statinfer

Time Series Analysis & Forecasting

Venkat Reddy

Contents

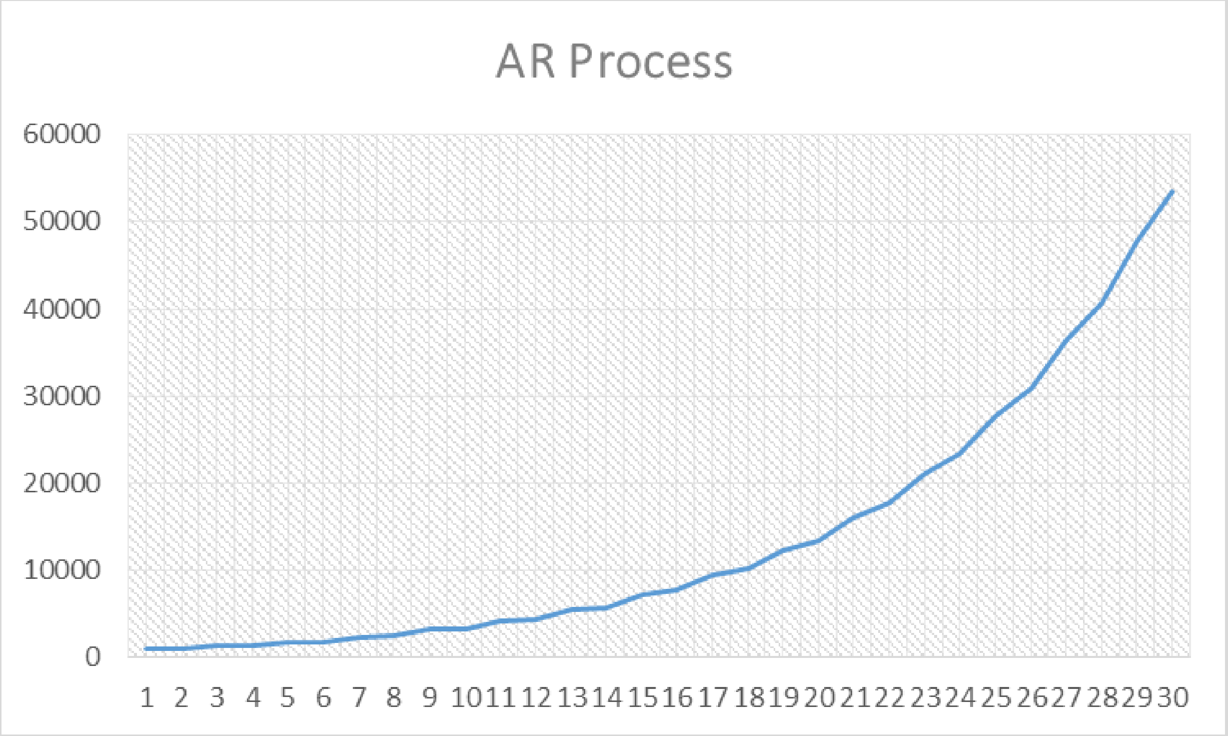
ARIMA Models

- Autoregressive Integrated Moving-average
- A “stochastic” modeling approach that can be used to calculate the probability of a future value lying between two specified limits

AR & MA Models

- Autoregressive AR process:
  - Series current values depend on its own previous values
  - AR(p) - Current values depend on its own p-previous values
  - P is the order of AR process
- Moving average MA process:
  - The current deviation from mean depends on previous deviations
  - MA(q) - The current deviation from mean depends on q- previous deviations
  - q is the order of MA process
- Autoregressive Moving average ARMA process

AR Process

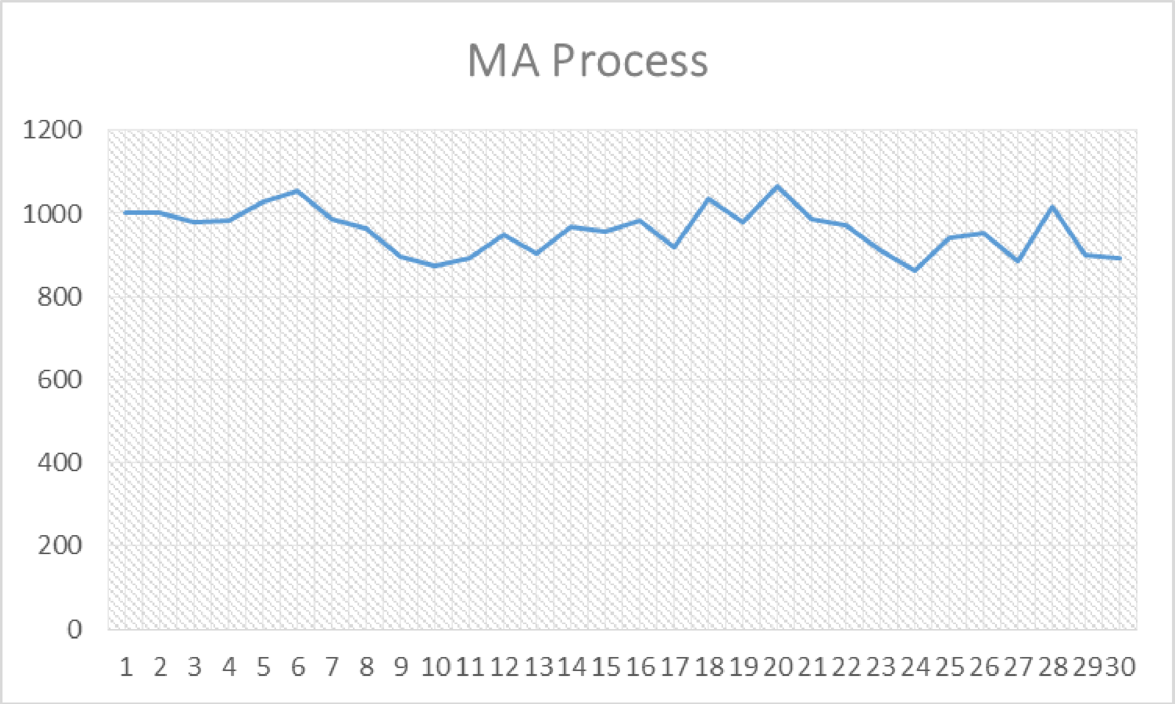


AR(1) $y_t = a1^* \, y_{t-1}$

AR(2) $y_t = a1^* \, y_{t-1} + a2^* \, y_{t-2}$

AR(3) $y_t = a1^* \, y_{t-1} + a2^* \, y_{t-2} + a3^* \, y_{t-2}$

MA Process



MA(1) $\varepsilon_t = b1*\varepsilon_{t-1}$

MA(2) $\varepsilon_t = b1*\varepsilon_{t-1} + b2*\varepsilon_{t-2}$

MA(3) $\varepsilon_t = b1*\varepsilon_{t-1} + b2*\varepsilon_{t-2} + b3*\varepsilon_{t-3}$

ARIMA Models

- Autoregressive (AR) process:
  - Series current values depend on its own previous values
- Moving average (MA) process:
  - The current deviation from mean depends on previous deviations
- Autoregressive Moving average (ARMA) process
- Autoregressive Integrated Moving average (ARIMA)process.

- ARIMA is also known as Box-Jenkins approach. It is popular because of its generality;
- It can handle any series, with or without seasonal elements, and it has well-documented computer programs

ARIMA Model

Y_t → AR Component→ MA Component → ε_t

(long term) (short term) (white noise error)

ARMA (2,1) $y_t = a_1 y_{t-1} + a_2 y_{t-2} + b_1 \varepsilon_{t-1}$

ARMA (3,1) $y_t = a_1 y_{t-1} + a_2 y_{t-2} + a_3 y_{t-3} + b_1 \varepsilon_{t-1}$

To build a time series model issuing ARIMA, we need to study the time series and identify p,q

ARIMA equations

- ARIMA(1,0,0)
  - $y_t = a_1 y_{t-1} + \varepsilon_t$
- ARIMA(2,0,0)
  - $y_t = a_1 y_{t-1} + a_2 y_{t-2} + \varepsilon_t$

Overall Time series Analysis & Forecasting Process

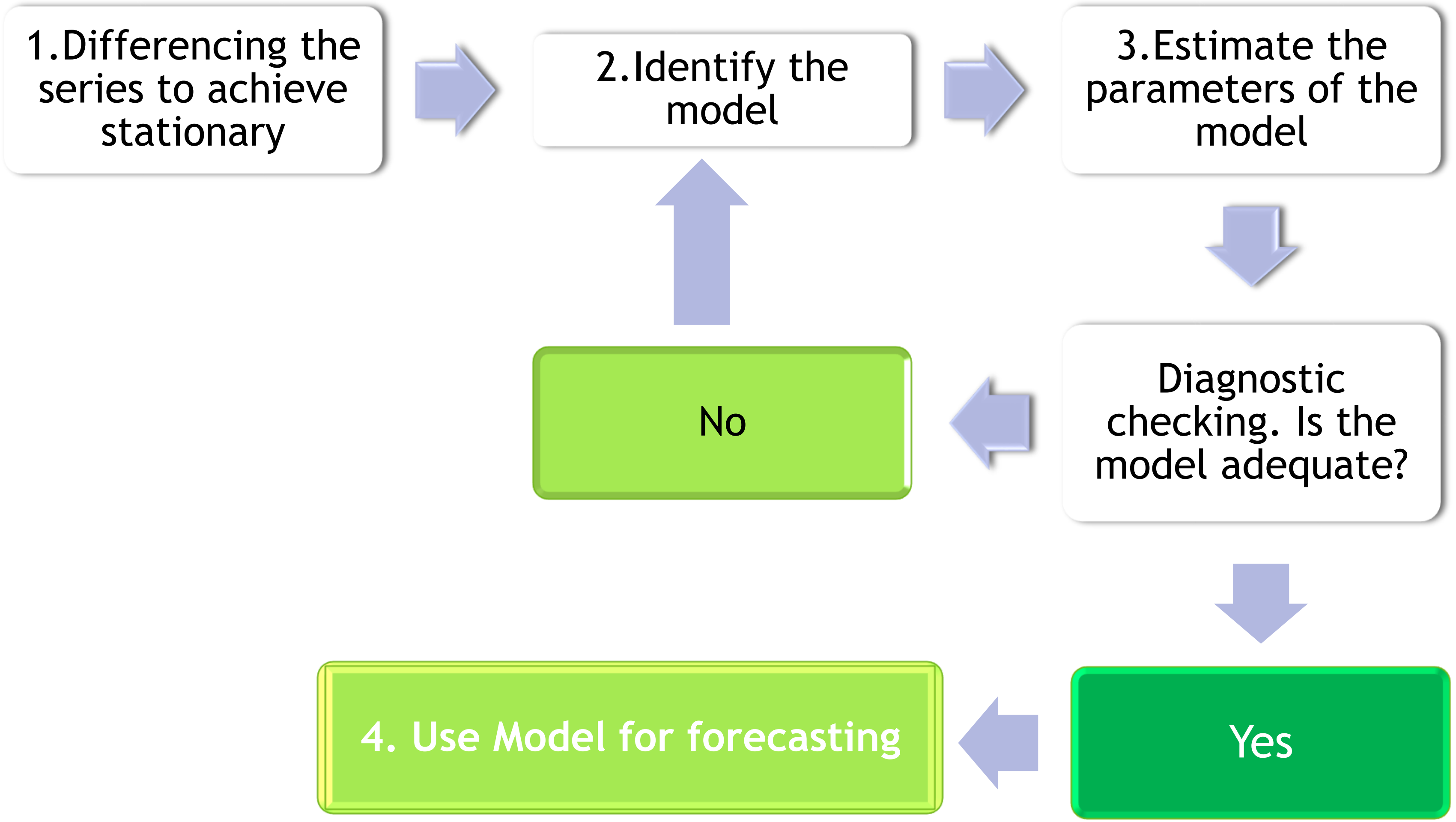
- Prepare the data for model building- Make it stationary
- Identify the model type
- Estimate the parameters
- Forecast the future values

ARIMA (p,d,q) modeling

To build a time series model issuing ARIMA, we need to study the time series and identify p,d,q

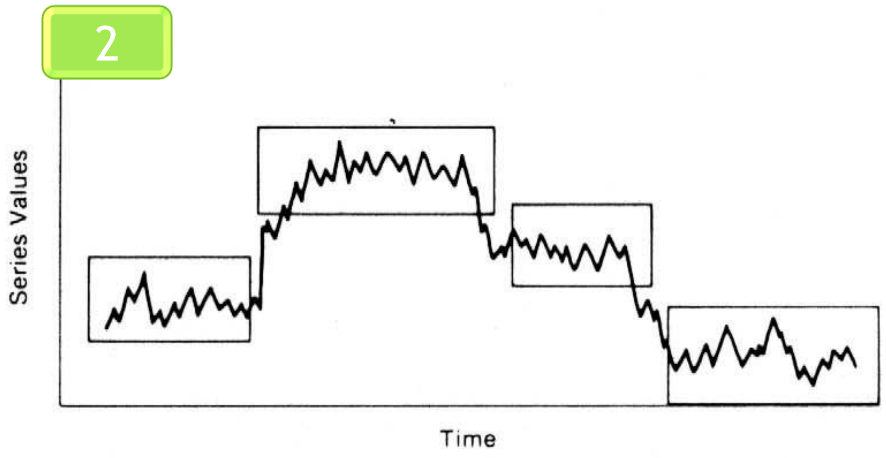
- **Ensuring Stationarity**
  - Determine the appropriate values of d
- **Identification:**
  - Determine the appropriate values of p & q using the ACF, PACF, and unit root tests
  - p is the AR order, d is the integration order, q is the MA order
- **Estimation :**
  - Estimate an ARIMA model using values a1 , a2, b1 b2 etc., you think are appropriate.
- **Diagnostic checking:**
  - Check residuals of estimated ARIMA model(s) to see if they are white noise; pick best model with well behaved residuals.
- **Forecasting:**
  - Produce out of sample forecasts or set aside last few data points for in-sample forecasting.

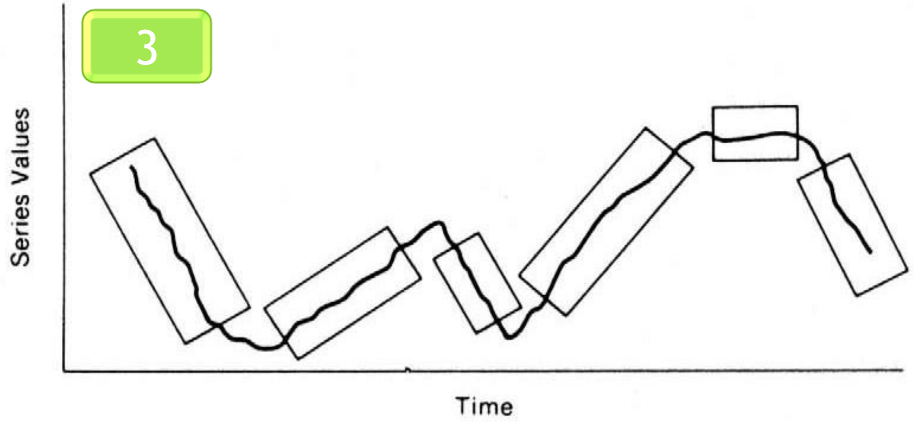
The Box-Jenkins Approach



Step-1 : Stationarity

Some non stationary series





Stationarity

- In order to model a time series with the Box-Jenkins approach, the series has to be stationary
- In practical terms, the series is stationary if tends to wonder more or less uniformly about some fixed level
- In statistical terms, a stationary process is assumed to be in a particular state of statistical equilibrium, i.e., $p(x_t)$ is the same for all t
- In particular, if z_t is a stationary process, then the first difference $\nabla z_t = z_t - z_{t-1}$ and higher differences $\nabla^d z_t$ are stationary

Testing Stationarity

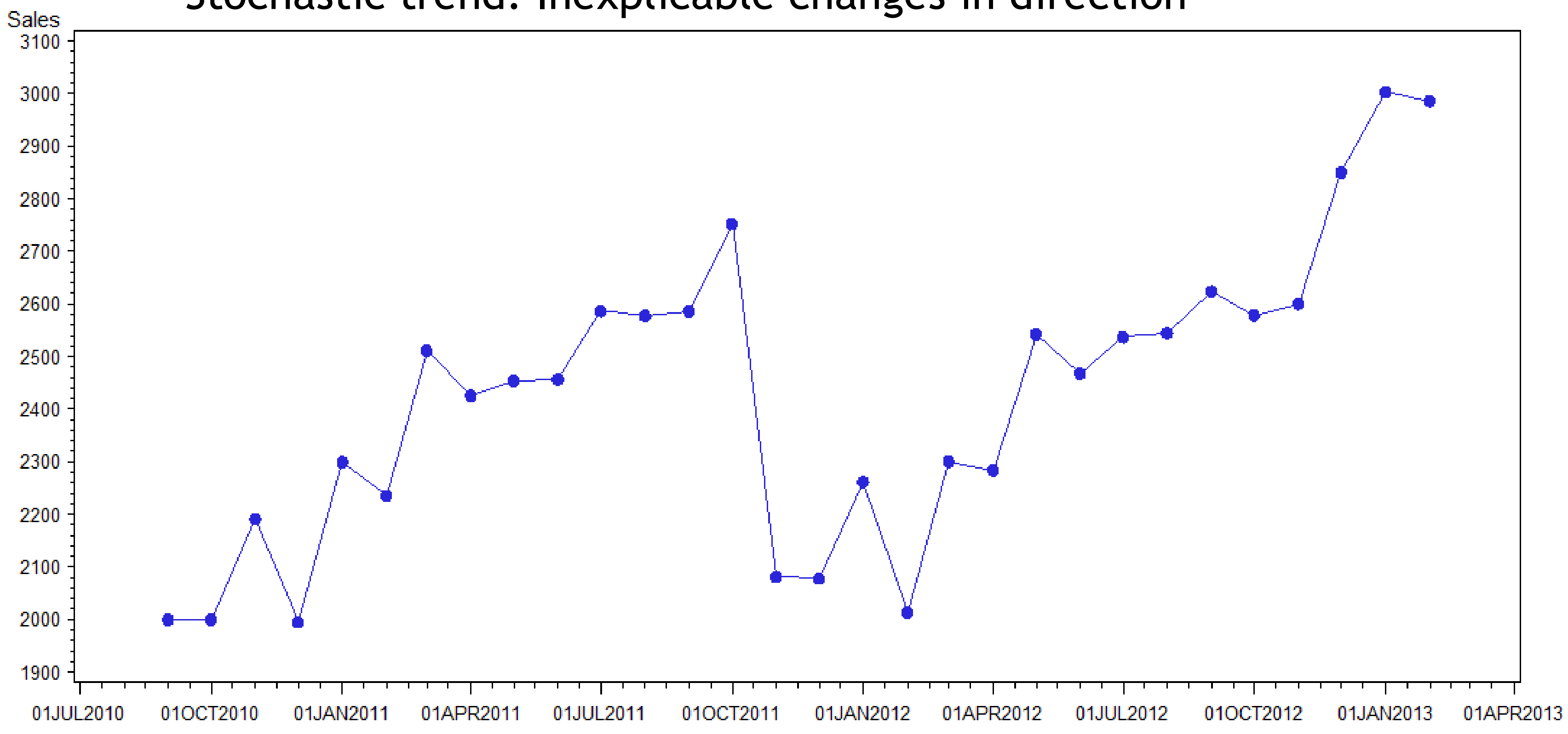
- Dickey-Fuller test
  - P value has to be less than 0.05 or 5%
  - If p value is greater than 0.05 or 5%, you accept the null hypothesis, you conclude that the time series has a unit root.
  - In that case, you should first difference the series before proceeding with analysis.

- What DF test ?
  - Imagine a series where a fraction of the current value is depending on a fraction of previous value of the series.
  - DF builds a regression line between fraction of the current value Δy_t and fraction of previous value δy_{t-1}
  - The usual t-statistic is not valid, thus D-F developed appropriate critical values. **If P value of DF test is <5% then the series is stationary**

Demo: Testing Stationarity

- Sales_1 data

Stochastic trend: Inexplicable changes in direction

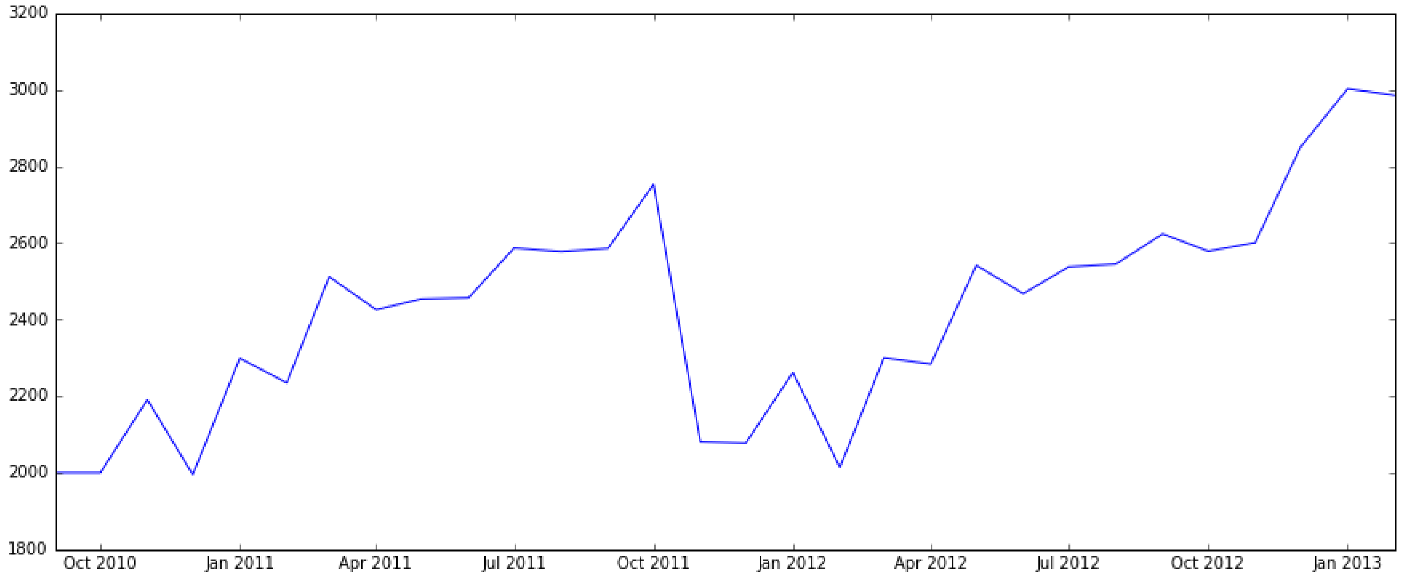


Demo: Testing Stationarity

```
import pandas as pd
ts_data1 =pd.read_csv("D:\\Google Drive\\Training\\5. Machine Learning 
Python\\3.Reference\\8. Time Series Analysis in 
Python\\ARIMA\\Datasets\\Sales_TS_Data1.csv",header=0,parse_dates=[0], 
index_col='Month')

ts_data1
ts_data1.shape
ts_data1.head()
ts_data1.dtypes

ts1 = ts_data1["Sales"]
print(ts1)
```

Demo: Testing Stationarity

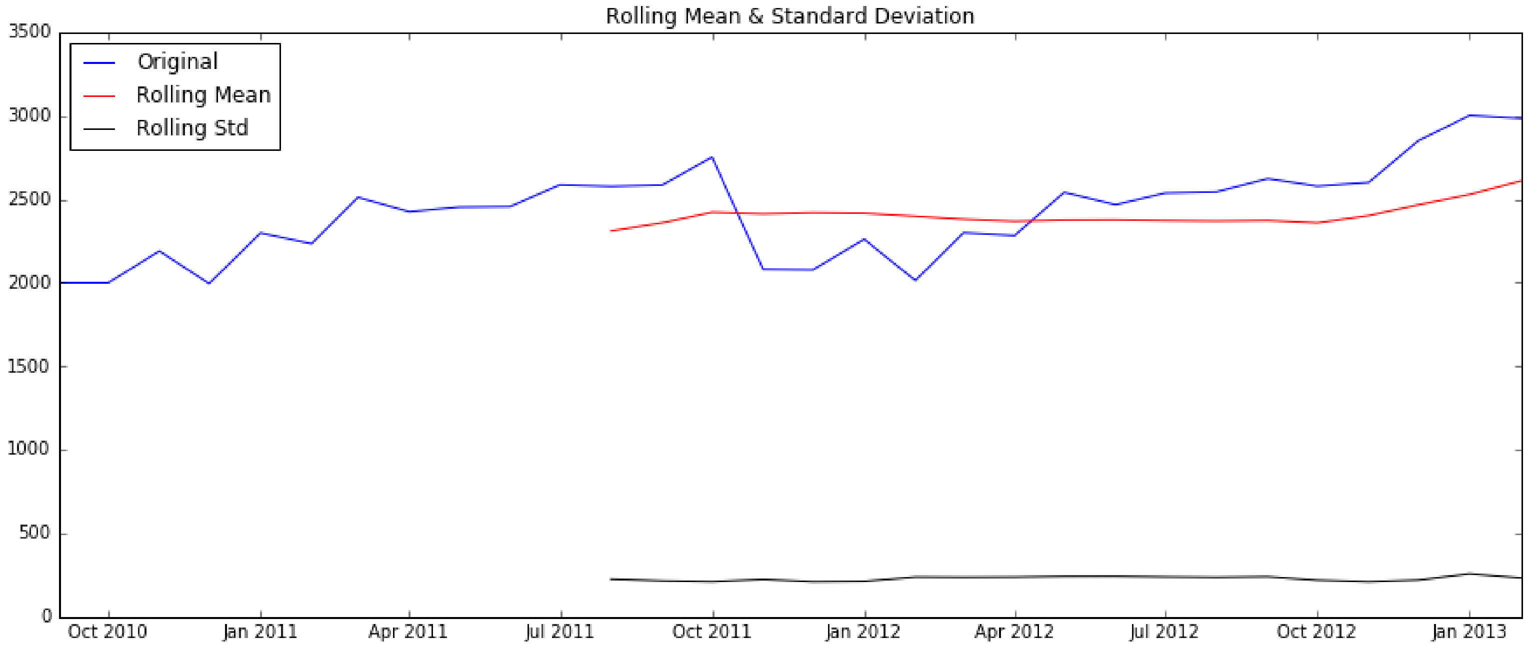
```
#Drawing Time Series
import numpy as np
import matplotlib.pylab as plt
from scipy import stats
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 15, 6
# Image length and width

#Plotting
plt.plot(ts1)
```



Demo: Testing Stationarity

```
from statsmodels.tsa.stattools import adfuller
def test_stationarity(timeseries):
      #Determing rolling statistics
    rolmean = pd.rolling_mean(timeseries, window=12)
    rolstd = pd.rolling_std(timeseries, window=12)

   #Plot rolling statistics:
    fig = plt.figure(figsize=(15, 6))
    orig = plt.plot(timeseries, color='blue',label='Original')
    mean = plt.plot(rolmean, color='red', label='Rolling Mean')
    std = plt.plot(rolstd, color='black', label = 'Rolling Std')
    plt.legend(loc='best')
    plt.title('Rolling Mean & Standard Deviation')
    plt.show()

    #Perform Dickey-Fuller test:
    print ('Results of Dickey-Fuller Test:')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print (dfoutput)

#Test for Stationarity
test_stationarity(ts1)
```

Demo: Testing Stationarity




```
Results of Dickey-Fuller Test:
Test Statistic                 -1.773996
p-value                         0.393446
#Lags Used                      0.000000
Number of Observations Used    29.000000
Critical Value (1%)            -3.679060
Critical Value (10%)           -2.623158
Critical Value (5%)            -2.967882
dtype: float64
```

Achieving Stationarity

- Differencing : Transformation of the series to a new time series where the values are the differences between consecutive values
- Procedure may be applied consecutively more than once, giving rise to the "first differences", "second differences", etc.
- Regular differencing (RD)

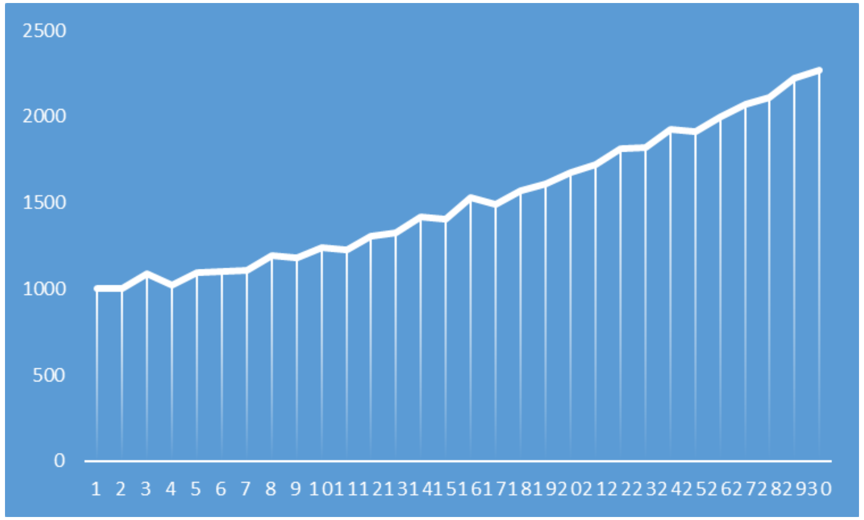
(1st order) $\nabla x_t = x_t - x_{t-1}$

(2nd order) $\nabla^2 x_t = (\nabla x_t - \nabla x_{t-1}) = x_t - 2x_{t-1} + x_{t-2}$

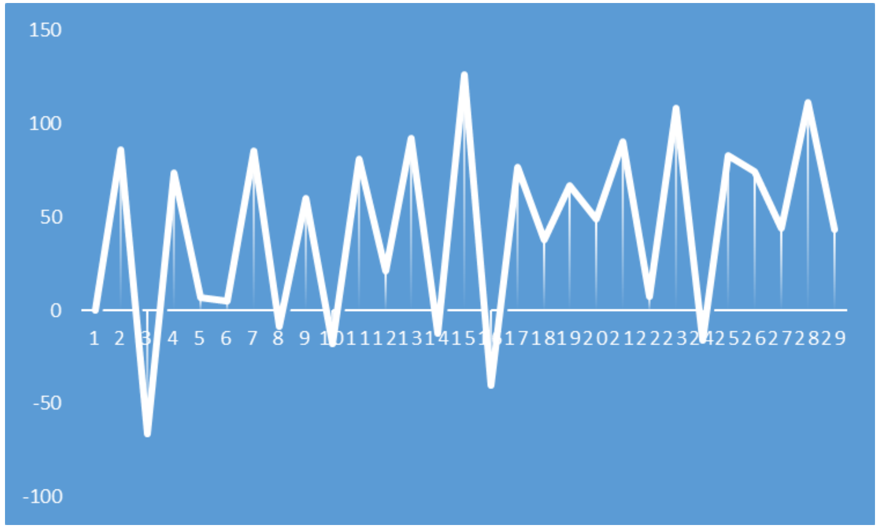
- It is unlikely that more than two regular differencing would ever be needed
- Sometimes regular differencing by itself **is not** sufficient and prior transformation is also needed

Differentiation

Actual Series



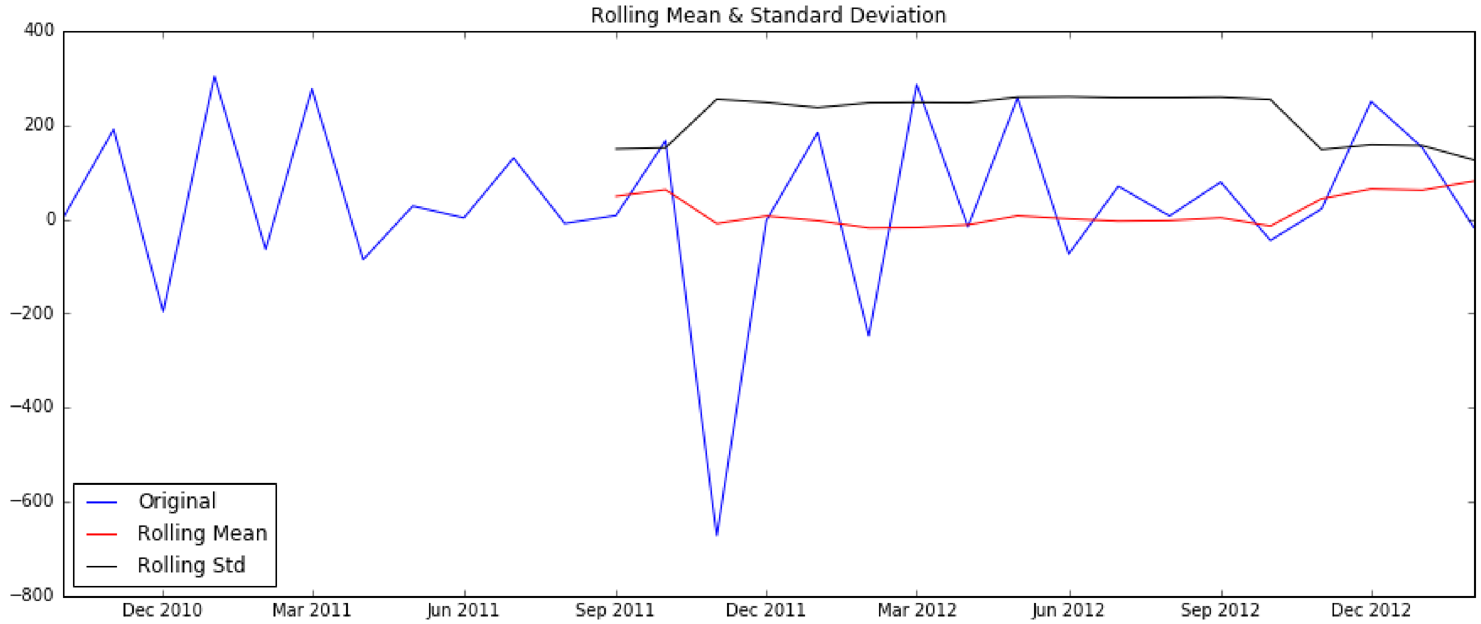
Series After Differentiation



Demo: Achieving Stationarity

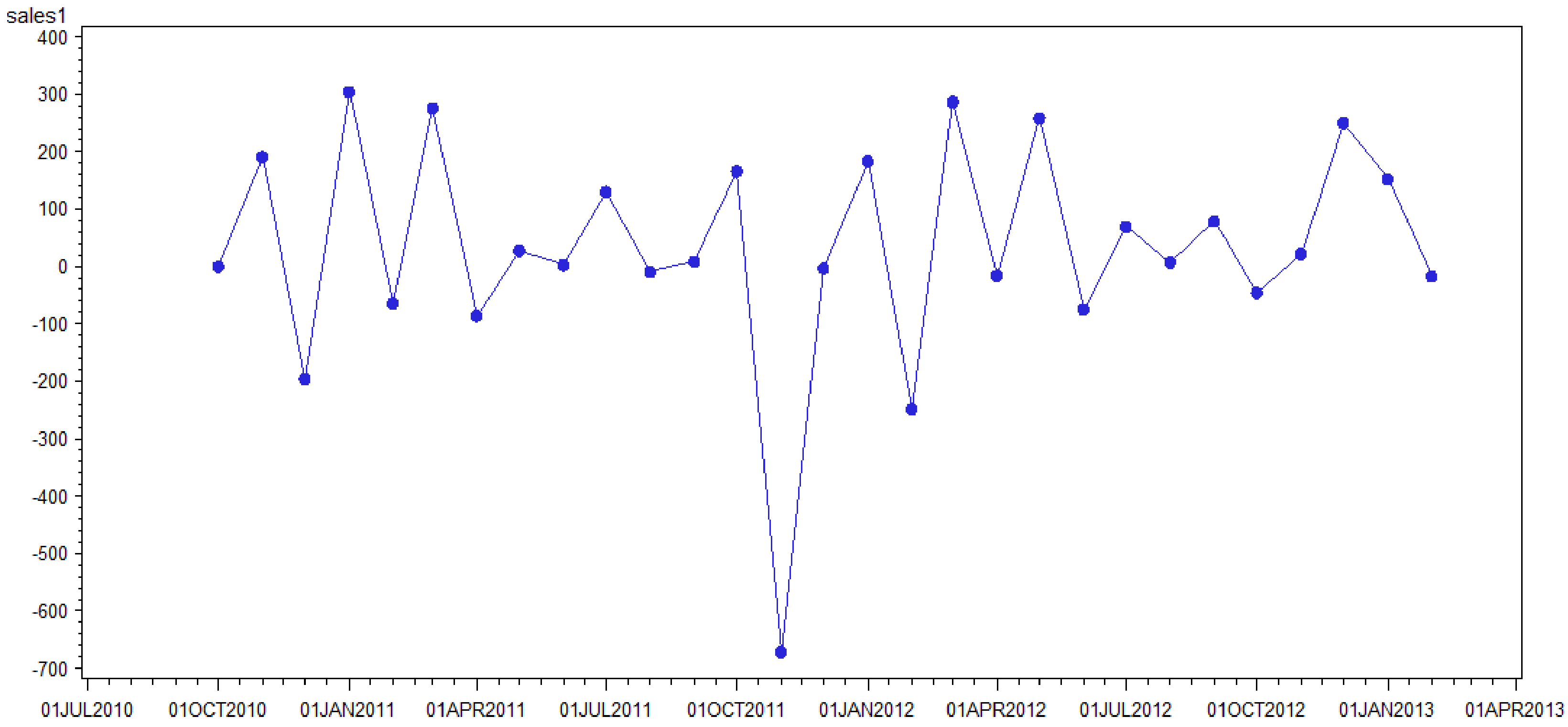
```
#The above series is not stationary
#Let us take lag1 difference
ts1_det = ts_data1['Sales'] - ts_data1['Sales'].shift()
ts1_det.plot(figsize=(15, 6))
plt.plot(ts1_det)

#Test for stationarity on lag1 difference data
#Need to remove missing values
ts1_det.dropna(inplace=True)
test_stationarity(ts1_det)
```

Demo: Achieving Stationarity




```
Results of Dickey-Fuller Test:
Test Statistic                 -7.711682e+00
p-value                         1.258977e-11
#Lags Used                      0.000000e+00
Number of Observations Used     2.800000e+01
Critical Value (1%)            -3.688926e+00
Critical Value (10%)           -2.625296e+00
Critical Value (5%)            -2.971989e+00
dtype: float64
```

Demo: Achieving Stationarity



ARIMA Model

Y_t → AR filter → Integration filter → MA filter → ε_t

(long term) (stochastic trend) (short term) (white noise error)

ARIMA (2,0,1) $y_t = a_1 y_{t-1} + a_2 y_{t-2} + b_1 \varepsilon_{t-1}$

ARIMA (3,0,1) $y_t = a_1 y_{t-1} + a_2 y_{t-2} + a_3 y_{t-3} + b_1 \varepsilon_{t-1}$

ARIMA (1,1,0) $\Delta y_t = a_1 \Delta y_{t-1} + \varepsilon_t$, where $\Delta y_t = y_t - y_{t-1}$

ARIMA (2,1,0) $\Delta yt = a_1 \Delta y_{t-1} + a_2 \Delta y_{t-2} + \varepsilon_t$ where $\Delta yt = yt - yt-1$

To build a time series model issuing ARIMA, we need to study the time series and identify p,d,q

ARIMA equations

- ARIMA(1,0,0)
  - $y_t = a_1 y_{t-1} + \varepsilon_t$
- ARIMA(2,0,0)
  - $y_t = a_1 y_{t-1} + a_2 y_{t-2} + \varepsilon_t$
- ARIMA (2,1,1)
  - $\Delta yt = a_1 \Delta y_{t-1} + a_2 \Delta y_{t-2} + b_1 \varepsilon_{t-1}$ where $\Delta yt = yt - yt\text{-}1$

Achieving Stationarity-Other methods

- Is the trend stochastic or deterministic?
  - If stochastic (inexplicable changes in direction): use differencing
  - If deterministic(plausible physical explanation for a trend or seasonal cycle) : use regression
- Check if there is variance that changes with time
  - YES : make variance constant with log or square root transformation
- Remove the trend in mean with:
  - 1st/2nd order differencing
  - Smoothing and differencing (seasonality)
- If there is seasonality in the data:
  - Moving average and differencing
  - Smoothing

Step2 : Identification

Identification of orders p and q

- Identification starts with d
- ARIMA(p,d,q)
- What is Integration here?
- First we need to make the time series stationary
- We need to learn about ACF & PACF to identify p,q

- Once we are working with a stationary time series, we can examine the ACF and PACF to help identify the proper number of lagged y (AR) terms and ε (MA) terms.

Autocorrelation Function (ACF)

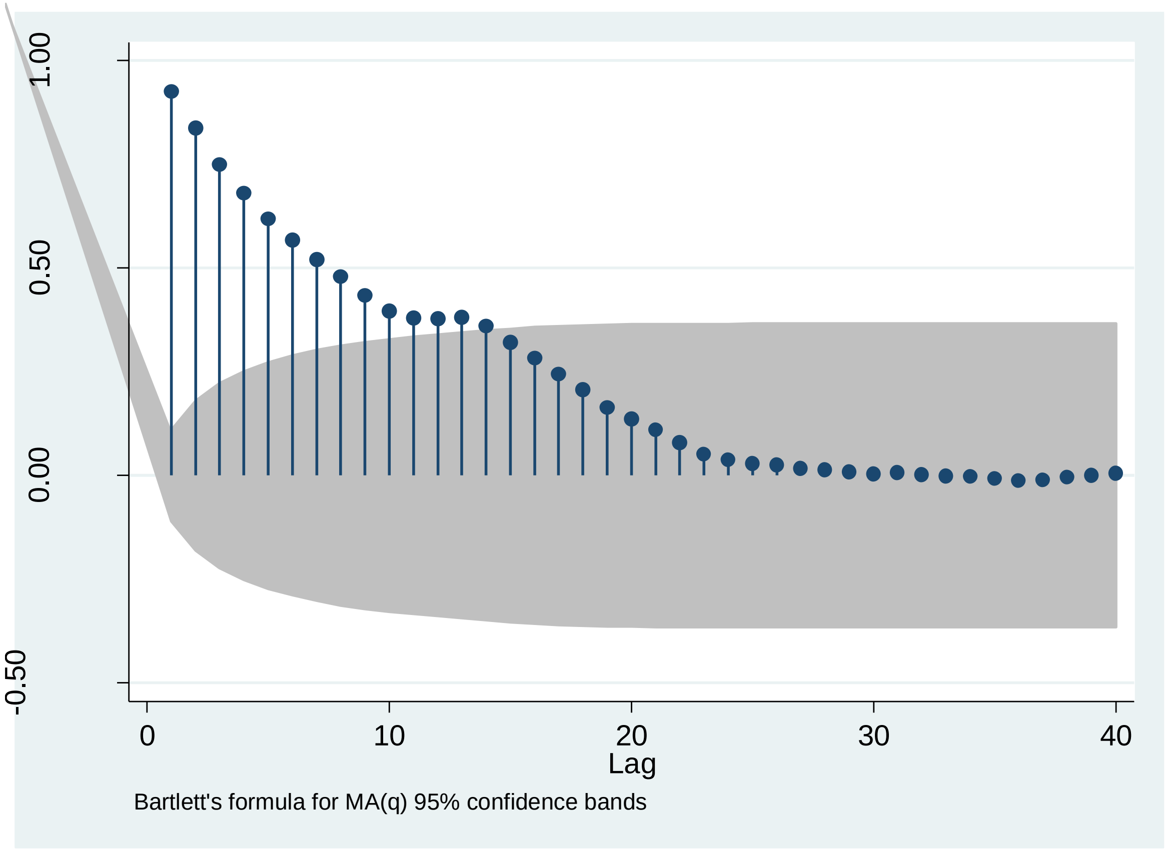
- Autocorrelation is a correlation coefficient. However, instead of correlation between two different variables, the correlation is between two values of the same variable at times X_i and X_{i+k}.
- Correlation with lag-1, lag2, lag3 etc.,
- The ACF represents the degree of persistence over respective lags of a variable.

$\rho_k = \gamma_k / \gamma_0$ = covariance at lag k/ variance

$$\rho_k = \frac{E[(y_t - \mu)(y_{t-k} - \mu)]^2}{E[(y_t - \mu)^2]}$$

ACF (0) = 1, ACF (k) = ACF (-k)

ACF Graph

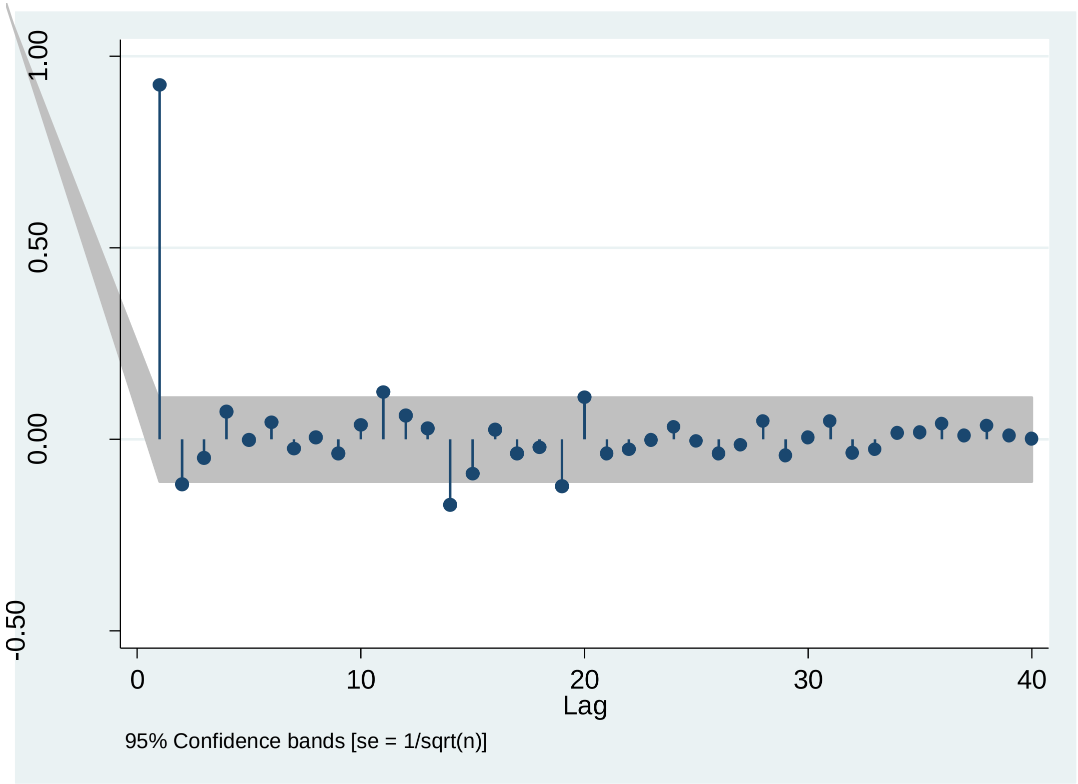


1.00
0.50
0.00
-0.50
0
10
20
30
40
Lag
Bartlett's formula for MA(q) 95% confidence bands

Partial Autocorrelation Function (PACF)

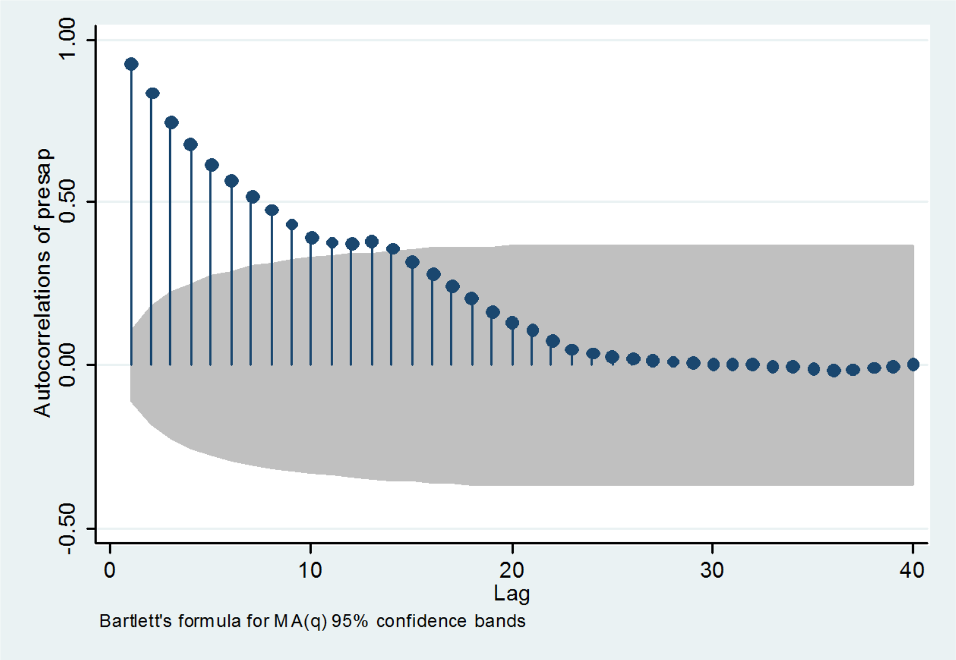
- The exclusive correlation coefficient
- Partial regression coefficient - The lag k partial autocorrelation is the partial regression coefficient, θ_{kk} in the k^{th} order auto regression
- In general, the "partial" correlation between two variables is the amount of correlation between them which is not explained by their mutual correlations with a specified set of other variables.
- For example, if we are regressing a variable Y on other variables X1, X2, and X3, the partial correlation between Y and X3 is the amount of correlation between Y and X3 that is not explained by their common correlations with X1 and X2.
- $y_t = \theta_{k1}y_{t-1} + \theta_{k2}y_{t-2} + \ldots + \theta_{kk}y_{t-k} + \varepsilon_t$
- **Partial correlation** measures the degree of association between two random variables, with the effect of a set of controlling random variables removed.

PACF Graph



Identification of AR Processes & its order -p

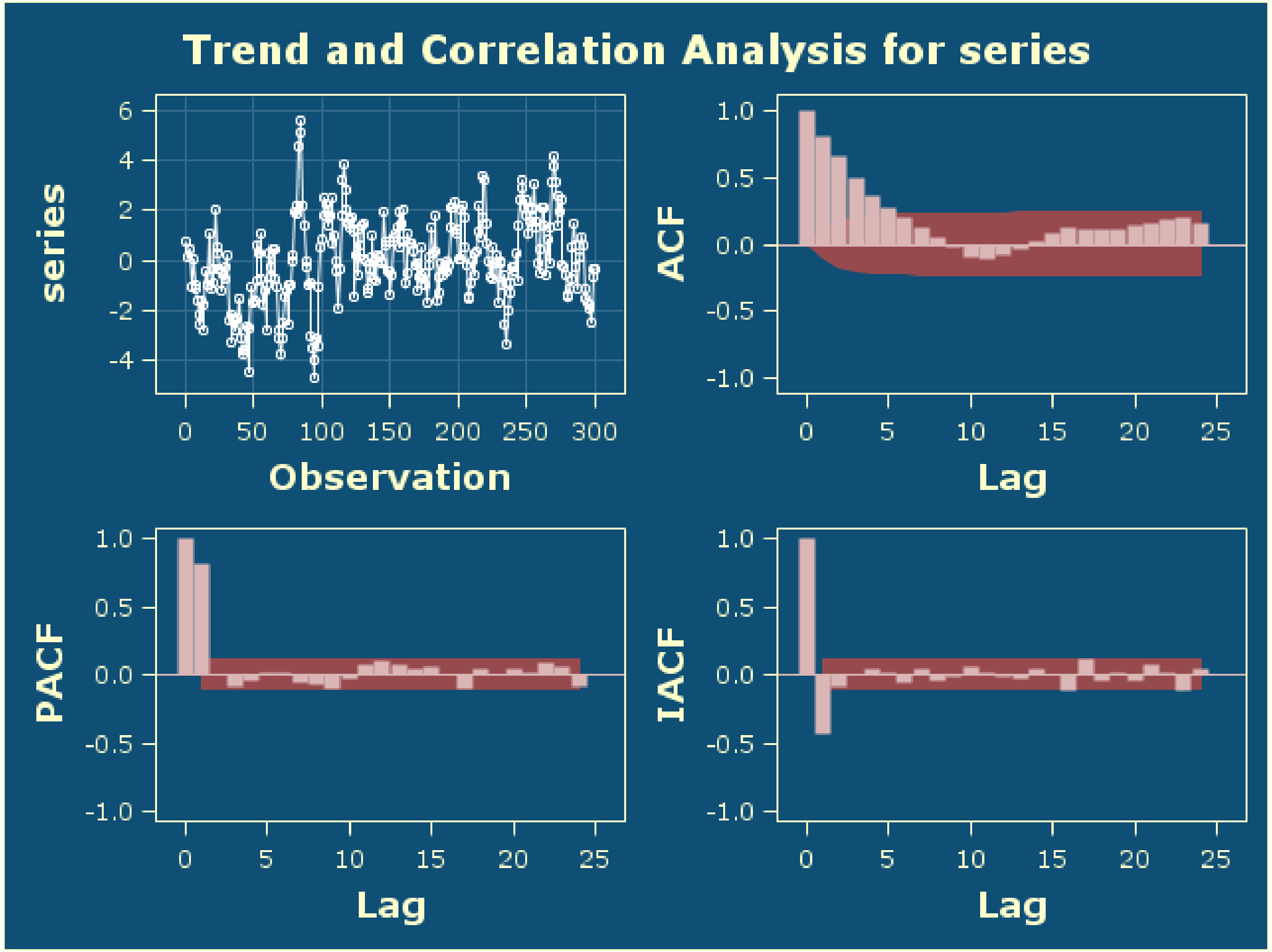
- For AR models, the ACF will dampen exponentially
- The PACF will identify the order of the AR model:
  - The AR(1) model ($y_t = a_1 y_{t-1} + \varepsilon_t$) would have one significant spike at lag 1 on the PACF.
  - The AR(3) model ($y_t = a_1 y_{t-1} + a_2 y_{t-2} + a_3 y_{t-3} + \varepsilon_t$) would have significant spikes on the PACF at lags 1, 2, & 3.



statinfer.com

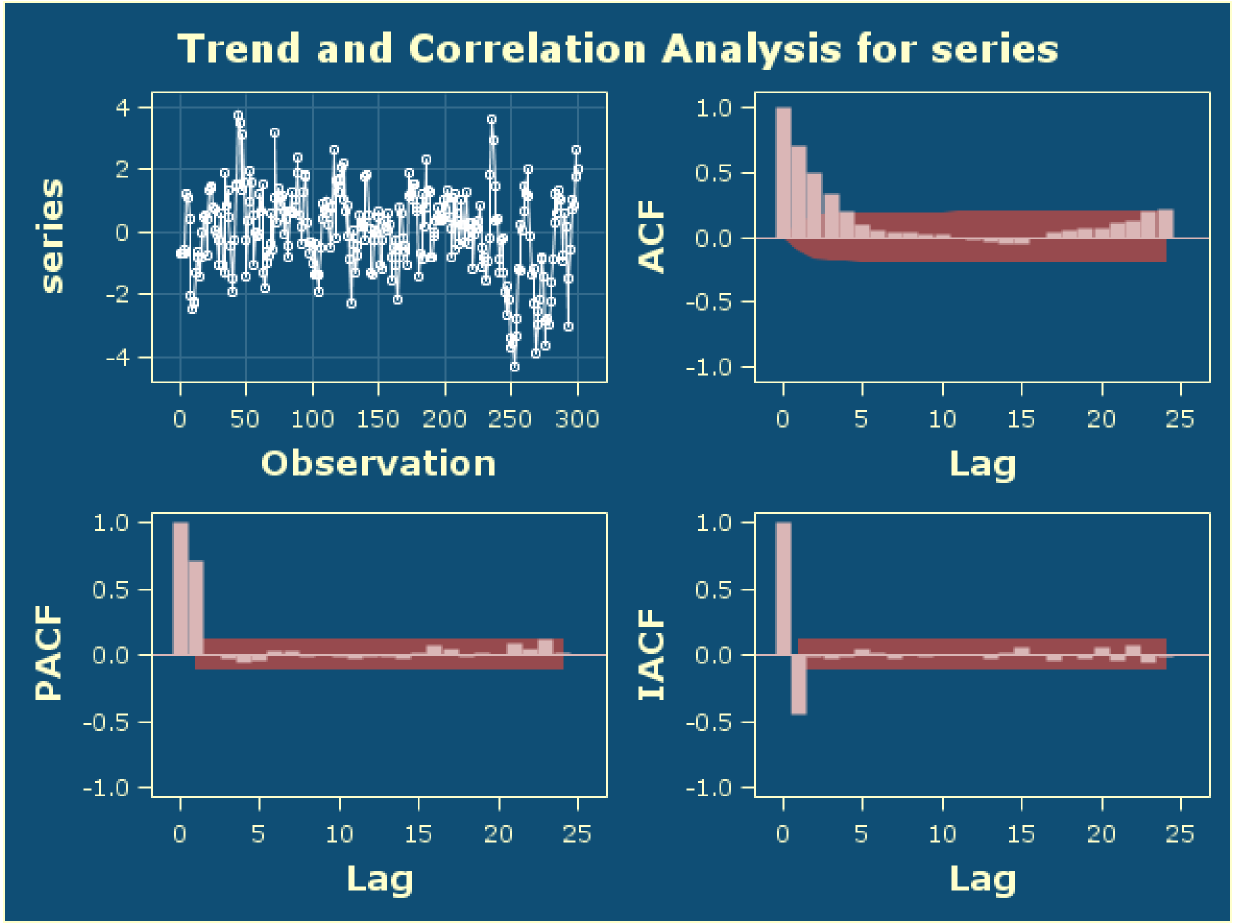
AR(1) model

$$y_t = 0.8y_{t-1} + \varepsilon_t$$



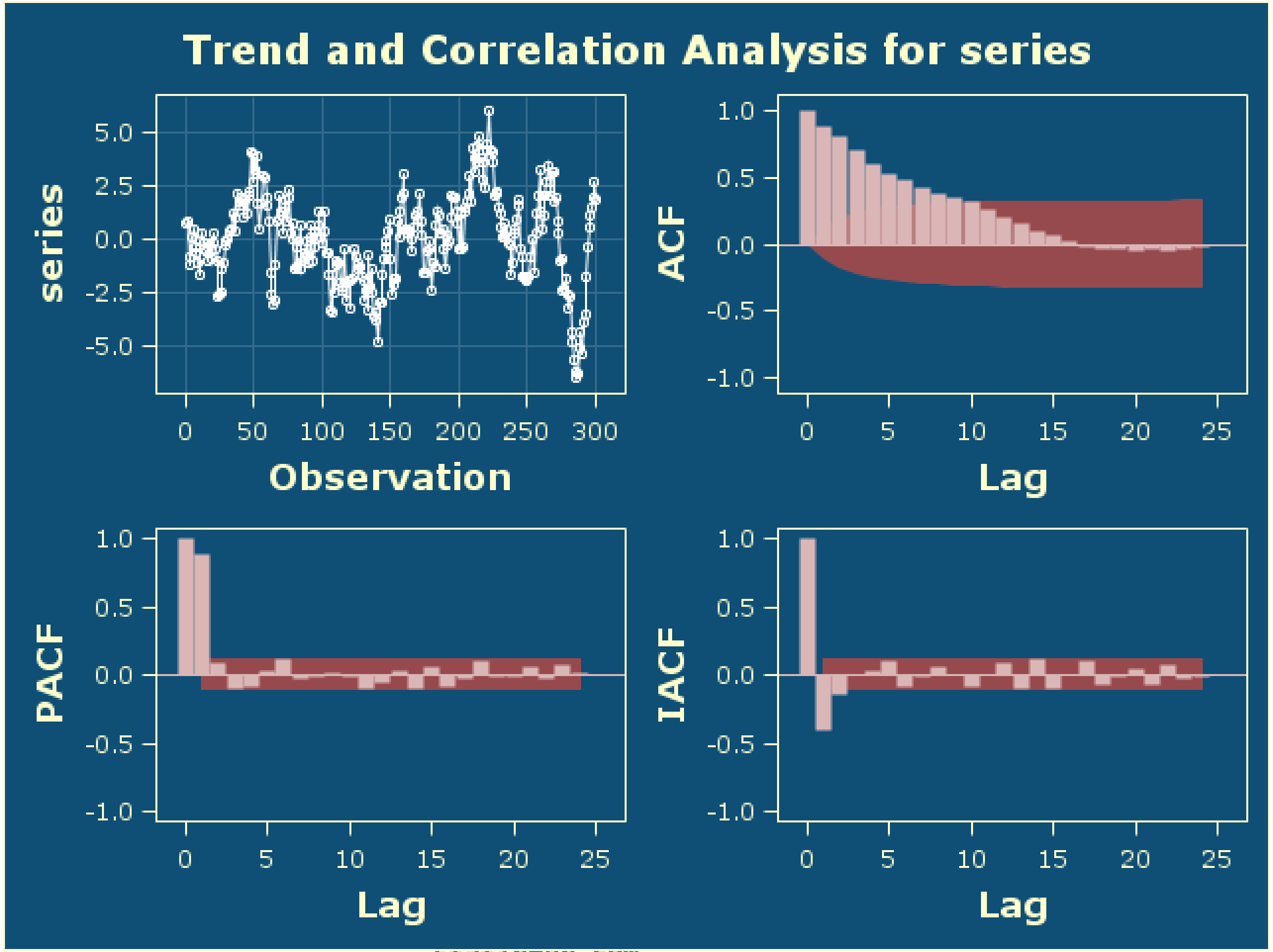
AR(1) model

$$y_t = 0.77y_{t-1} + \varepsilon_t$$



AR(1) model

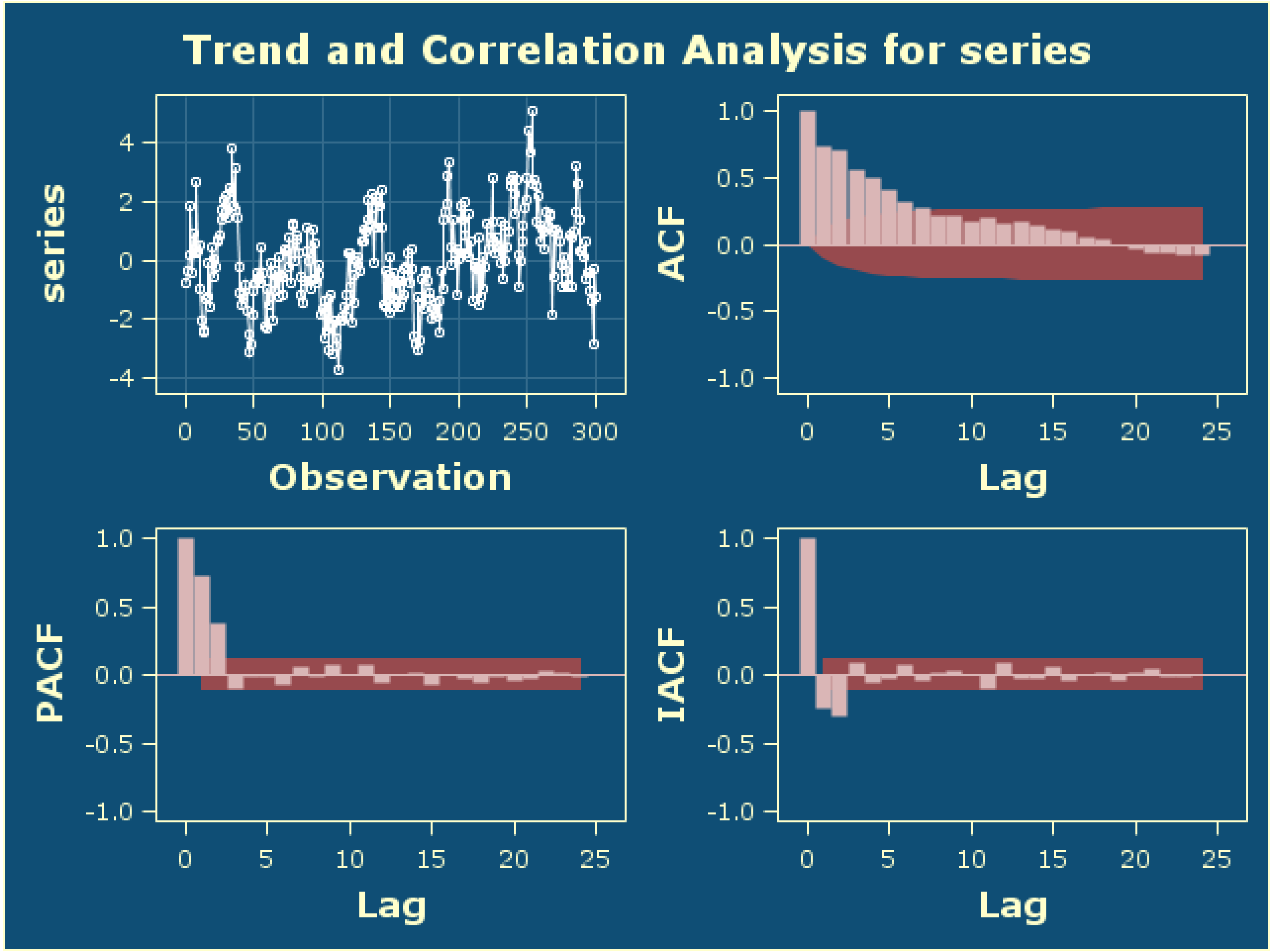
$$y_t = 0.95y_{t-1} + \varepsilon_t$$



statinfer.com

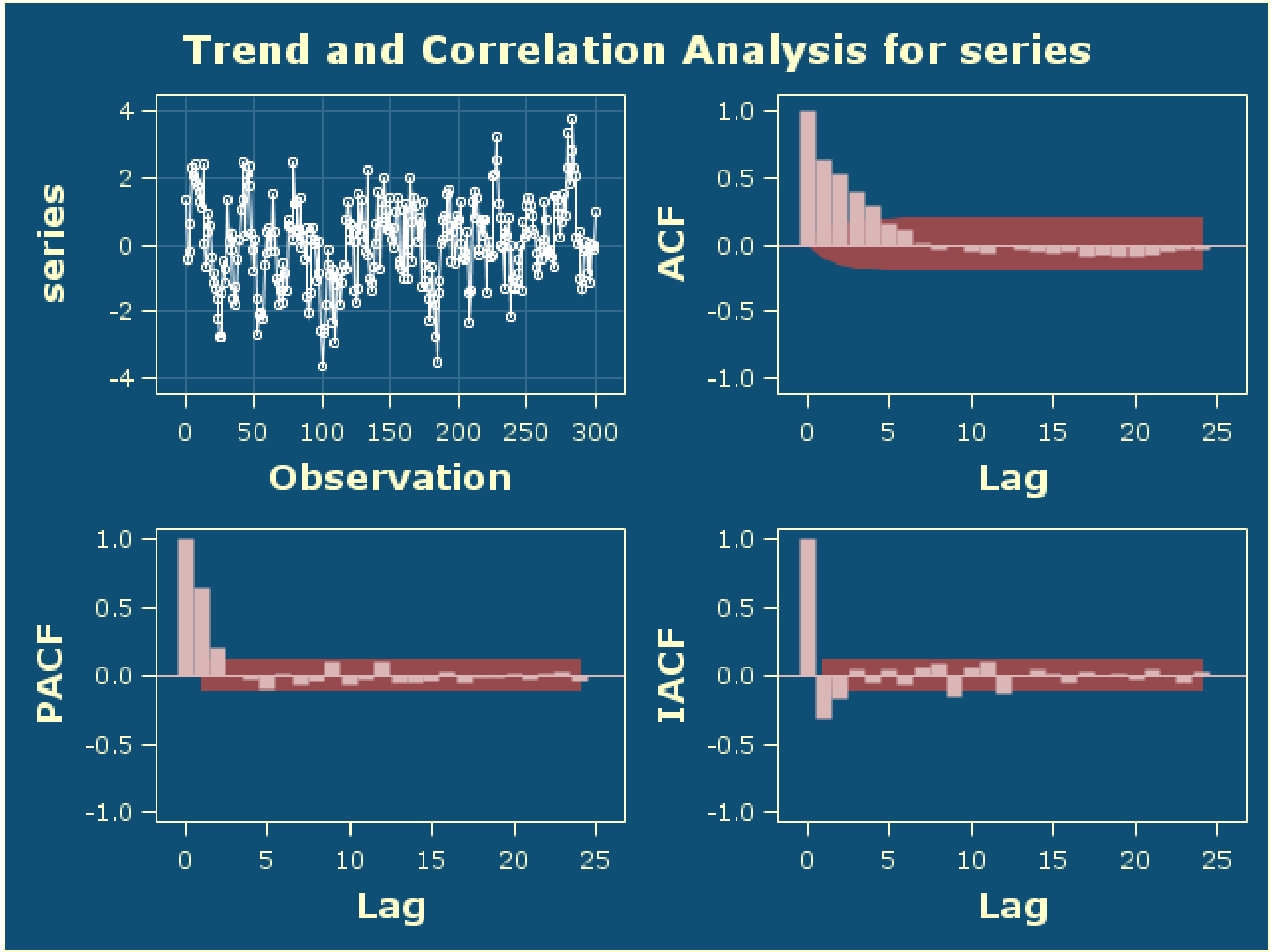
AR(2) model

$$y_t = 0.44y_{t-1} + 0.4y_{t-2} + \varepsilon_t$$



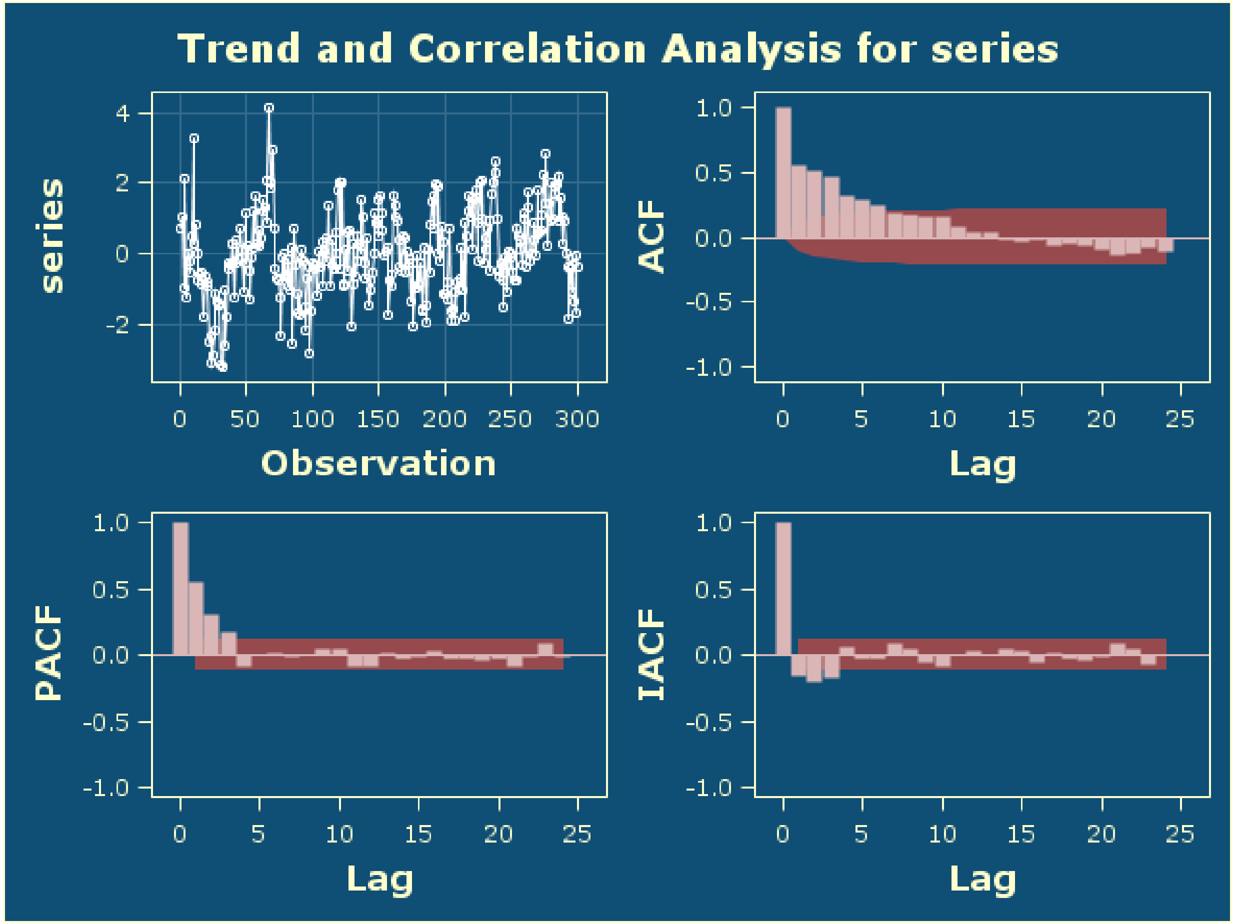
AR(2) model

$$y_t = 0.5y_{t-1} + 0.2y_{t-2} + \varepsilon_t$$



AR(3) model

$$y_t = 0.3y_{t-1} + 0.3y_{t-2} + 0.1y_{t-3} + \varepsilon_t$$



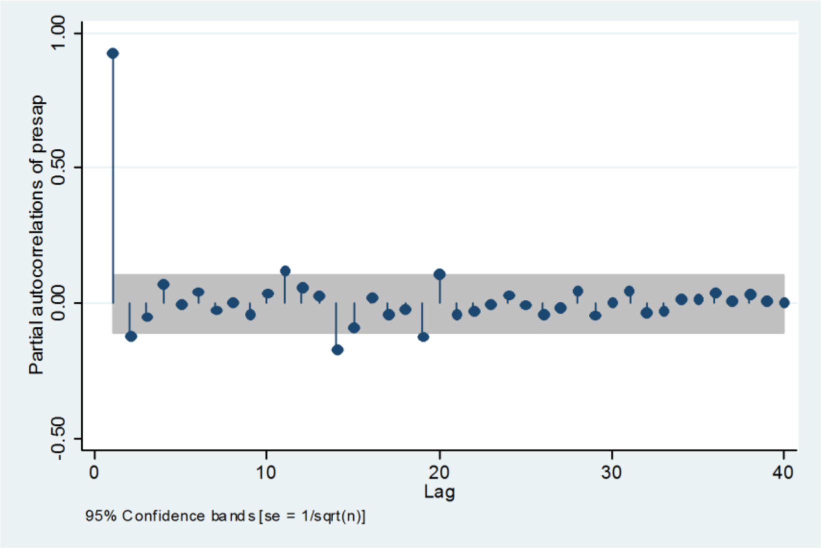
Once again

Properties of the ACF and PACF of MA, AR and ARMA Series

Process	MA(q)	AR(p)	ARMA(p,q)
Auto-correlation function	Cuts off	Infinite. Tails off. Damped Exponentials and/or Cosine waves	Infinite. Tails off. Damped Exponentials and/or Cosine waves after q-p.
Partial Autocorrelation function	Infinite. Tails off. Dominated by damped Exponentials & Cosine waves.	Cuts off	Infinite. Tails off. Dominated by damped Exponentials & Cosine waves after p-q.

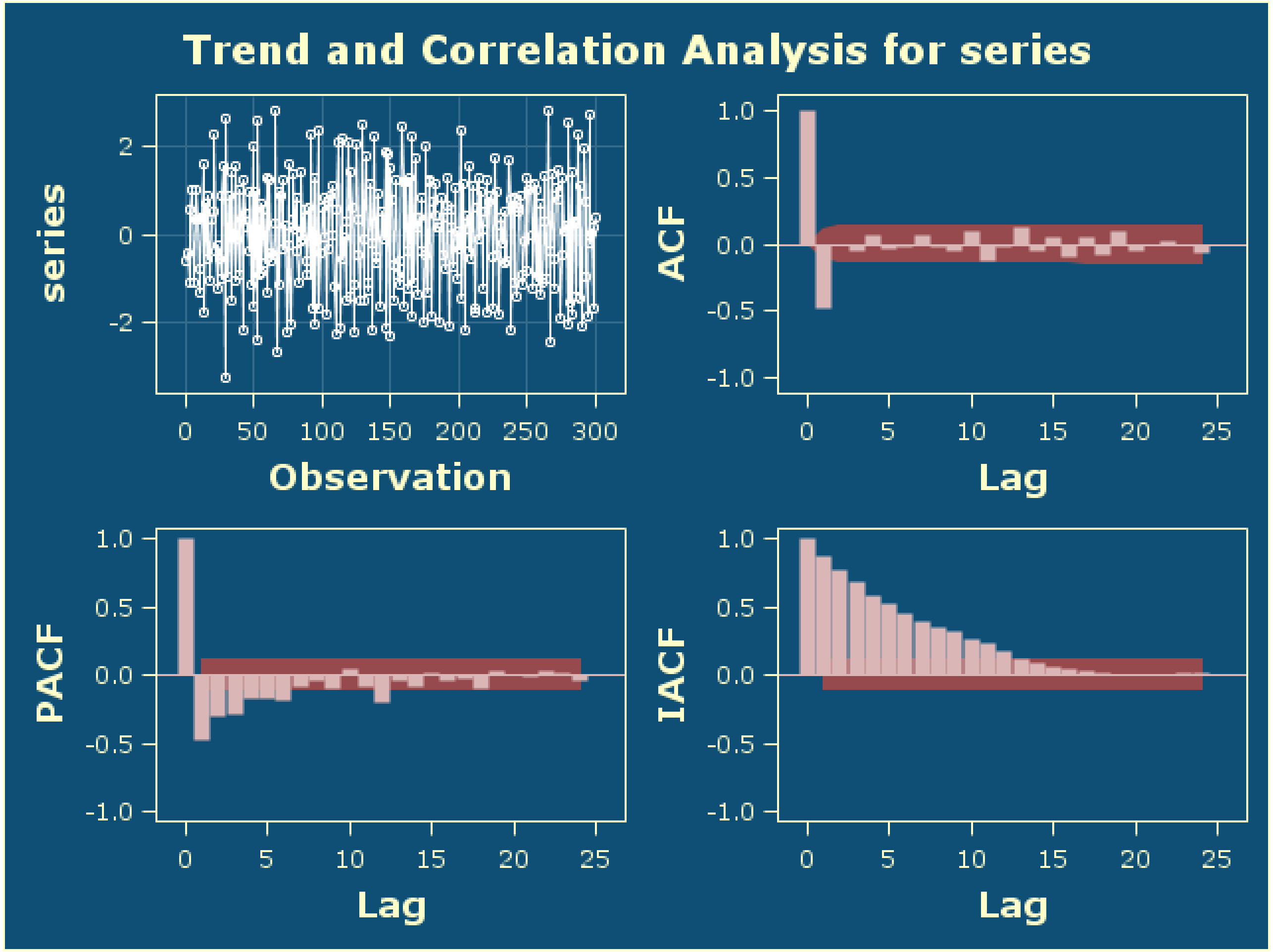
Identification of MA Processes & its order - q

- Recall that a MA(q) can be represented as an AR(∞), thus we expect the opposite patterns for MA processes.
- The PACF will dampen exponentially.
- The ACF will be used to identify the order of the MA process.
- MA(1) ($y_t = \varepsilon_t + b_1 \varepsilon_{t-1}$) has one significant spike in the ACF at lag 1.
- MA (3) ($y_t = \varepsilon_t + b_1 \varepsilon_{t-1} + b_2 \varepsilon_{t-2} + b_3 \varepsilon_{t-3}$) has three significant spikes in the ACF at lags 1, 2, & 3.



MA(1)

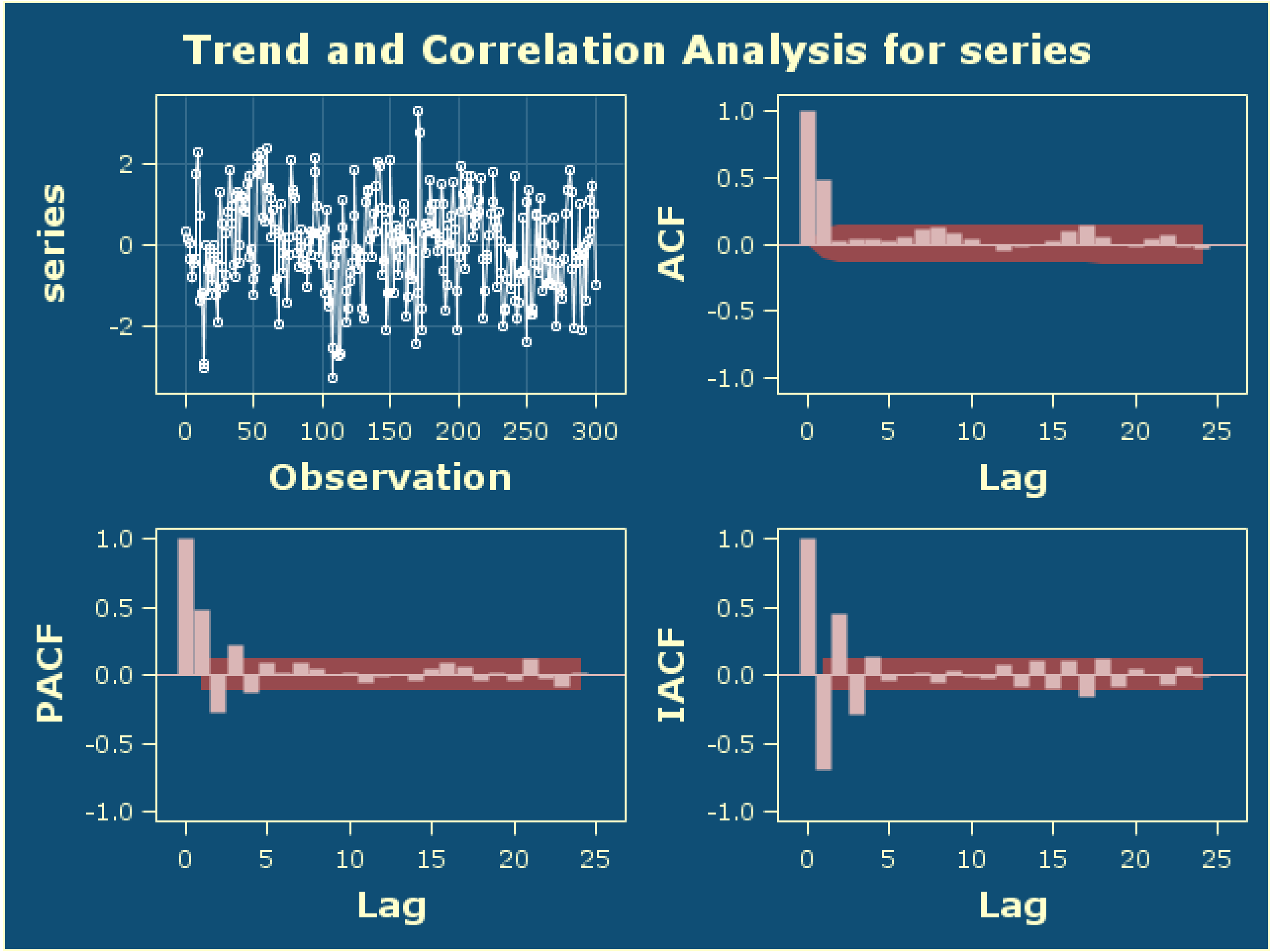
$y_t = -0.9\varepsilon_{t-1}$



statinfer.com

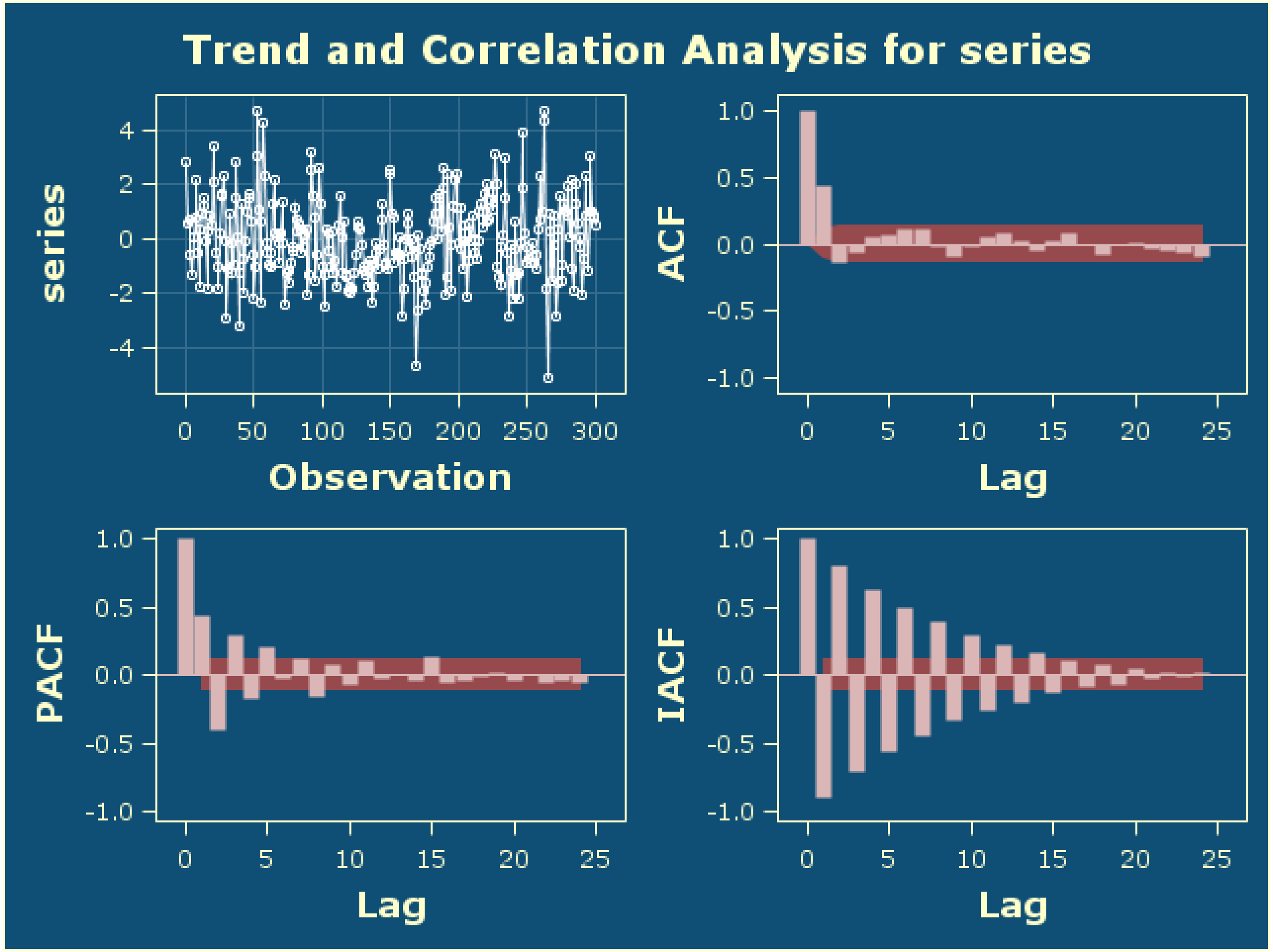
MA(1)

$$y_t = 0.7\varepsilon_{t-1}$$



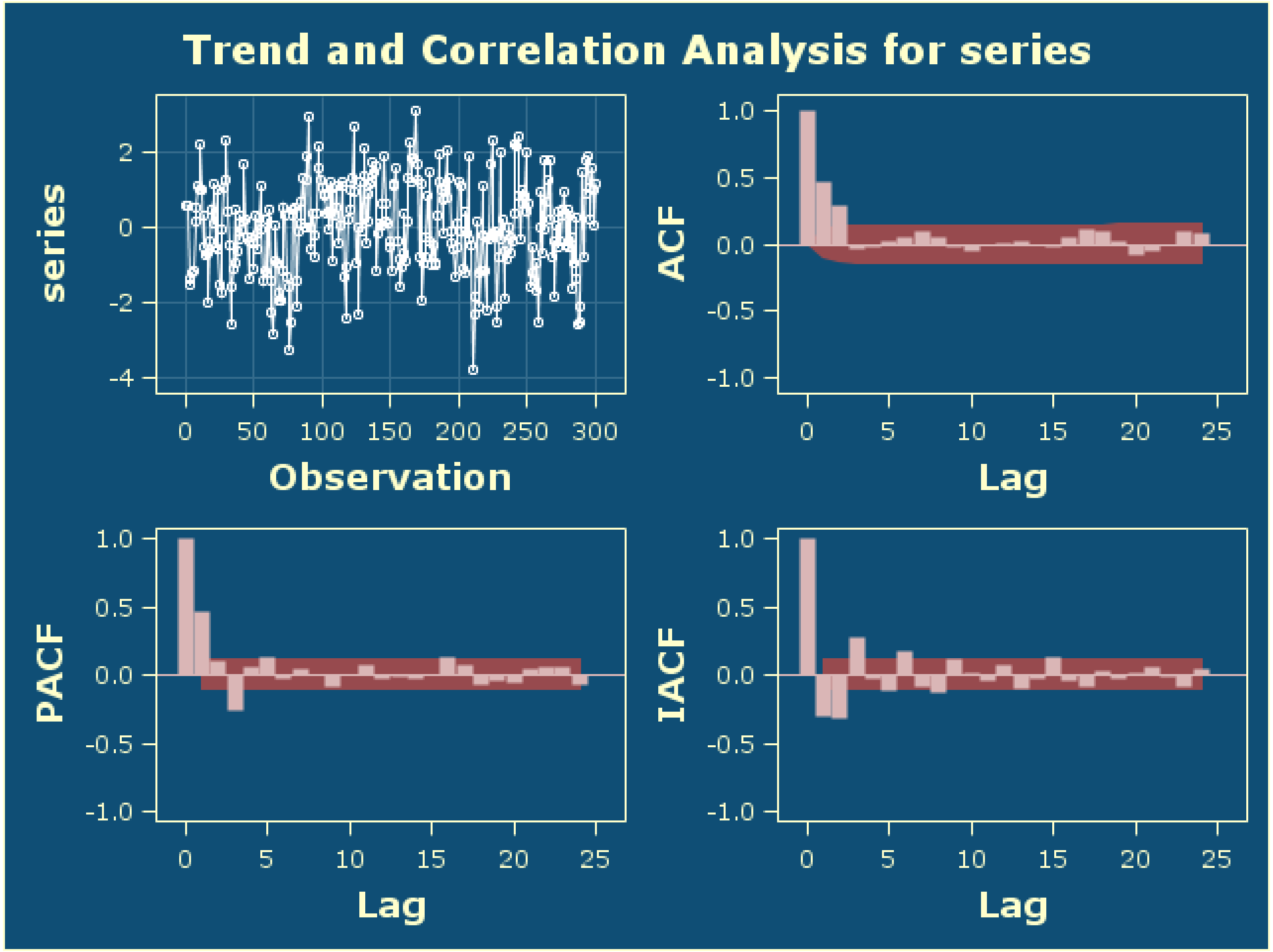
MA(1)

$y_t = 0.99\varepsilon_{t-1}$



MA(2)

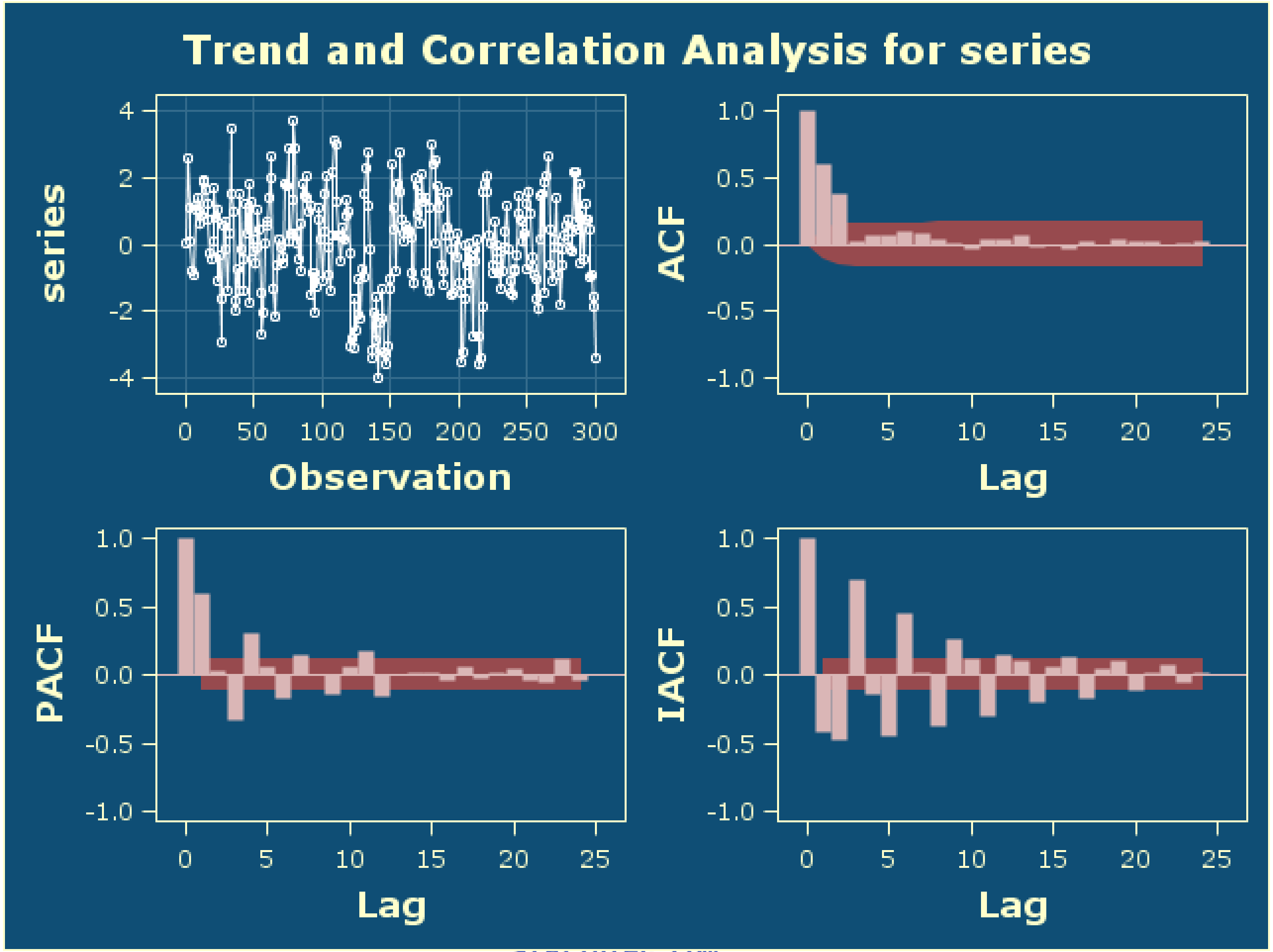
$y_t = 0.5\varepsilon_{t-1} + 0.5\varepsilon_{t-2}$



statinfer.com

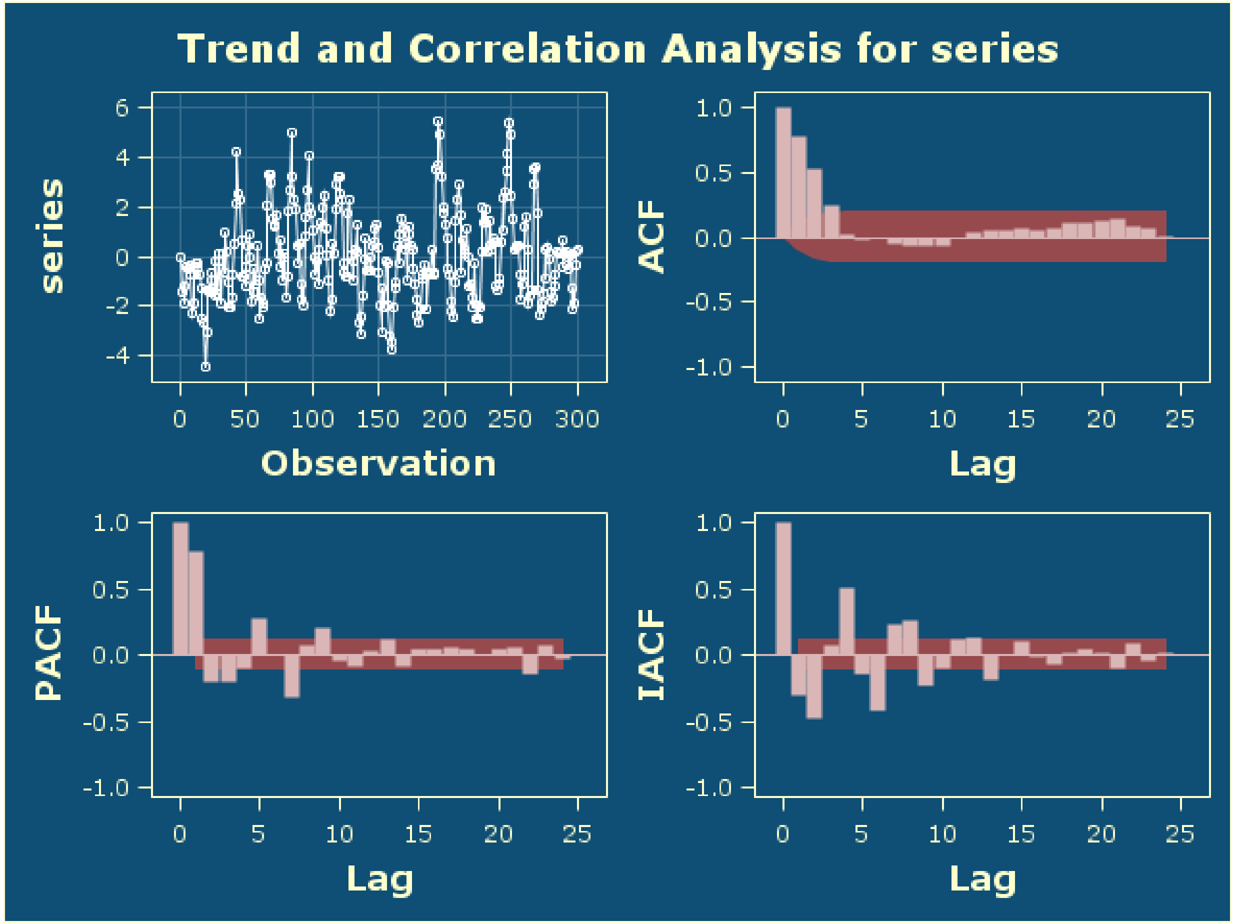
MA(2)

$$y_t = 0.8\varepsilon_{t-1} + 0.9\varepsilon_{t-2}$$



MA(3)

$$y_t = 0.8\varepsilon_{t-1} + 0.9\varepsilon_{t-2} + 0.6\varepsilon_{t-3}$$



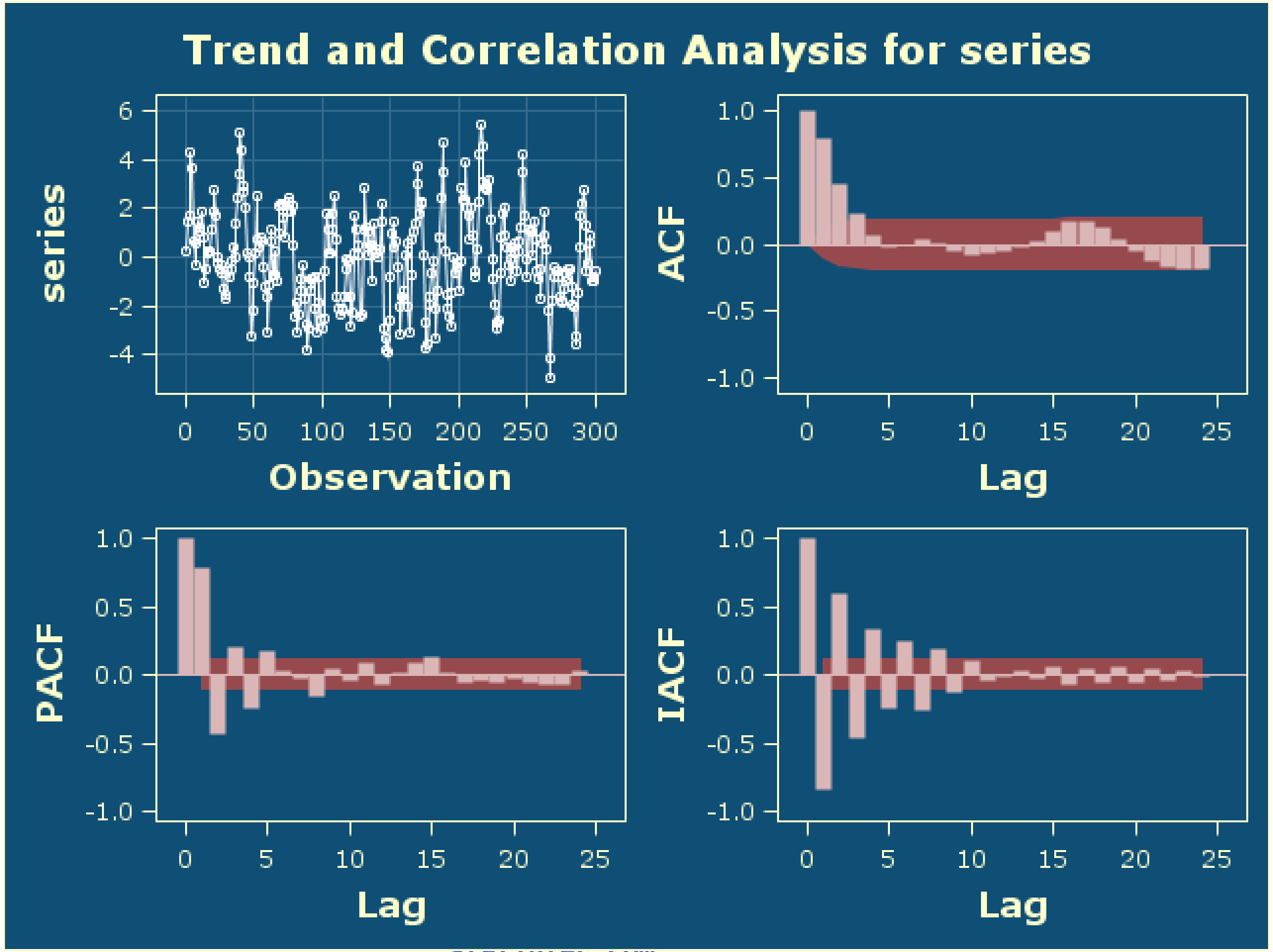
Once again

Properties of the ACF and PACF of MA, AR and ARMA Series

Process	MA(q)	AR(p)	ARMA(p,q)
Auto-correlation function	Cuts off	Infinite. Tails off. Damped Exponentials and/or Cosine waves	Infinite. Tails off. Damped Exponentials and/or Cosine waves after q-p.
Partial Autocorrelation function	Infinite. Tails off. Dominated by damped Exponentials & Cosine waves.	Cuts off	Infinite. Tails off. Dominated by damped Exponentials & Cosine waves after p-q.

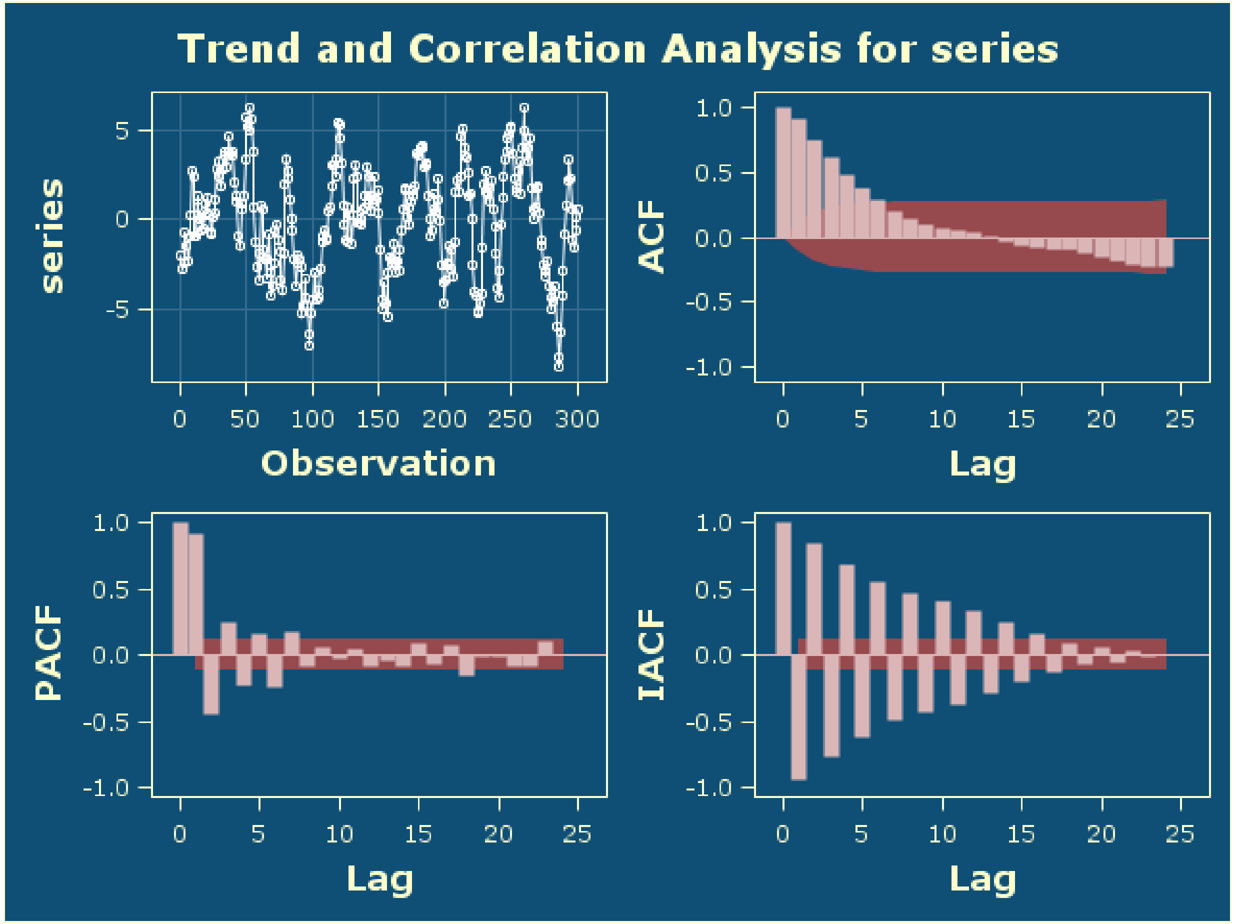
ARMA(1,1)

$$y_t = 0.6y_{t-1} + 0.8\varepsilon_{t-1}$$



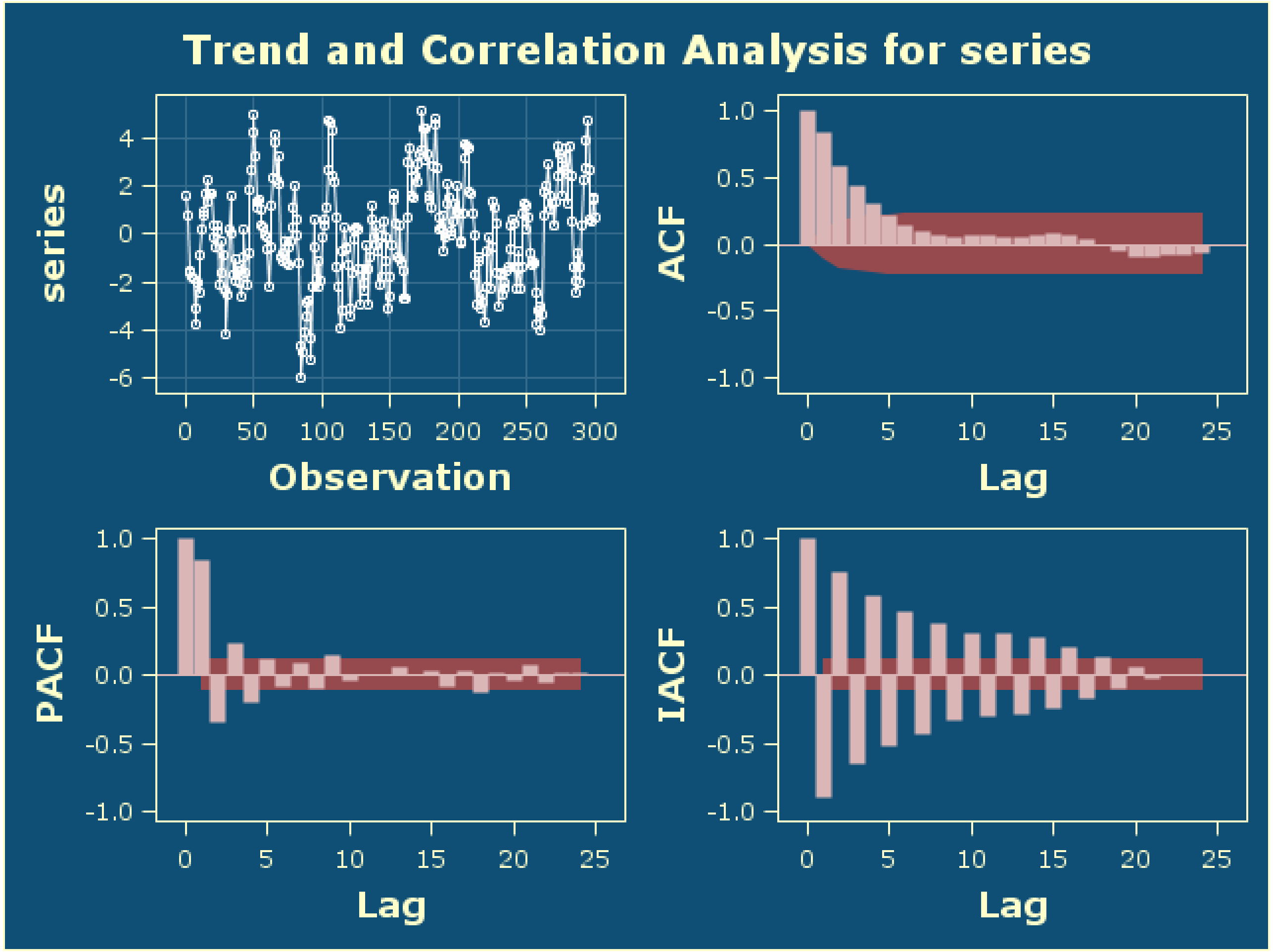
ARMA(1,1)

$$y_t = 0.78y_{t-1} + 0.9\varepsilon_{t-1}$$



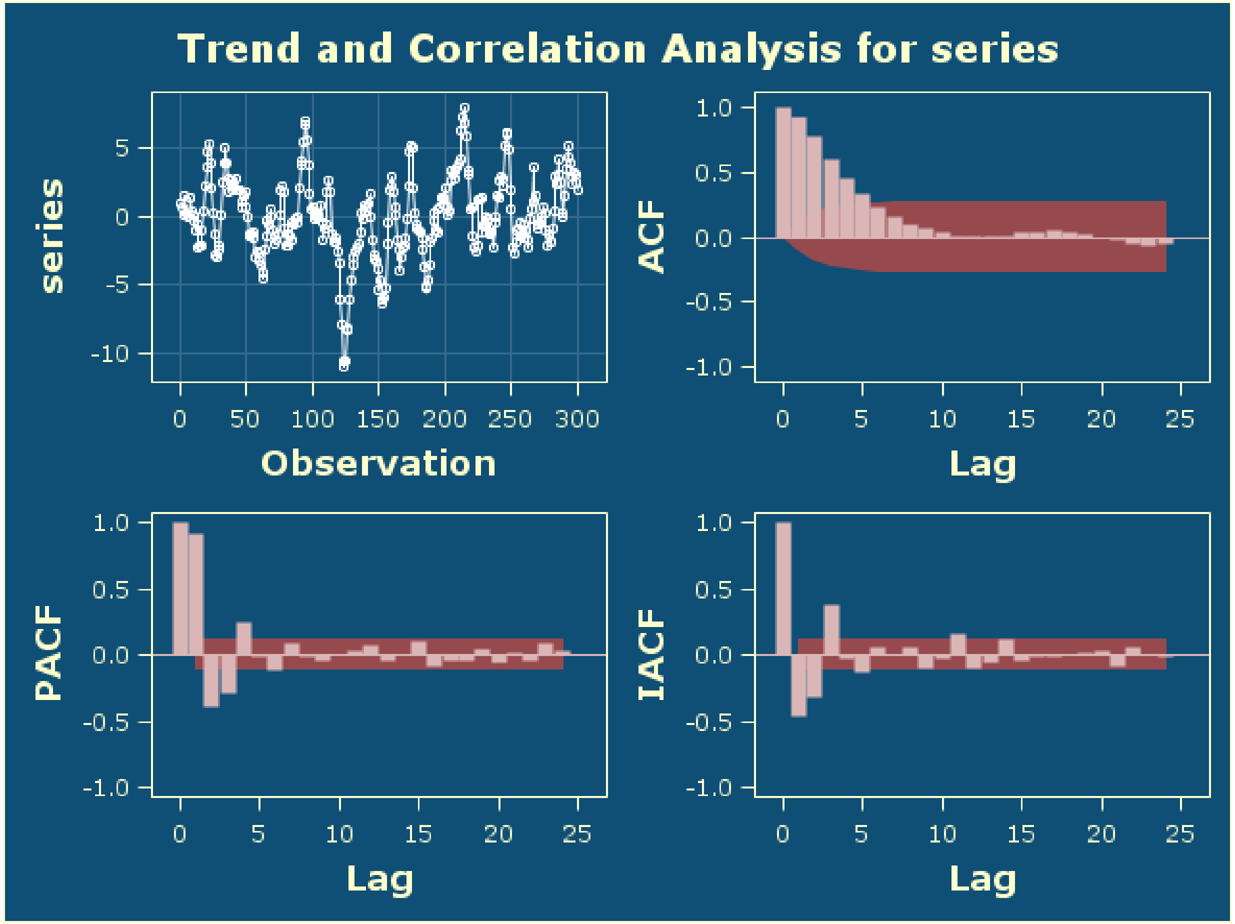
ARIMA(2,1)

$$y_t = 0.4y_{t-1} + 0.3y_{t-2} + 0.9\varepsilon_{t-1}$$



ARMA(1,2)

$$y_t = 0.8y_{t-1} + 0.4\varepsilon_{t-1} + 0.55\varepsilon_{t-2}$$



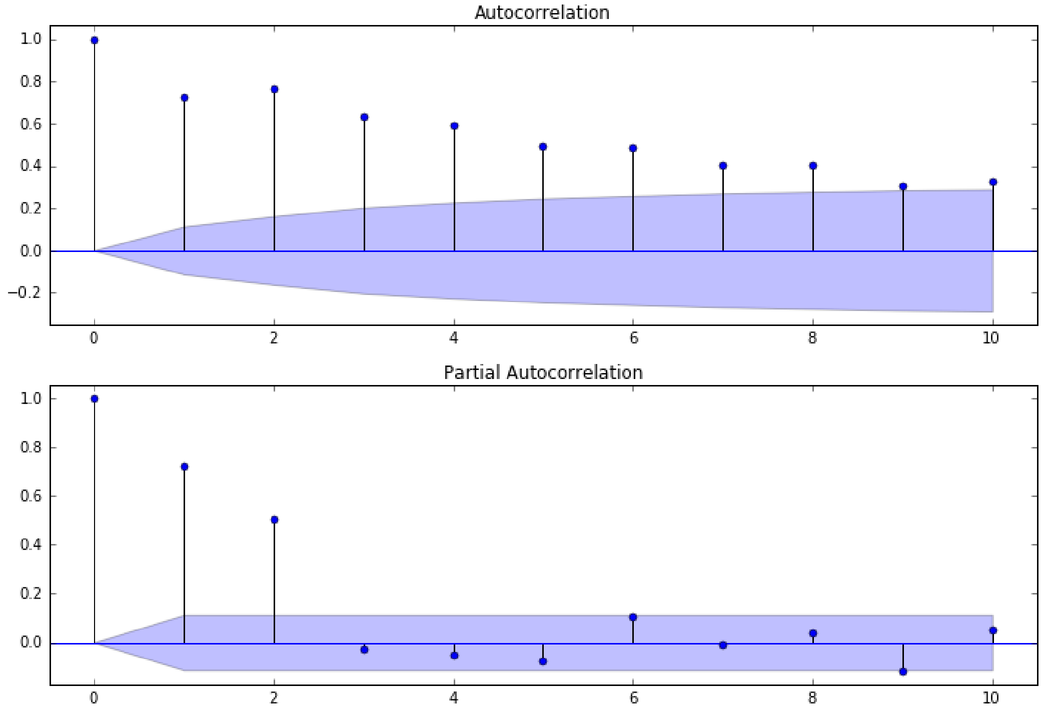
ARMA Model Identification

Properties of the ACF and PACF of MA, AR and ARMA Series

Process	MA(q)	AR(p)	ARMA(p,q)
Auto-correlation function	Cuts off	Infinite. Tails off. Damped Exponentials and/or Cosine waves	Infinite. Tails off. Damped Exponentials and/or Cosine waves after q-p.
Partial Autocorrelation function	Infinite. Tails off. Dominated by damped Exponentials & Cosine waves.	Cuts off	Infinite. Tails off. Dominated by damped Exponentials & Cosine waves after p-q.

Demo1: Identification of the model

```
#Plotting
plt.plot(Call_volume_ts)

#First lets test stationarity
test_stationarity(Call_volume_ts)

#Plot ACF and PACF
import statsmodels.api as sm

fig = plt.figure(figsize=(12,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(Call_volume_ts, lags=10, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(Call_volume_ts, lags=10,
ax=ax2)
```



- ACF is dampening, PCF graph cuts off. - Perfect example of an AR process

statinfer

LAB: Identification of model

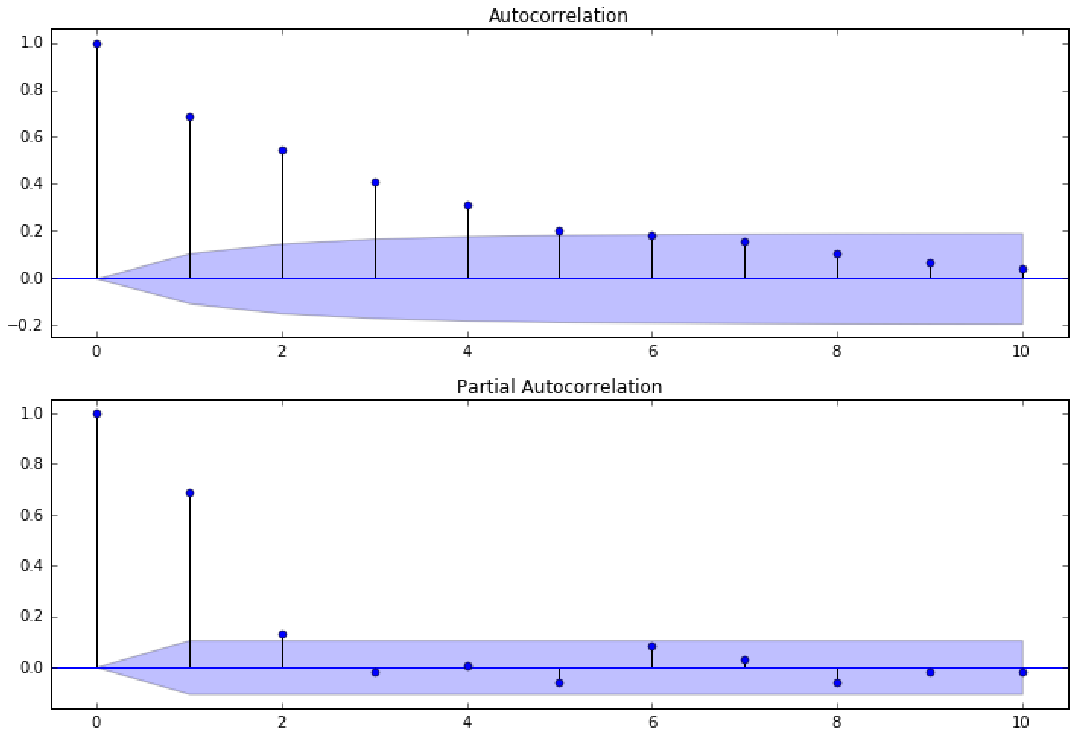
- Download web views data
- What does ACF & PACF graphs say?
- Identify the model using below table
- Write the model equation

Properties of the ACF and PACF of MA, AR and ARMA Series

Process	MA(q)	AR(p)	ARMA(p,q)
Auto-correlation function	Cuts off	Infinite. Tails off. Damped Exponentials and/or Cosine waves	Infinite. Tails off. Damped Exponentials and/or Cosine waves after q-p.
Partial Autocorrelation function	Infinite. Tails off. Dominated by damped Exponentials & Cosine waves.	Cuts off	Infinite. Tails off. Dominated by damped Exponentials & Cosine waves after p-q.

Code: Identification of model

```
#First lets test stationarity
test_stationarity(web_views)

#Plot ACF and PACF
import statsmodels.api as sm

fig = plt.figure(figsize=(12,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(web_views, lags=10, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(web_views, lags=10, ax=ax2)
```



Step3 : Estimation

Parameter Estimate

- We already know the model equation. AR(1,0,0) or AR(2,1,0) or ARIMA(2,1,1)
- We need to estimate the coefficients using Least squares. Minimizing the sum of squares of deviations

$$min \quad \sum_t \varepsilon_t^2$$

$$min \quad \sum_{t=2}^{T} (y_t - \phi y_{t-1})^2$$

Demo1: Parameter Estimation

- Call Volumes data

```
from statsmodels.tsa.arima_model import ARIMA
import statsmodels as sm
model = ARIMA(Call_volume_ts,order=(2, 0, 0))
results_AR = model.fit()
results_AR.summary()

plt.title('Fitted Vs Actual Values')
plt.plot(Call_volume_ts)
plt.plot(results_AR.fittedvalues, color='red')
```

```
                              ARMA Model Results
==============================================================================
Dep. Variable:             Calls_in_mm   No. Observations:                  300
Model:                      ARMA(2, 0)   Log Likelihood                -447.912
Method:                        css-mle   S.D. of innovations              1.075
Date:                 Mon, 20 Feb 2017   AIC                            903.823
Time:                         21:35:40   BIC                            918.638
Sample:                     01-01-2012   HQIC                           909.752
                          - 10-26-2012
=====================================================================================
                        coef    std err          z      P>|z|      [95.0% Conf. Int.]
-------------------------------------------------------------------------------------
const                 9.4602      0.431     21.969      0.000         8.616    10.304
ar.L1.Calls_in_mm     0.3549      0.049      7.170      0.000         0.258     0.452
ar.L2.Calls_in_mm     0.5055      0.050     10.208      0.000         0.408     0.603
                                    Roots
=============================================================================
                 Real           Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1           1.0986           +0.0000j            1.0986            0.0000
AR.2          -1.8006           +0.0000j            1.8006            0.5000
```

Lab: Parameter Estimation

- Estimate the parameters for webview data

Code: Parameter Estimation

```
from statsmodels.tsa.arima_model import ARIMA
import statsmodels as sm
model1 = ARIMA(web_views,order=(1, 0, 0))
results_AR1 = model1.fit()
results_AR1.summary()

plt.plot(web_views)
plt.plot(results_AR1.fittedvalues, color='red'
plt.title('Fitted Vs Actual Values')
```

```
                              ARMA Model Results
==============================================================================
Dep. Variable:                  views   No. Observations:                  340
Model:                     ARMA(1, 0)   Log Likelihood                 307.284
Method:                       css-mle   S.D. of innovations              0.098
Date:                Mon, 20 Feb 2017   AIC                           -608.568
Time:                        21:43:50   BIC                           -597.081
Sample:                    01-01-2016   HQIC                          -603.991
                         - 12-05-2016
===============================================================================
                  coef    std err          z      P>|z|      [95.0% Conf. Int.]
-------------------------------------------------------------------------------
const           0.9937      0.017     58.810      0.000         0.961     1.027
ar.L1.views     0.6877      0.039     17.530      0.000         0.611     0.765
                                    Roots
=============================================================================
                 Real           Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1           1.4541           +0.0000j            1.4541            0.0000
-----------------------------------------------------------------------------
"""
```

Step4 : Forecasting

Forecasting

- Now the model is ready
- We simply need to use this model for forecasting

```
results_AR.forecast(steps=10)
#Returns: forecast : array
#Array of out of sample forecasts stderr : array
#Array of the standard error of the forecasts.
#conf_int : array 2d array of the confidence interval for the forecast
```

```
(array([ 8.88847212,  9.19381543,  9.07664097,  9.18941593,  9.1702036 ,
         9.22039592,  9.22849616,  9.25674426,  9.27086398,  9.29015497]),
 array([ 1.07455887,  1.14021984,  1.32685014,  1.39589728,  1.4817005 ,
         1.53358436,  1.58278296,  1.61866059,  1.64964397,  1.67395587]),
 array([[  6.78237543,  10.99456881],
        [  6.95902561,  11.42860525],
        [  6.47606249,  11.67721945],
        [  6.45350754,  11.92532431],
        [  6.26612399,  12.07428321],
        [  6.2146258 ,  12.22616604],
        [  6.12629857,  12.33069375],
        [  6.08422781,  12.42926071],
        [  6.03762122,  12.50410674],
        [  6.00926176,  12.57104819]]))
```

LAB: Forecasting using ARIMA

- Forecast the number of sunspots for next three hours

```
results_AR1.forecast(steps=10)
```

Goodness of fit

- Remember… **Residual analysis** and Mean deviation, Mean Absolute Deviation and Root Mean Square errors?
- Four common techniques are the:
  - Mean absolute deviation,
    $$\text{MAD} = \sum_{i=1}^{n} \frac{\left|Y_i - \hat{Y}_i\right|}{n}$$
  - Mean absolute percent error
    $$\text{MAPE} = \frac{100}{n} \sum_{i=1}^{n} \frac{\left|Y_i - \hat{Y}_i\right|}{Y_i}$$
  - Mean square error,
    $$\text{MSE} = \sum_{i=1}^{n} \frac{\left(Y_i - \hat{Y}_i\right)^2}{n}$$
  - Root mean square error.
    $$\text{RMSE} = \sqrt{\text{MSE}}$$

statinfer

Validation: How good is my model?

- Does our model really give an adequate description of the data
- Two criteria to check the goodness of fit
  - Akaike information criterion (AIC)
  - Schwartz Bayesiancriterion (SBC)/Bayesian information criterion (BIC).
- These two measures are useful in comparing two models.
- The smaller the AIC & SBC the better the model

Lab: Overall Steps on sunspot example

- Import the time series data
- Prepare the data for model building- Make it stationary
- Identify the model type
- Estimate the parameters
- Forecast the future values

statinfer

Conclusion

Conclusion

- In this session we discussed basics of Box Jenkins methods of time series analysis and forecasting
- Time series model building involves multiple iterations. It is not easy to identify the best fit in one shot. We need to try different models to identify the best fit model
- We discussed ARIMA. We can even mix regression and build ARIMAX model. Which involves TS model and predictor TS

Thank you