

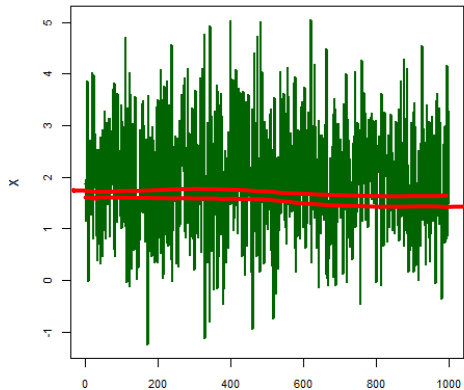
Elements of Forecasting

Stationarity, Differencing, and Order of Integration

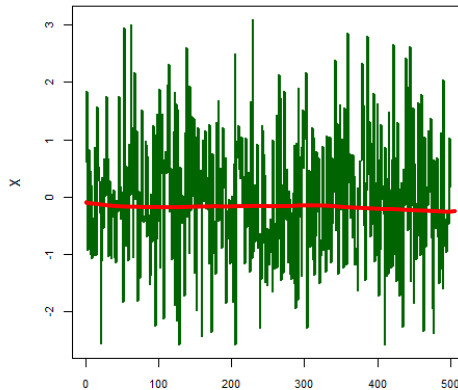
Dr. Sung Won Kim

“Which One is Stationary/Non-Stationary?”

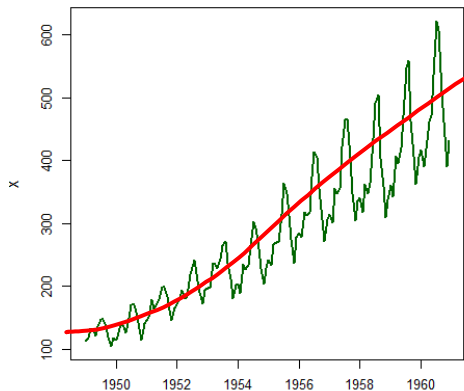
Time Series No.1



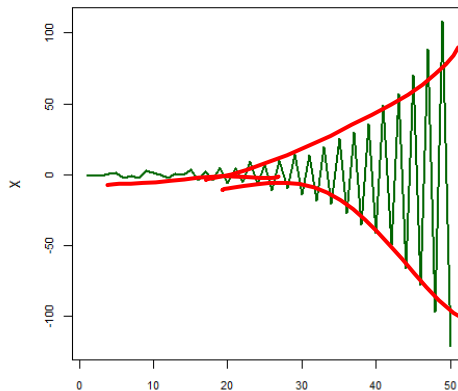
Time Series No.2



Time Series No.3



Time Series No.4



What Is Stationarity?

A time series is stationary when its statistical properties such as mean, variance, covariance, etc. remain unchanged over time.

Two Types of Stationarity

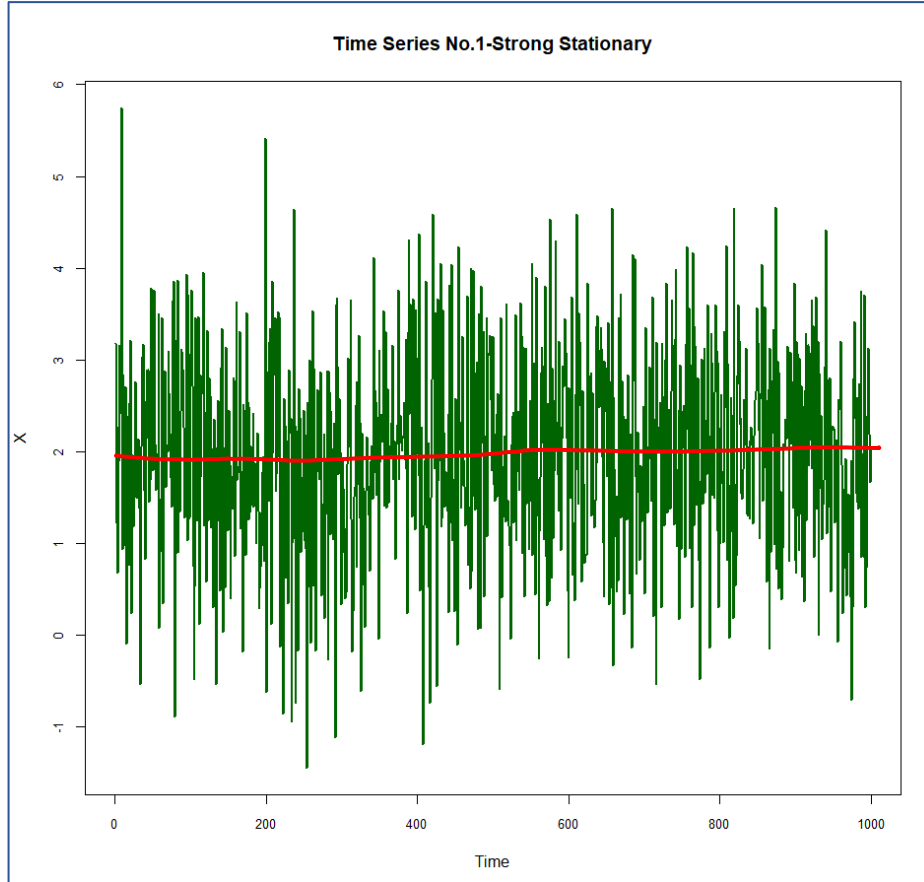
Strong stationarity

Weak stationarity

Strong (Strict-Sense) Stationarity

A time series has a strong-sense stationarity if its joint probability distribution remained unchanged when shifted along any time period.

Strong Stationarity Example



$$X = 2 + Y$$
$$Y \sim N(\mu = 0, \sigma^2 = 1)$$

Normal Distribution with
mean = 0, variance = 1

Weak (Wide-Sense) Stationarity

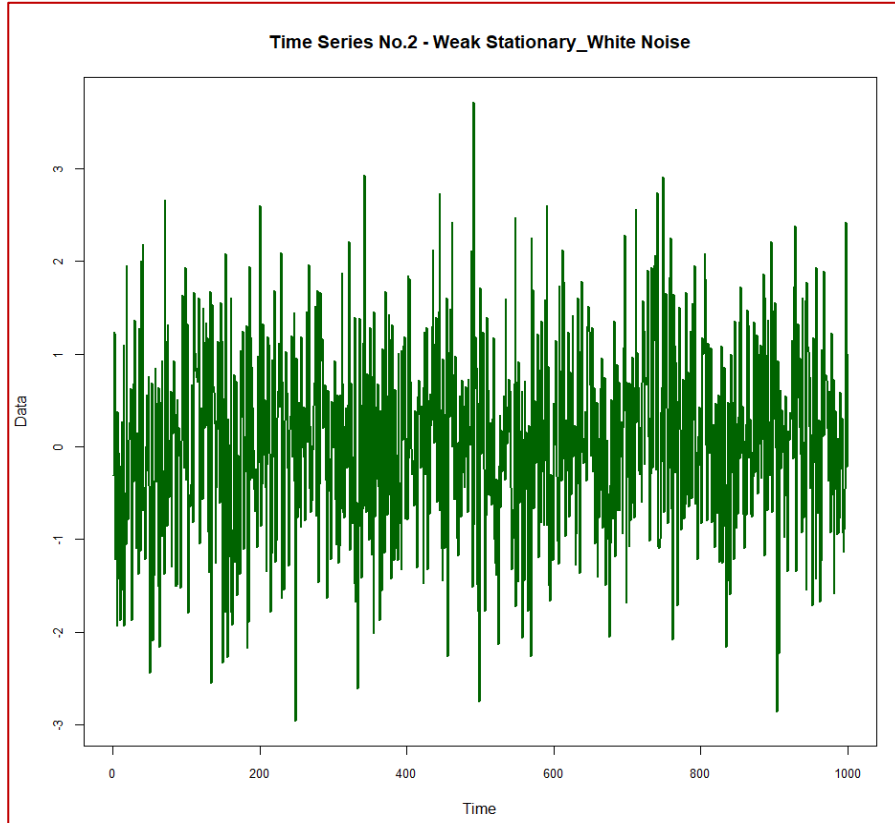
A time series has a weak-sense stationarity when it satisfies the three below conditions:

- Its mean doesn't vary over time.

- Its covariance doesn't vary over time.

- Its variance is finite.

Weak Stationarity Example



White noise is a sample of weak-sense stationary because:

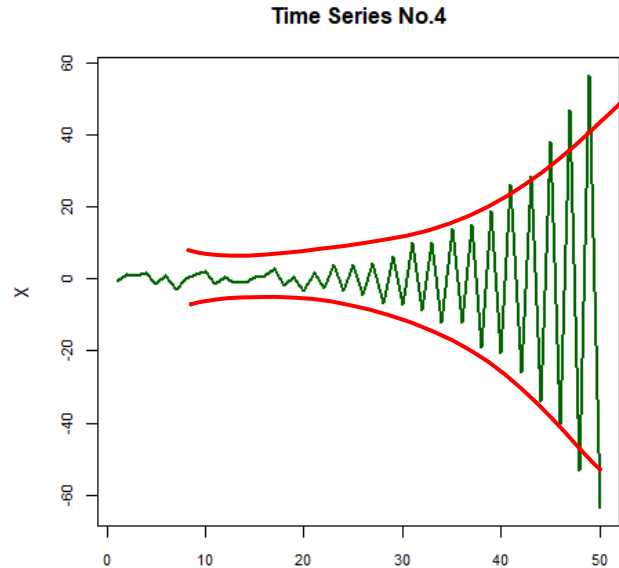
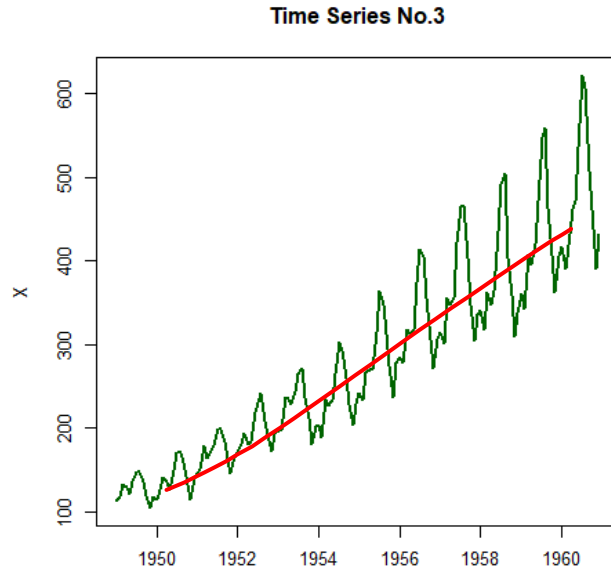
Its mean = 0

Its covariance = 0

Its variance is finite

Nonstationary Time Series

A time series with a trend or with a seasonality is nonstationary because its properties can be affected by a trend or a seasonality at some periods of time.



Why Do We Need Stationarity?

A stationary time series helps prediction become easier because its statistical properties in the future will be the same as they have been in the past.

Before building any forecasting model, we need to test for the stationarity of the time series.

Stationarity Testing

There are three main stationarity tests:

- Augmented Dickey-Fuller (ADF) Test

- KPSS-Test

- Phillips-Perron Test

Stationarity Testing Example

Let's test the stationarity of the Apple stock price series

Collected from Jan 3rd 2019 to Apr 30th 2019 (3 month period)

Source: Yahoo Finance website

Perform ADF testing for stationarity

Augumented Dickey-Fuller (ADF) Test

Null Hypothesis: Series is nonstationary.

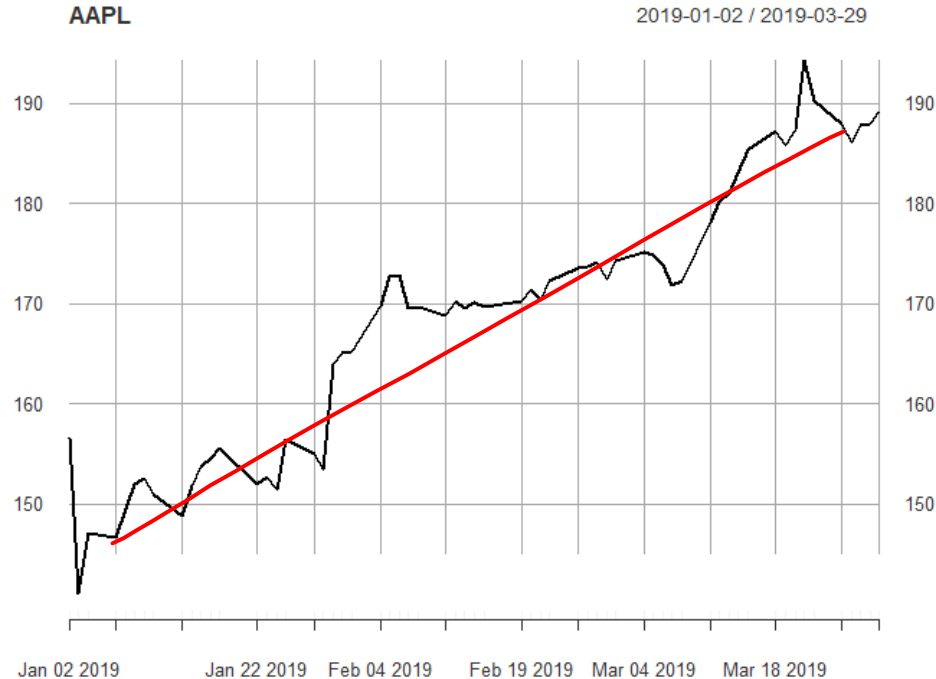
Reject Null Hypothesis: Series is stationary.

Fail to Reject: Series is nonstationary.

Dickey, David A., and Wayne A. Fuller (1981). "Likelihood ratio statistics for autoregressive time series with a unit root." *Econometrica: Journal of the Econometric Society*: 1057-1072

Mindy Mallory (2018). "Basic Time-Series Analysis, the Game." <http://blog.mindymallory.com/2018/01/basic-time-series-analysis-the-game/>

APPLE Price Series Stationarity Testing



Step 1: Plot the data to eye-check if it looks stationary or not.

Graph doesn't look stationary. Let's take a stationarity test to be confident on this observation.

APPLE Price Series Stationarity Testing (cont.)

Step 2: Perform ADF test in R.

Code:

Stationarity Testing

- `adf=adf.test(AAPL)`
- `print(adf)`

```
Augmented Dickey-Fuller Test  
data: AAPL  
Dickey-Fuller = -2.4869, Lag order = 3, p-value = 0.3779  
alternative hypothesis: stationary
```

What Should We Do Next?

Most forecasting models require stationarity in data.

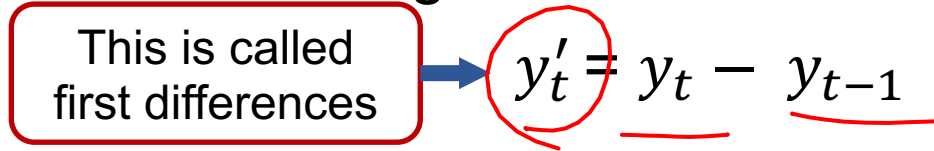
Since the Apple stock price series is not stationary, **differencing** can help!

Differencing

Differencing helps to stabilize the mean.

The differenced series is the change between each observation in the original series.

This is called first differences


$$y'_t = y_t - y_{t-1}$$

Where:

y'_t : first differences of the data

y_t : the data at time t

y_{t-1} : the data at time t-1

Second-Order Differencing

Occasionally, the data will not appear to be stationary after first differences.

Continue differencing the data, called second-order differencing.

$$\begin{aligned} y_t'' &= y_t' - y_{t-1}' \\ &= (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) \\ &= y_t - 2y_{t-1} + y_{t-2} \end{aligned}$$

Where:

y_t'' : second differences of the data

In practice, it is almost never necessary to go beyond second-order differences.

Seasonal Differencing

The seasonal difference of a time series is the series of changes from one season to the next season.

Example: For monthly data, in which there are 12 periods in a season, the seasonal difference of Y at period t is $Y(t) - Y(t-12)$.

Removing the seasonal pattern in the time series is called first difference of seasonal difference.

Illustration of Seasonal Differencing

Date	Beer Sales (\$billion)	First Differences of Beer Sales		
Jan-17	4.79			
Feb-17	4.96	0.17	$= 4.96 - 4.79$	
Mar-17	5.64	0.68		
Apr-17	5.98	0.34		
May-17	6.08	0.1		
Jun-17	6.55	0.47		
Jul-17	6.11	-0.44		
Aug-17	5.37	-0.74		
Sep-17	5.17	-0.2	$= 5.17 - 4.79$	
Oct-17	5.48	0.31		
Nov-17	4.49	-0.99		
Dec-17	4.65	0.16		
Jan-18	5.17	0.52	Seasonal Difference	
Feb-18	5.57	0.4	0.38	
Mar-18	6.92	1.35	0.61	0.23
Apr-18	7.1	0.18	1.28	0.67
May-18	7.02	-0.08	1.12	-0.16
Jun-18	7.58	0.56	0.94	-0.18
Jul-18	6.93	-0.65	1.03	0.09
			0.82	-0.21

First Difference of seasonal difference

$= 0.61 - 0.38$

Differencing Apple Stock Price Series Example

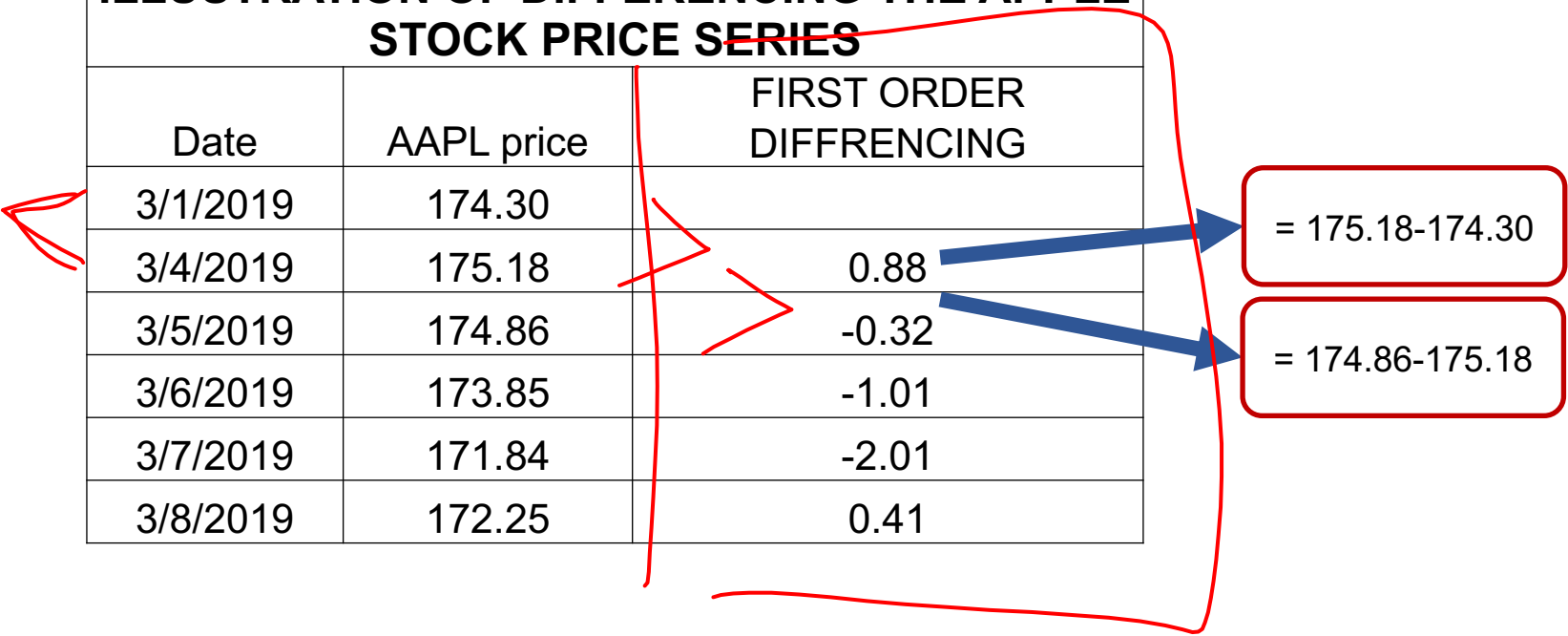
Now, let's go back to Apple stock price series.
Since Apple stock price series is non stationary:

- Conduct first differencing for the price series.

- Then, perform the stationarity test on first differences to see if the first-differenced series is stationary or not.

First Differencing Illustration

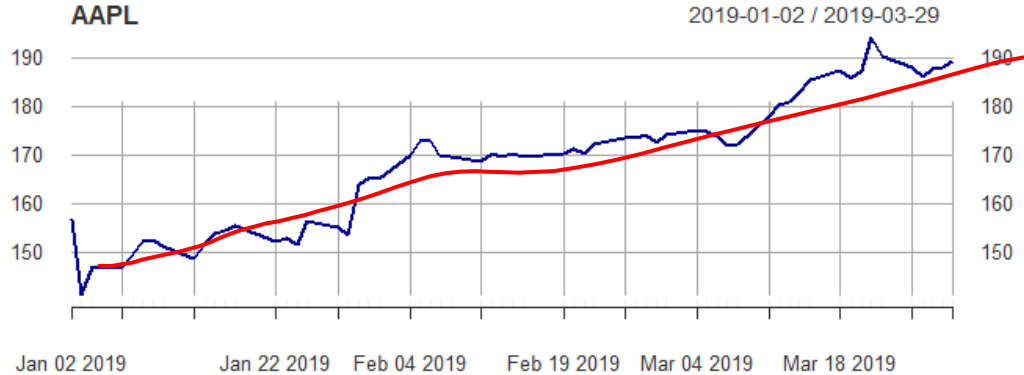
ILLUSTRATION OF DIFFERENCING THE APPLE STOCK PRICE SERIES		
Date	AAPL price	FIRST ORDER DIFFRENCING
3/1/2019	174.30	
3/4/2019	175.18	0.88
3/5/2019	174.86	-0.32
3/6/2019	173.85	-1.01
3/7/2019	171.84	-2.01
3/8/2019	172.25	0.41


$$= 175.18 - 174.30$$

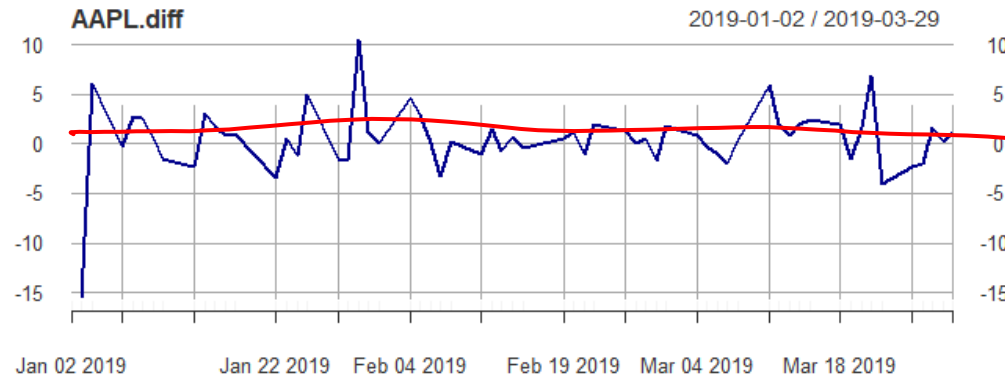
$$= 174.86 - 175.18$$

Plot AAPL Price Series Before and After Differencing

Original
Data



First
Differences
Data



Apple Price Series Stationarity Testing on First Differences

Perform ADF test on R

```
# Stationarity testing on first differences
```

```
AAPL.diff= diff(AAPL)
```

```
AAPL.diff = na.omit(AAPL.diff)
```


```
diff.adf =adf.test(AAPL.diff)
```

```
print(diff.adf)
```


Interpret ADF Test Results

Augmented Dickey-Fuller Test

```
data: AAPL.diff  
Dickey-Fuller = -4.0506, Lag order = 3, p-value = 0.01348  
alternative hypothesis: stationary
```



p-value = 0.013, which is less than 0.05 => Reject the null hypothesis => the Apple Stock Price series on **first differences** is **stationary**

Conclusion on the Stationarity of Apple Price Series

Apple price series is nonstationary on its original data, but stationary on first differences.

It has order of integration 1, denoted as $I(1)$.

Order of Integration

Order of integration is used to identify the minimum number of differences needed to be taken from original series to get a stationary one, denoted as $I(d)$.

When the series is stationary at level, it has order of integration 0, denoted as $I(0)$.

When the series is nonstationary at level but stationary at first differences, it has order of integration 1, denoted as $I(1)$.

What We Have Learned So Far?

Distinguish between nonstationarity/stationarity and strong-sense/weak-sense stationarity.

Understand why we need stationary data.

Perform stationarity test by using ADF test.

Know when we need to use first-/second-order differencing.

Know the order of integration.

Stationarity Testing Procedure

Step 1: Plot the graph to see if the time series looks stationary or not.

Step 2: Perform stationarity test by using ADF/PP/KPSS tests.

Step 3:

If the time series is stationary=> The time series is integrated at order 0, $I(0)$.

If the series is nonstationary, difference the data.

Step 4: Plot the time series on first differences to see if it looks stationary or not (Normally, the time series looks stationary after first differencing.)

Step 5: Perform the stationarity test on first differences by using ADF/PP/KPSS tests.

If the time series is stationary=> The time series is integrated at order 1, $I(1)$.

If the series is nonstationary, difference the data a second time=> repeat the procedure from Step 3 to Step 5.

Practice Test

We have S&P500.csv file with price series from 01/02/2019–05/31/2019.

1. Plot the series.
2. Do the stationarity test to see if the series is stationary.
3. If the series is nonstationary, do the first differencing.
4. Plot the series in first differences.
5. Do the stationarity test on first differences to see if it is stationary.
6. Find the order of integration.

Use R codes or any programming language that you are familiar with.

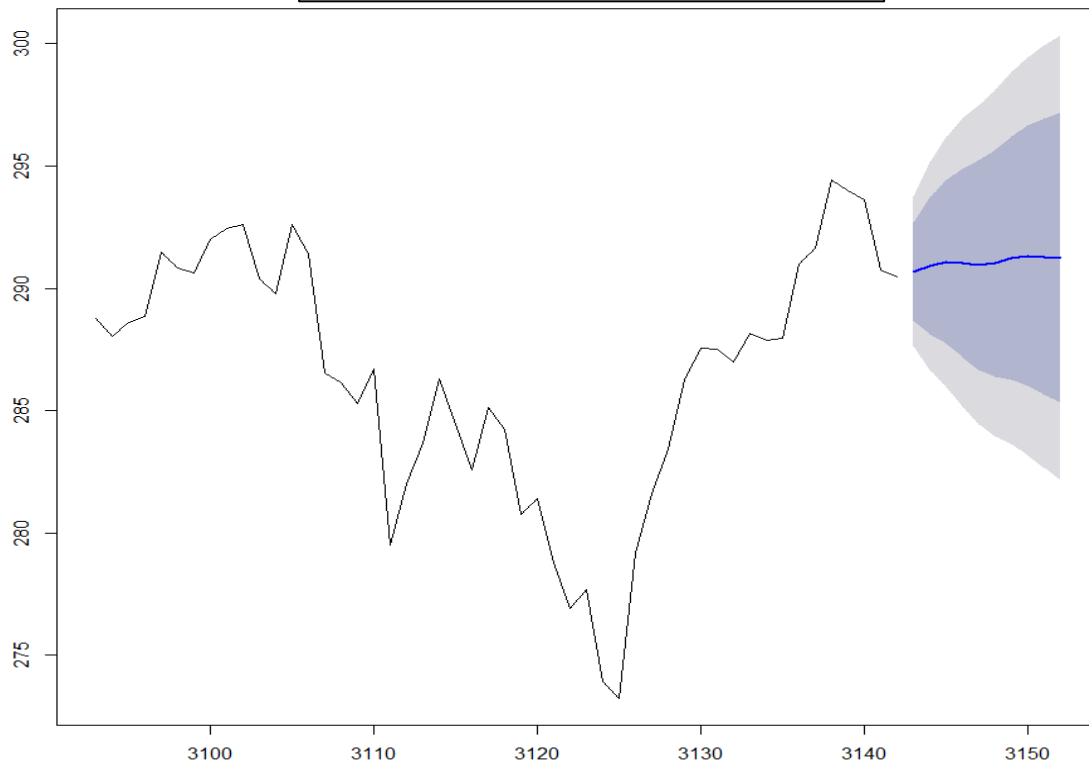
Elements of Forecasting

ARIMA Model

Dr. Sung Won Kim

SPY Prices

Forecasting SPY Prices 10 Days Ahead



ARIMA Model

Auto Regressive Integrated Moving Average (ARIMA)

Used to forecast a single time series

Includes three parts:

Auto Regressive – AR (p)

Integrated – I(d)

Moving Average – MA (q)



ARIMA (p,d,q)

Prerequisite: The data should be stationary.

Why Should We Use ARIMA?

Provide information to help investors, government regulators, policy makers, etc. make decisions

Before buying a bond or a stock, an investor really wants to know if it's worth buying and holding on to it for a period of time => ARIMA model helps them estimate the future value of such an asset based on the information from its past values.

Government regulators and policy makers use ARIMA model to forecast the future trend of some economic series in order to formulate policies.

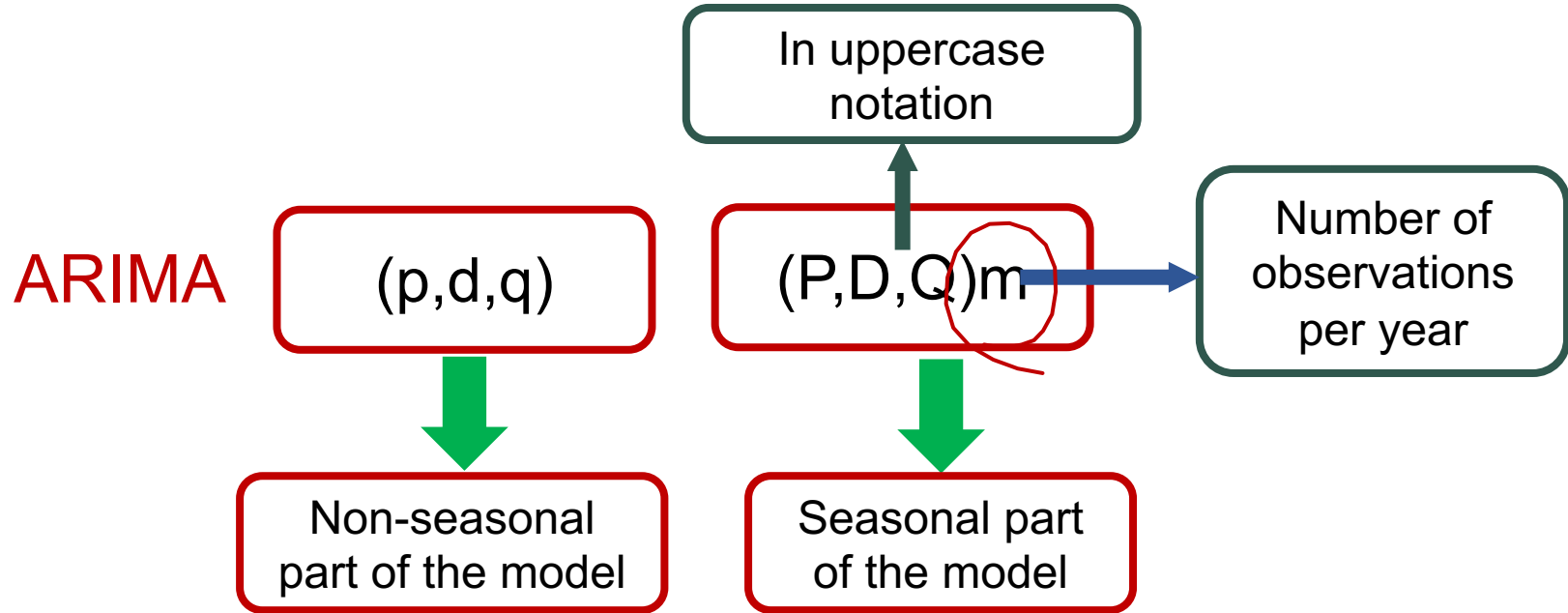
Two Types of ARIMA Models

Non-seasonal ARIMA model (p,d,q)

Seasonal ARIMA model $(P,D,Q)_m$

Seasonal vs. Non-seasonal ARIMA

Compared with non-seasonal ARIMA model, a seasonal ARIMA model includes additional seasonal terms inside.



Non-seasonal Model: ARIMA(p, d, q)



$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

where y'_t is the differenced series.

p = order of the autoregressive part

y'_t

d = degree of first differencing

q = order of moving average part

Non-seasonal Model: ARIMA(p, d, q)



$(2, 1, 1)$

$$y'_t = c + \phi_1 y'_{t-1} + \dots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \dots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

$$y'_t = c + \phi_1 y'_{t-1} + \phi_2 y'_{t-2} + \theta_1 \varepsilon_{t-1} + \varepsilon_t$$

ARIMA Breakdown – “I” Part

I: Integrated (d)

Integrated of order zero, $I(0)$

Integrated of order one, $I(1)$

This step simply requires differencing the series until it gets stationary.

ARIMA Breakdown – “AR” Part

AR(p): Autoregressive model of order p

p: Lags of the stationarized time series

ARIMA(1,0,0) has a value of $p=1$.

AR(p) tells us how many lags of the stationarized time series should be used in the forecasting equation.

ARIMA Breakdown – “MA” Part

MA(q): Moving average

q: Lags of forecast errors

ARIMA(0,0,1) has a value of $q=1$.

MA tells us how many lags of the forecast errors should be included in the forecasting equation.

Special Cases of ARIMA Models

ARIMA (0,0,0): A white noise model

ARIMA (0,1,0): A random walk model

ARIMA (0,1,0)+c: A random walk with drift model

ARIMA (1,0,0): First-order autoregressive model

ARIMA (0,0,1): Moving-average model

Elements of Forecasting

ARIMA Modelling Example

Dr. Sung Won Kim

ARIMA Modelling Procedure

1. Plot the data to see if there are any abnormal observations.
2. Conduct the stationarity test. If the series is nonstationary, take differences of the data until achieved stationarity.
3. Examine the pattern of autocorrelations (ACF) and partial autocorrelations (PACF) to determine if lags of the stationarized series and/or lags of the forecast errors should be included in the forecasting equation.
4. Fit the model that is suggested.

ARIMA Modelling Example

To illustrate ARIMA model, I will use the S&P 500 exchange traded fund prices series from 2007–2019 (which is 3142 working days) pulled from Yahoo Finance website.

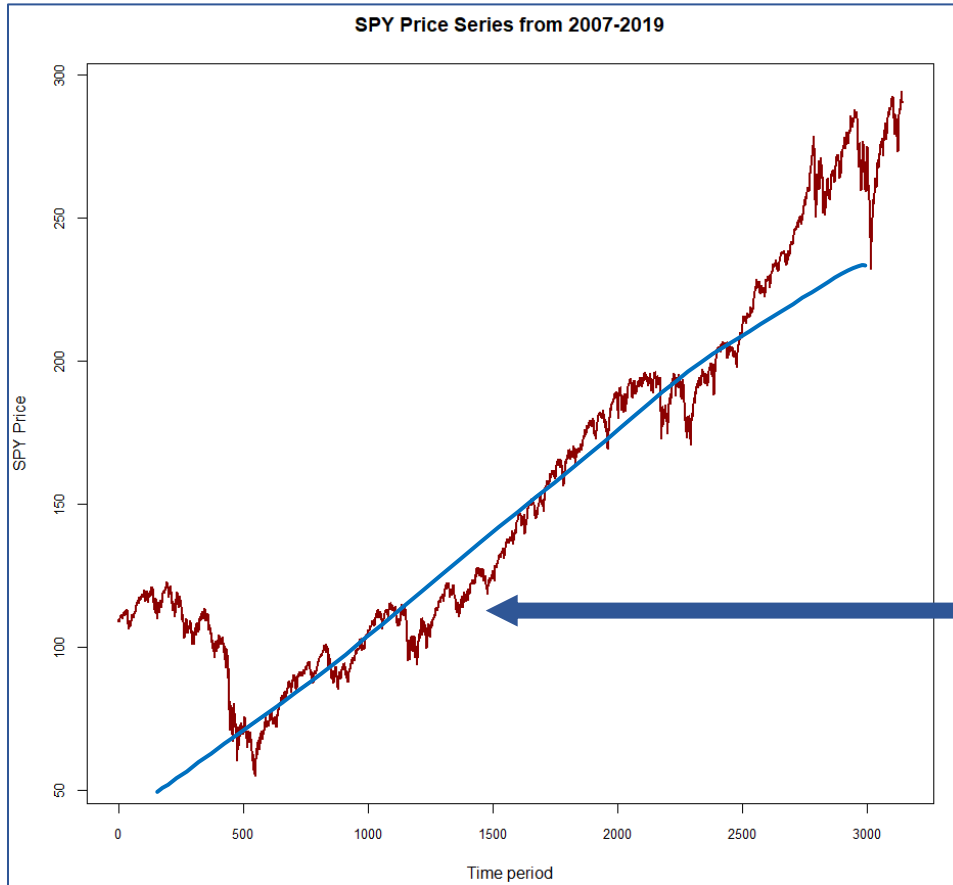
```
install.packages("ggplot2")
```

```
install.packages("tseries")
```

```
install.packages("forecast")
```

```
install.packages("urca")
```

Step 1: Plot the Data



```
#Announce SPY series data
SPY=(data$Price)
head(SPY, n=10)
#Plot the data
plot(SPY, col="darkred", xlab="Time period",
      ylab="SPY Price", type="l", lwd=2, cex=2, main="SPY
      Price Series from 2007-2019", cex.axis=.8)
```

The SPY price series
look nonstationary →
Needs stationarity test

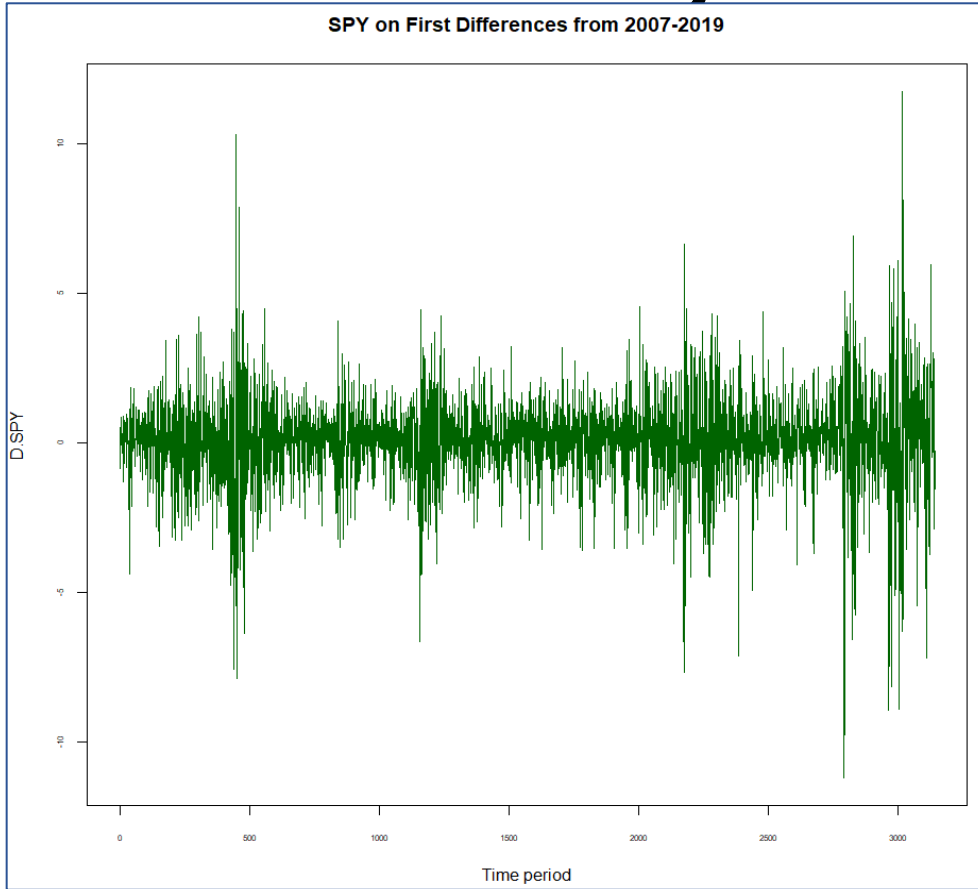
Step 2: Stationarity Testing

```
#####  
# Augmented Dickey-Fuller Test Unit Root Test #  
#####  
  
Test regression none  
  
Call:  
lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)  
  
Residuals:  
      Min       1Q   Median       3Q      Max   
-11.5572  -0.6464   0.0551   0.7659  11.3740  
  
Coefficients:  
              Estimate Std. Error t value Pr(>|t|)      
z.lag.1      0.0003793  0.0001664   2.280  0.0227 *      
z.diff.lag -0.0426809  0.0178426  -2.392  0.0168 *      
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 1.573 on 3138 degrees of freedom  
Multiple R-squared:  0.003309, Adjusted R-squared:  0.002674  
F-statistic: 5.209 on 2 and 3138 DF, p-value: 0.005514  
  
value of test-statistic is: 2.2795  
  
Critical values for test statistics:  
      1pct   5pct  10pct  
tau1 -2.58 -1.95 -1.62
```

```
# Stationarity testing  
StationarityTest =  
ur.df(SPY,type="none",selectlags =  
"AIC")  
summary(StationarityTest)
```

The series is
nonstationary at 1%
significance level →
Needs to be differenced
to make sure the series
stationary at 1%

Plot the Stationary SPY Series



```
#Stationarity Tesing on first Differences  
## Make First Differences on the series
```

```
D.SPY =diff(SPY, lag = 1, differences = 1)
```

```
##Plot the grpah on First Differences
```

```
plot(D.SPY, col="darkgreen", xlab="Time  
period", ylab="D.SPY", type="l",lwd=1,  
cex=1, main="SPY on First Differences from  
2007-2019",cex.axis=.5)
```


Stationarity Testing on First Differences

```
#####  
# Augmented Dickey-Fuller Test Unit Root Test #  
#####
```

```
Test regression none
```

```
call:  
lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
```

```
Residuals:  
      Min       1Q   Median       3Q      Max   
-11.4673  -0.5829   0.1238   0.8328  11.2531
```

```
Coefficients:  
              Estimate Std. Error t value Pr(>|t|)      
z.lag.1      -1.08463    0.02574  -42.133  <2e-16 ***  
z.diff.lag    0.04221    0.01785   2.365    0.0181 *    
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 1.573 on 3137 degrees of freedom  
Multiple R-squared:  0.5212,    Adjusted R-squared:  0.5209  
F-statistic: 1708 on 2 and 3137 DF,  p-value: < 2.2e-16
```

```
value of test-statistic is: -42.1325
```

```
critical values for test statistics:
```

```
      1pct   5pct  10pct  
tau1 -2.58  -1.95  -1.62
```

```
## Stationarity Testing on First Differences  
Stationarity_Diff= ur.df(D.SPY,type="none",selectlags = "AIC")  
summary(Stationarity_Diff)
```

Now, we have a
stationary series!

Determine Lags for AR & MA

To identify the appropriate lags for AR and MA, we can use:

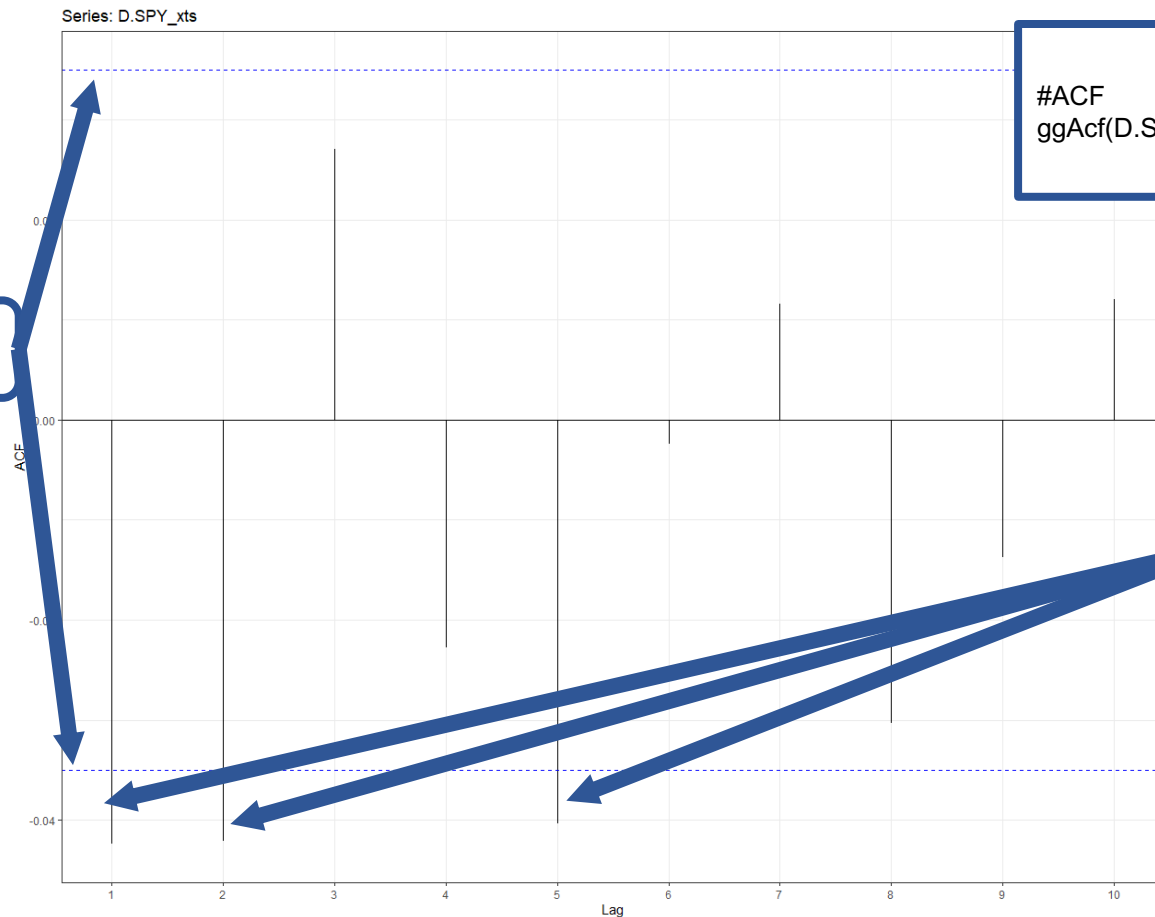
The plots of ACF and PACF

ACF – Autocorrelation function

PACF – Partial autocorrelation function

PACF will indicate AR terms, and ACF will indicate MA terms.

Step 3: ACF Plot

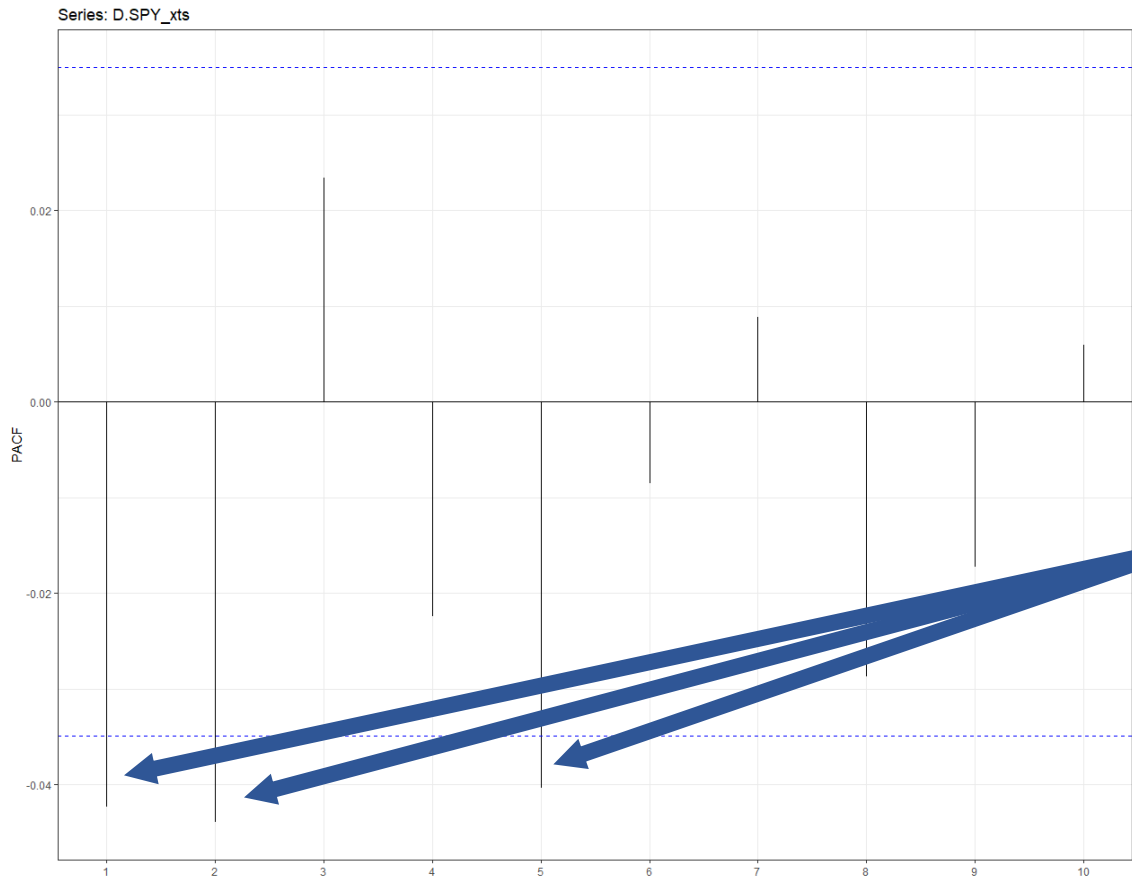


Significance
lines

```
#ACF  
ggAcf(D.SPY_xts, lag.max = 10) + theme_bw()
```

There is
significance auto-
correlation for the
first, second, and
fifth lags

Step 3: PACF Plot



```
# PACF plot  
ggPacf(D.SPY_xts, lag.max =  
10) + theme_bw()
```

PACF is more focused on isolating which lags are important. It can be used to inform how many lagged returns should be included in a model to explain returns.

We should include at least two, and possibly up to five, lags to AR(p).

ARIMA Model

From ACF and PACF plots, we already figured out p , d , q for ARIMA (p, d, q).

PACF lets us know the value of $AR(p)$: 1, 2, and 5 lags.

ACF informs us the value of $MA(q)$: 1, 2, and 5 lags

$I(d)$ is how many times you difference the data: here d will be 0 since we will build ARIMA model on stationary series.

Note: The fifth lag is a judgement call because at some level it is hard to believe returns from five days ago will be useful in predicting or explaining current returns.

The next step is to build up ARIMA (1,0,1), ARIMA (1,0,2), ARIMA (2,0,1), ARIMA (2,0,2) and pick “the best” one!

ARIMA Modelling in R

Fit the ARIMA Model

ARIMA1 ← Arima(D.SPY_xts, order = c(1, 0, 1))

ARIMA2 ← Arima(D.SPY_xts, order = c(1, 0, 2))

ARIMA3 ← Arima(D.SPY_xts, order = c(2, 0, 1))

ARIMA4 ← Arima(D.SPY_xts, order = c(2, 0, 2))

```
> summary(ARIMA1)
Series: D.SPY
ARIMA(1,0,1) with non-zero mean

Coefficients:
      ar1      ma1      mean
    0.7801 -0.8179  0.0580
s.e.  0.1292  0.1191  0.0232
```

```
> summary(ARIMA3)
Series: D.SPY
ARIMA(2,0,1) with non-zero mean

Coefficients:
      ar1      ar2      ma1      mean
   -0.4358  -0.0623   0.3928   0.0577
s.e.   0.2489   0.0186   0.2490   0.0261
```

```
> summary(ARIMA2)
Series: D.SPY
ARIMA(1,0,2) with non-zero mean

Coefficients:
      ar1      ma1      ma2      mean
   -0.4267   0.3841  -0.0615   0.0578
s.e.   0.2584   0.2578   0.0184   0.0260
```

```
> summary(ARIMA4)
Series: D.SPY
ARIMA(2,0,2) with non-zero mean

Coefficients:
      ar1      ar2      ma1      ma2      mean
   -0.6206  -0.8085   0.5935   0.7646   0.0581
s.e.   0.1118   0.1754   0.1238   0.1870   0.0272
```

ARIMA Modelling in R – Conclusion

In our four estimated ARIMA models:

Coefficients of ARIMA 1 and ARIMA 4 are not significant (standard errors are too large, bigger than 5%).

Only second lag of MA part in ARIMA 2 and second lag of AR part in ARIMA 3 are significant (standard errors are less than 5%)!

Forecasting SPY Future Prices in R

In R, we have “`auto.arima`” function from the `forecast` package, which automatically selects a good choice for p , d , q based on the data. Since our four forecasted models do not give us the best one, we can utilize this function.

I will take the original SPY price series, not the differenced one. The `auto.arima` function will check for the proper d as part of the routine.

Forecasting SPY Future Prices in R (cont.)

```
#Building ARIMA Forecasting Graph
```

```
## Find best-fitted ARIMA model
```

```
Fitted.Arima = auto.arima(SPY)
```

```
## Plot forecated ARIMA with the last 50 days
```

```
q=forecast(Fitted.Arima,h=10)
```

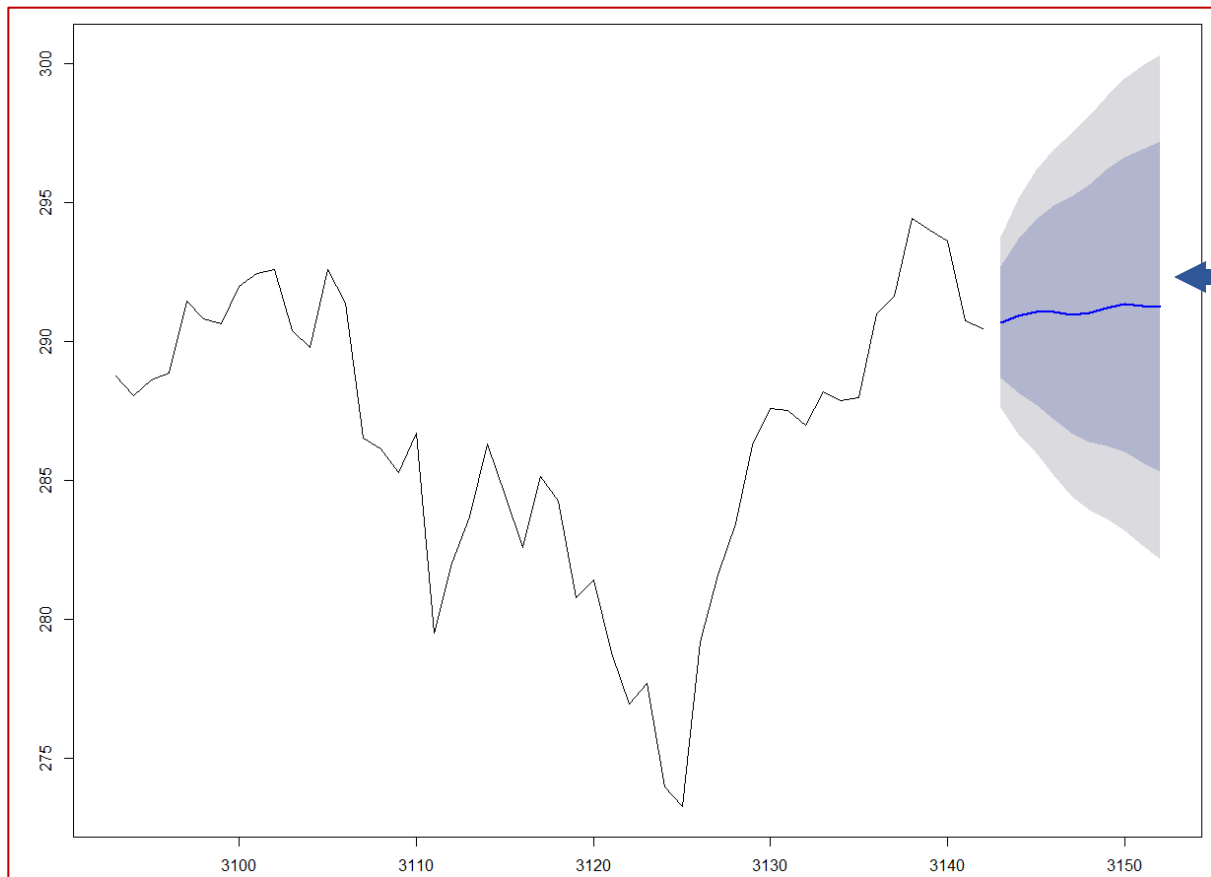
```
summary(q)
```

```
plot(q, include=50)
```

h=10 means I want to
forecast SPY stock price
in the next 10 days

The graph includes SPY
prices in the last 50 days

Forecasting SPY Future Prices in R



These are the
forecasted
prices in the
next 10 days!

auto.arima()



This R function uses AIC or BIC to evaluate the models.

Akaike's Information Criterion



$$AIC = T \log \left(\frac{SSE}{T} \right) + \underline{2(k + 2)}$$

Schwarz's Bayesian Information Criterion



$$BIC = T \log \left(\frac{SSE}{T} \right) + \underline{(k + 2) \log(T)}$$