, lags=4).fit()

C:\Users\Vishal\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:4
rovided, so inferred frequency MS will be used.
  self._init_dates(dates, freq)

print(ar_model.summary())

                            AutoReg Model Results
==============================================================================
Dep. Variable:                Rail PM   No. Observations:                  144
Model:                     AutoReg(4)   Log Likelihood               -2688.483
Method:               Conditional MLE   S.D. of innovations       52931630.440
Date:                Sat, 11 Feb 2023   AIC                           5388.966
Time:                        21:26:54   BIC                           5406.616
Sample:                    06-01-1990   HQIC                          5396.138
                         - 01-01-2002
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const      -1.289e+06   4.47e+06     -0.288      0.773   -1.01e+07    7.48e+06
Rail PM.L1    -0.1839      0.084     -2.190      0.029      -0.349      -0.019
Rail PM.L2    -0.2122      0.085     -2.499      0.012      -0.379      -0.046
Rail PM.L3    -0.1077      0.083     -1.294      0.196      -0.271       0.055
Rail PM.L4     0.1573      0.082      1.908      0.056      -0.004       0.319
                                    Roots
```

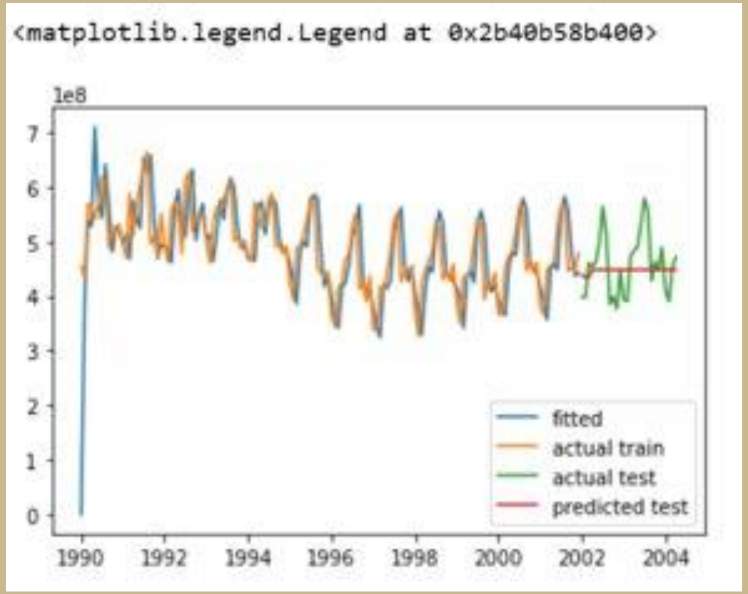


ARIMA

- ARIMA is a combination of the AR and MA model.
- The MA (moving average) model is a category of models that attempts to reduce the prediction errors by taking the error of the previous time index as an input.
- The AR model has already been covered
- The ARIMA model takes in 3 hyperparameters – (p,d,q)
- P is the number of significant lags as seen in the PACF. This is for the AR part
- D is the order of differencing required to make the series stationary
- Q is the number of significant lags as seen in the ACF. This is for the MA part

ARIMA example

- Let us model the same time series and see if we get improved results
- Since both ACF and PACF show 4 significant lags after differencing of 1 to make the series stationary, we set (p,d,q) to (4,1,4)
- Note that ideally, the ACF and/or PACF would show a geometric trend
- Observe how the model fails in the validation part. It fails to capture the seasonality.




```
import statsmodels.api as sm
ma_model = sm.tsa.arima.ARIMA(X_train, order=(0,1,4))
res = ma_model.fit()
print(res.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                Rail PM   No. Observations:                  144
Model:                 ARIMA(0, 1, 4)   Log Likelihood               -2749.466
Date:                Sat, 11 Feb 2023   AIC                           5508.931
Time:                        21:29:17   BIC                           5523.745
Sample:                    01-01-1990   HQIC                          5514.951
                         - 12-01-2001
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1         -0.0829      0.088     -0.937      0.349      -0.256       0.090
ma.L2          0.1990      0.075      2.656      0.008       0.052       0.346
ma.L3         -0.0934      0.062     -1.499      0.134      -0.215       0.029
ma.L4          0.3751      0.060      6.266      0.000       0.258       0.492
sigma2      2.694e+15   1.05e-17   2.56e+32      0.000    2.69e+15    2.69e+15
------------------------------------------------------------------------------
```

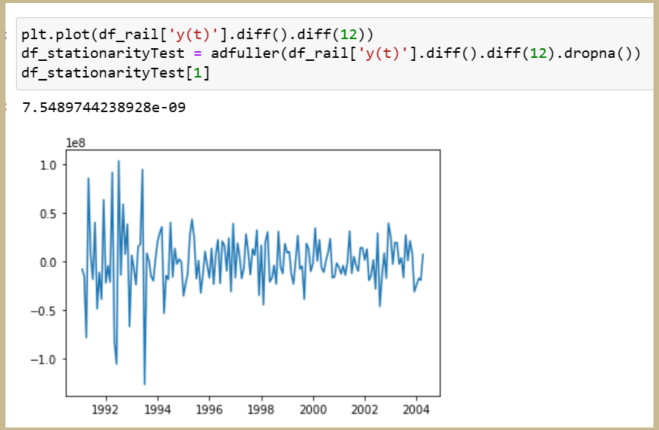
SARIMA

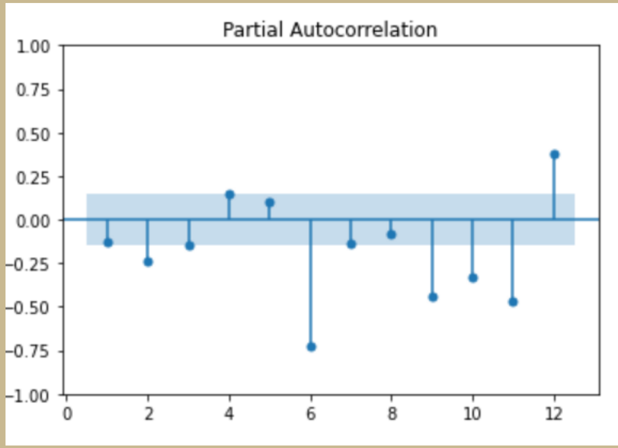
- SARIMA stands for seasonal ARIMA. ARIMA fails to capture seasonality by itself. So, we add a seasonal component.
- Y(t) = ay(t-1)+by(t-2)....+**s1y(t-12)**....beta+error
- Along with the (p,d,q) we also need to tune the seasonal order - (P,D,Q,S). For this we observe the signifiance of the lags at a seasonal level. For example, if the frequency of the data is 12 (12 months in a year), we observe what the ACF and PACF say about lag 12.

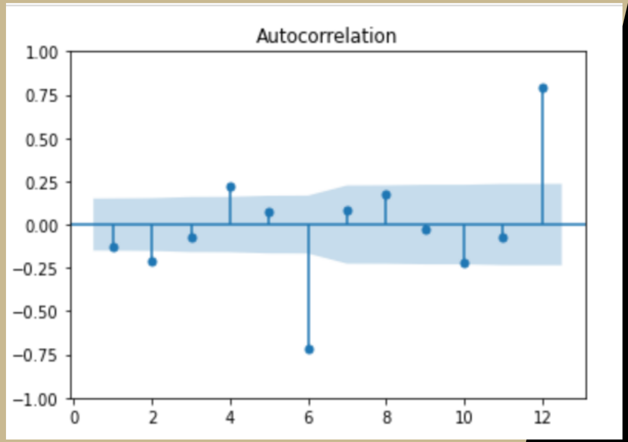
PURDUE UNIVERSITY

SARIMA example

- Again, let us attempt to improve our results on the rail time series





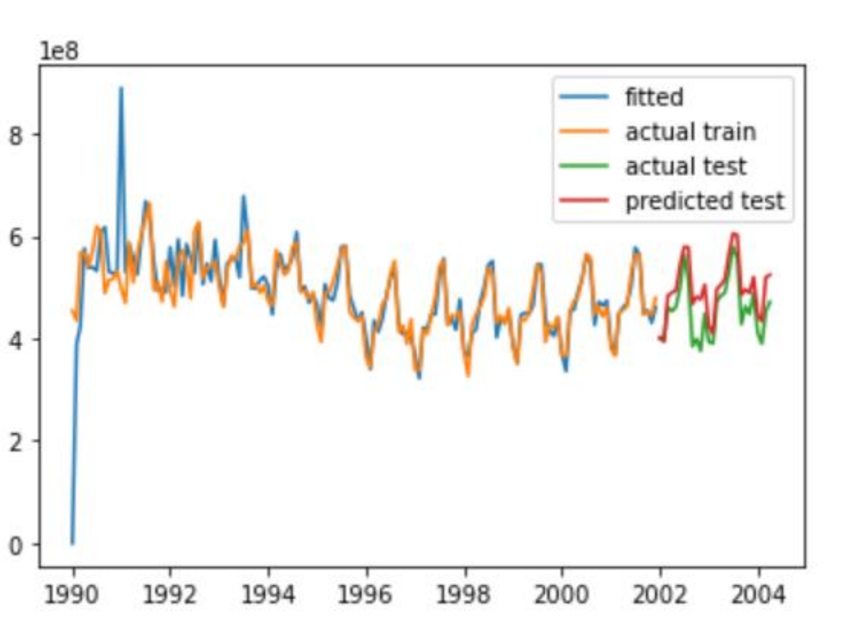


- In SARIMA, when the frequency is 12, differencing by 12 is equivalent to seasonal differencing by 1.
- ACF and PACF both show significant lag-12
- Setting seasonal order to (1,1,1,12)

SARIMA example

```
from statsmodels.tsa.statespace.sarimax import SARIMAX
sarimax_model = SARIMAX(X_train, order=(4,1,4),seasonal_order=(1,1,1,12))
res = sarimax_model.fit()
print(res.summary())
```

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.4628	0.335	1.381	0.167	-0.194	1.119
ar.L2	0.0793	0.327	0.242	0.808	-0.562	0.720
ar.L3	-0.6833	0.229	-2.985	0.003	-1.132	-0.235
ar.L4	-0.1010	0.205	-0.492	0.623	-0.503	0.301
ma.L1	-0.6294	0.340	-1.849	0.064	-1.296	0.038
ma.L2	-0.0021	0.319	-0.007	0.995	-0.627	0.623
ma.L3	0.7875	0.235	3.344	0.001	0.326	1.249
ma.L4	-0.1099	0.220	-0.500	0.617	-0.541	0.321
ar.S.L12	0.4878	0.084	5.803	0.000	0.323	0.653
ma.S.L12	-0.7377	0.104	-7.107	0.000	-0.941	-0.534
sigma2	6.662e+14	2.94e-15	2.26e+29	0.000	6.66e+14	6.66e+14



Though not all the features are significant, observe how the results improved

PURDUE UNIVERSITY

Smoothing Methods

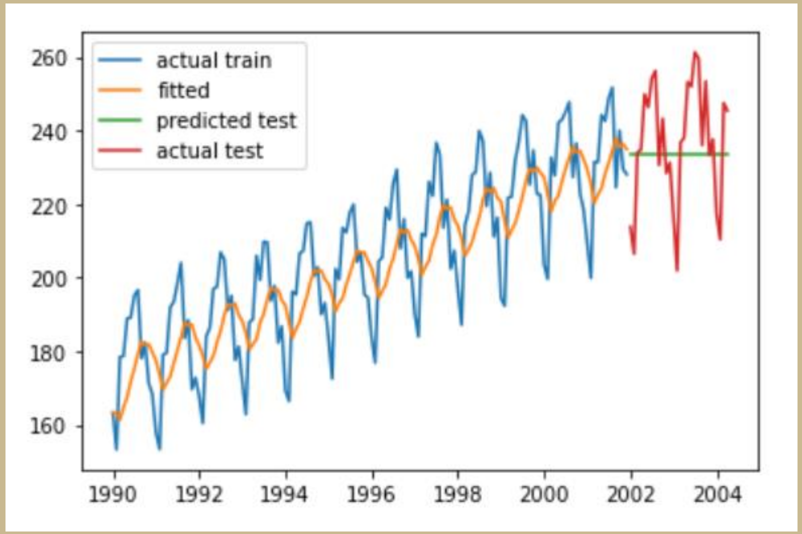
- Smoothing methods are involve averaging out past and present observations to get a ball-park forecast. Hence, 'smoothing'.
- The simplest smoothing method would just be averaging out, let us say, the previous 5 observation. 1,2,3,2,2 in the past points would yield a prediction of 10/5 = 2
- There are variations of smoothing:
- Simple Exponential Smoothing
- Double Exponential Smoothing
- Holt-Winter's Smoothing

Simple Exponential Smoothing

```python
from statsmodels.tsa.api import ExponentialSmoothing, SimpleExpSmoothing, Holt
```

```python
X_train = df['VMT (billions)'][:144]
X_test = df['VMT (billions)'][144:]
```

```python
ses_model = SimpleExpSmoothing(X_train).fit(
    smoothing_level=0.2, optimized=False)
```



- Form of weighted average where recent observations are given highest weights
- Larger the value of T, lesser is (1-alpha)^T
- Alpha is the learning rate, which is defined by the user.

PURDUE UNIVERSITY

Double Exponential Smoothing

$$F_{t+k} = L_t + kT_t$$

$$L_t = \alpha Y_t + (1-\alpha)(L_{t-1} + T_{t-1})$$
$$T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1}.$$

- Simple Exponential does not account for trend and yields flat forecasts.
- Double exponential smoothing factors in a trend component

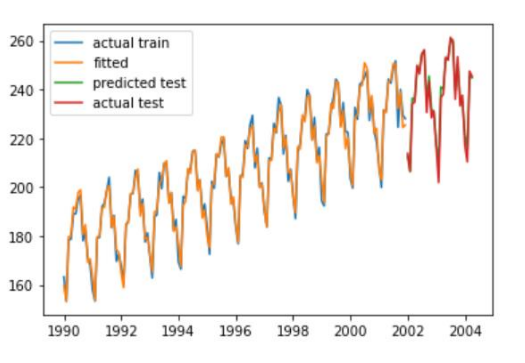
```
holt_model = Holt(X_train).fit(
    smoothing_level=0.2, smoothing_trend=0.2, optimized=False
)
```

- The Lt equation means that the level at time t is a weighted average of the actual value at time t and the level in the previous period, adjusted for trend
- The Tt equation means that the trend at time t is a weighted average of the trend in the previous period and the more recent information on the change in level.3

PURDUE UNIVERSITY

Triple (Holt-Winter's) Exponential Smoothing

$$F_{t+k} = (L_t + kT_t)\, S_{t+k-M}$$




```
exp_model = ExponentialSmoothing(
    X_train,
    seasonal_periods=12,
    trend="add",
    seasonal="add",
    use_boxcox=True,
    initialization_method="estimated",
).fit()
```

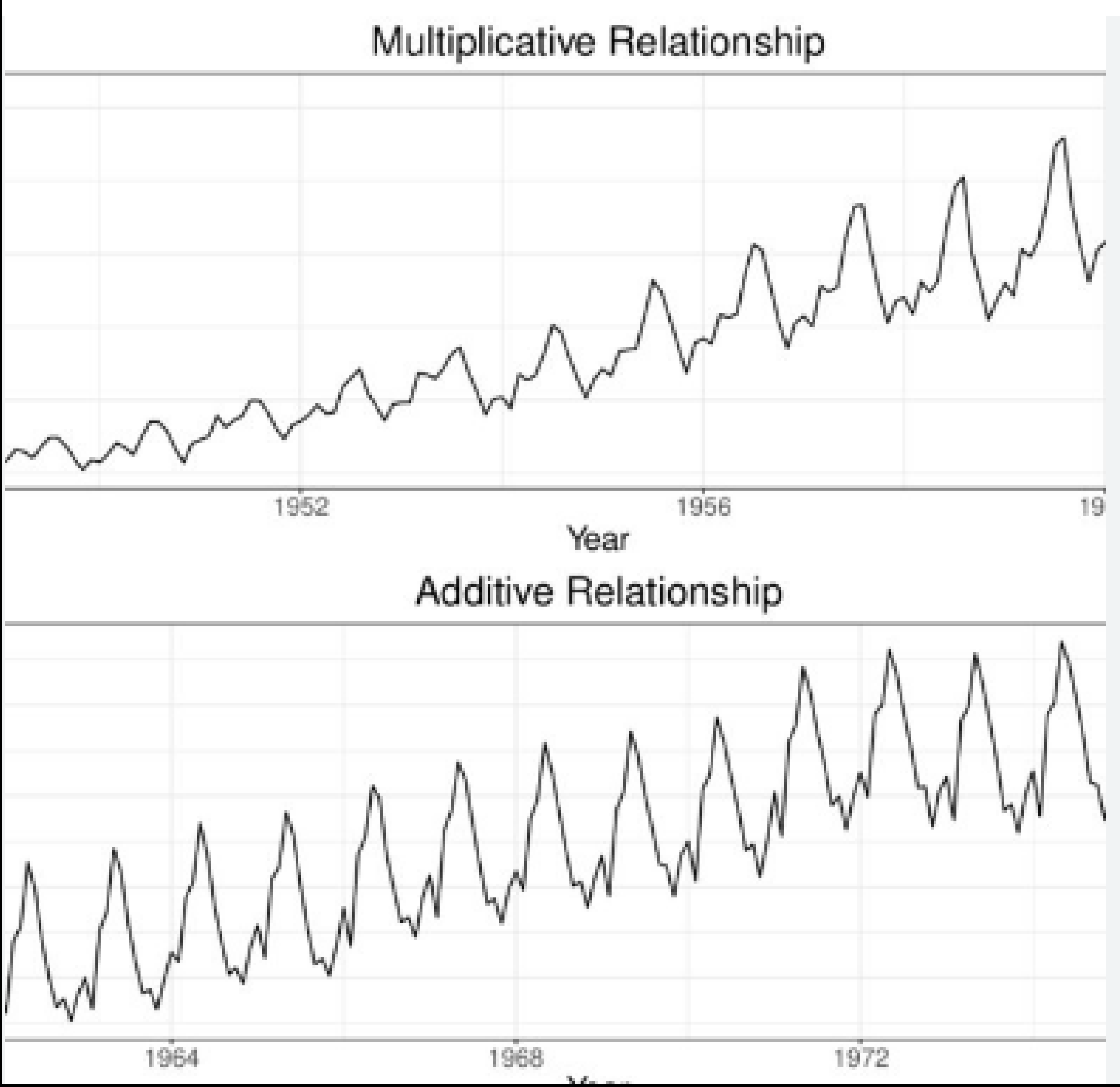
$$\begin{aligned} L_t &= \alpha Y_t / S_{t-M} + (1-\alpha)(L_{t-1} + T_{t-1}) \\ T_t &= \beta\,(L_t - L_{t-1}) + (1-\beta)T_{t-1} \\ S_t &= \gamma Y_t / L_t + (1-\gamma) S_{t-M}. \end{aligned}$$

- Holt-Winter's smoothing factors in level, trend, and seasonality.
- The trend equation now includes adjustment for seasonality
- The seasonality equation is added to the forecast function

PURDUE UNIVERSITY

Additive vs Multiplicative Trend/Seasonality

- Additive means linear (straight line), and multiplicative means there are changes to widths or heights of periods over time (percentage increase).



WHAT NEXT?

- Auto ARIMA
- Complex time-series data
- Deep Learning models for Time-series

PURDUE UNIVERSITY®

APPENDIX - EQUATIONS

- Simple linear regression - Y(t) = beta0 + beta1*t
- Multiple Linear Regression - beta0 + beta1*t + beta2*season1.....beta13*season12
- AR Model - Y(t) = ay(t-1)+by(t-2)....+beta+error
- MA Model - Y(t) = beta + ay(t-1)+be(t-1) + error
- Simple Exponential smoothing- $\hat{y}_{T+1|T} = \sum_{j=0}^{T-1} \alpha(1-\alpha)^j y_{T-j} + (1-\alpha)^T \ell_0.$
- Double Exponential Smoothing
$$\begin{aligned} L_t &= \alpha Y_t + (1-\alpha)(L_{t-1} + T_{t-1}) \\ T_t &= \beta\,(L_t - L_{t-1}) + (1-\beta) T_{t-1}. \end{aligned}$$
- Triple Exponential Smoothing
$$\begin{aligned} L_t &= \alpha Y_t / S_{t-M} + (1-\alpha)(L_{t-1} + T_{t-1}) \\ T_t &= \beta\,(L_t - L_{t-1}) + (1-\beta) T_{t-1} \\ S_t &= \gamma Y_t / L_t + (1-\gamma) S_{t-M}. \end{aligned}$$

PURDUE UNIVERSITY