79546 - Timeseries Emphirical Paper

Clear all variables in work space and install packages

```
rm(list=ls())

requiredPackages = c('quantmod','TTR','tseries')
for(p in requiredPackages){
  if(!require(p,character.only = TRUE)) install.packages(p) #install package if it does not exist
  library(p,character.only = TRUE)
}

#suppress `getSymbols` message
options("getSymbols.warning4.0"=FALSE)
```

Load the forecasting packages

```
library(fpp2)
```

```
## -- Attaching packages ---------------------------------------------- fpp2 2.4 --

## v ggplot2   3.3.5     v fma       2.4
## v forecast  8.16      v expsmooth 2.3

##
```

```
library(TTR)
library(quantmod)
library(ggplot2)
library(tseries)
```

Get to Know Time-Series Data

Load Dataset

Data is collected from Yahoo Finance using the Quantitative Financial Modeling Framework (Quantmod). Data obtained in eXtensible-Time-Series format is being used for data exploration.

```
#Download data from yahoo finance
df_tsm <- getSymbols('TSM',src='yahoo',auto.assign=FALSE,from="2011-01-01")

#Check the contents of the data
class(df_tsm)
```

```
## [1] "xts" "zoo"
```

```r
#List the number of rows in the data
nrow(df_tsm)
```

```
## [1] 2776
```

```r
#Print the last 6 rows of the data
tail(df_tsm)
```

```
##            TSM.Open TSM.High  TSM.Low TSM.Close TSM.Volume TSM.Adjusted
## 2022-01-04   130.87   135.50 130.3000    133.40   25554900       133.40
## 2022-01-05   130.71   130.88 126.8800    127.06   17891200       127.06
## 2022-01-06   127.00   129.00 124.8100    128.47   16249000       128.47
## 2022-01-07   126.55   127.14 123.3100    123.50   21239000       123.50
## 2022-01-10   125.11   125.87 123.2600    125.01   11857700       125.01
## 2022-01-11   126.54   129.55 125.4975    129.17   11861326       129.17
```

Stock price visualization

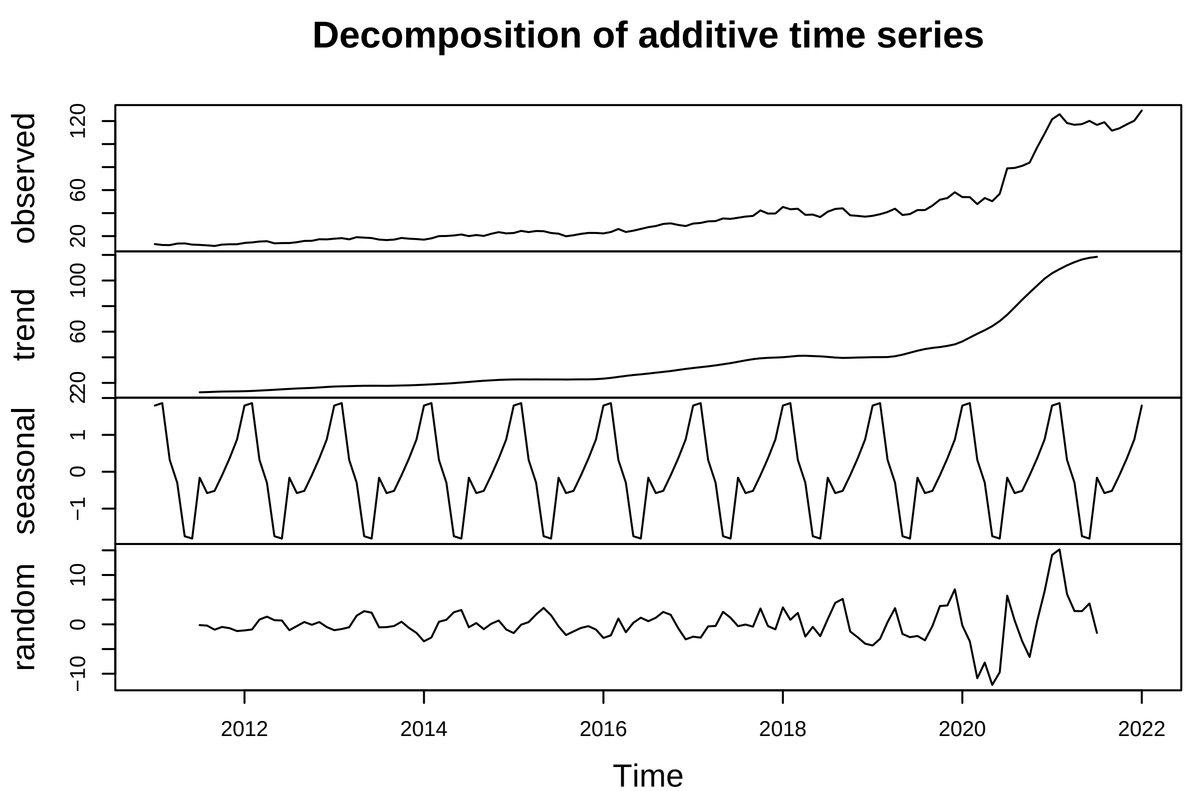
This shows the patterns of the data.

```r
tsm_title = "Taiwan Semiconductor Manufacturing Company Limited Stock Price (TSM) (2011-2022)"


chartSeries(df_tsm , name="TSM price 2011-2022")
```

TSM share price has grown upwards from 2011 to 2022

Time plot of the data

```
#returns the closing price
tsm_close = Cl(to.monthly(df_tsm))

#decompose the data
dc <- decompose(as.ts(tsm_close, start=c(2011,1)))
plot(dc)
```



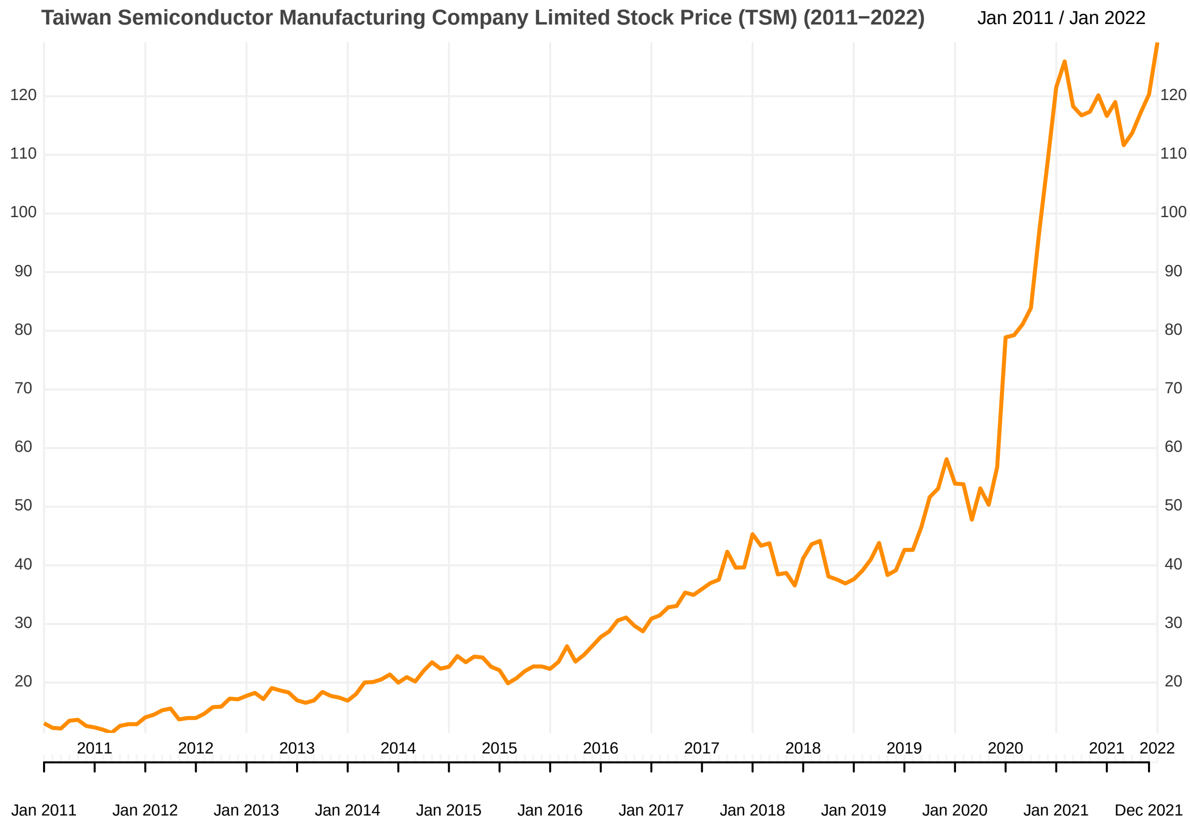

```
#Display seasonal stock data
dc$seasonal
```

```
##              Jan         Feb         Mar         Apr         May         Jun
## 2011  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2012  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2013  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2014  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2015  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2016  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2017  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2018  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
```

```
## 2019  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2020  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2021  1.79468171  1.86118201  0.32059876 -0.29815104 -1.74860927 -1.81240113
## 2022  1.79468171
##              Jul         Aug         Sep         Oct         Nov         Dec
## 2011 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2012 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2013 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2014 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2015 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2016 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2017 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2018 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2019 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2020 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2021 -0.16154515 -0.57835953 -0.51898464 -0.09623412  0.36151508  0.87630733
## 2022
```

The output shows for plots of TSM closing price which are:
- **Observed**: Original plot of the data.
- **Trend** : There is an upward trend that is significant from 2018.
- **Seasonal** There is repetitive seasonal fluctuation of data.The closing price reached the highest in January and the lowest in June.This shows that a good time to sell is beginning of the year and the risht time to buy is mid year
- **Random** irregular or random fluctuation not captured by the trend and seasonal.

```
#plot(df_tsm$TSM.Close,main = tsm_title)
chart_Series(tsm_close,name=tsm_title)
```



From the figure above TSM stock price has a `strong positive trend`.This shows that it is `non-stationary`

Stationarity Test on the stock price

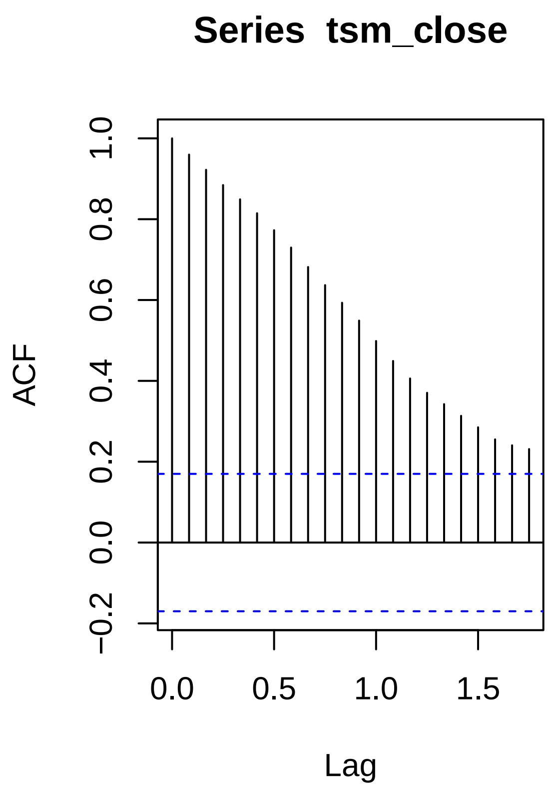
```
adf.test(tsm_close)
```

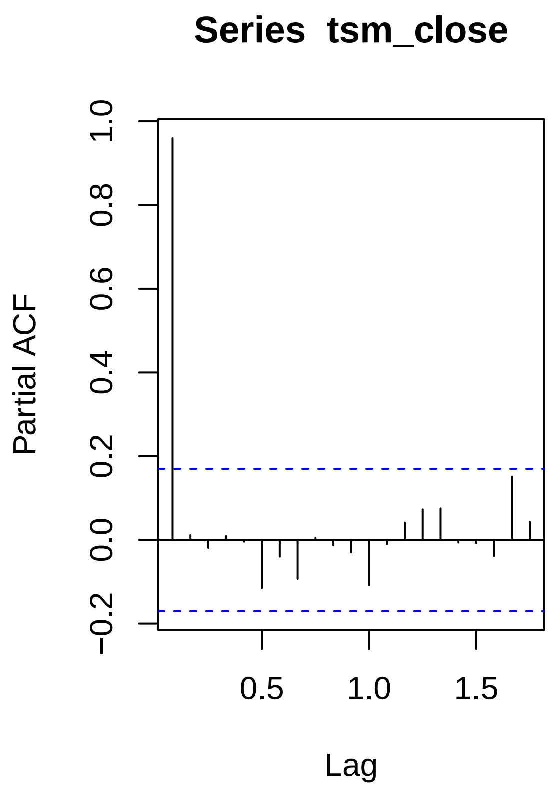
```
##
##  Augmented Dickey-Fuller Test
##
## data:  tsm_close
## Dickey-Fuller = -0.44478, Lag order = 5, p-value = 0.983
## alternative hypothesis: stationary
```

The p-value is not less than 0.05 hence we fail to reject null hypothesis.
This means that the time series is non-stationary.

ACF and PACF plots of the time-series data

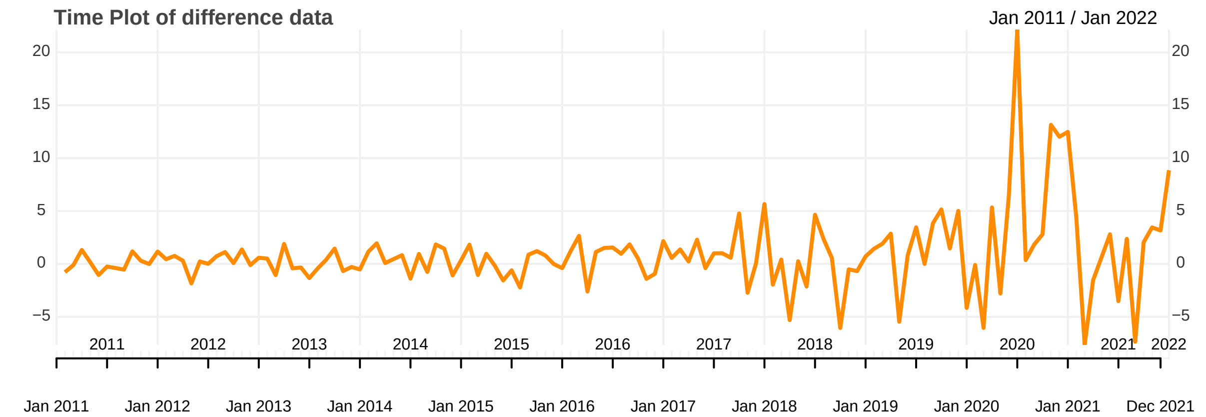
```
par(mfrow=c(1,2))
acf(tsm_close)
pacf(tsm_close)
```

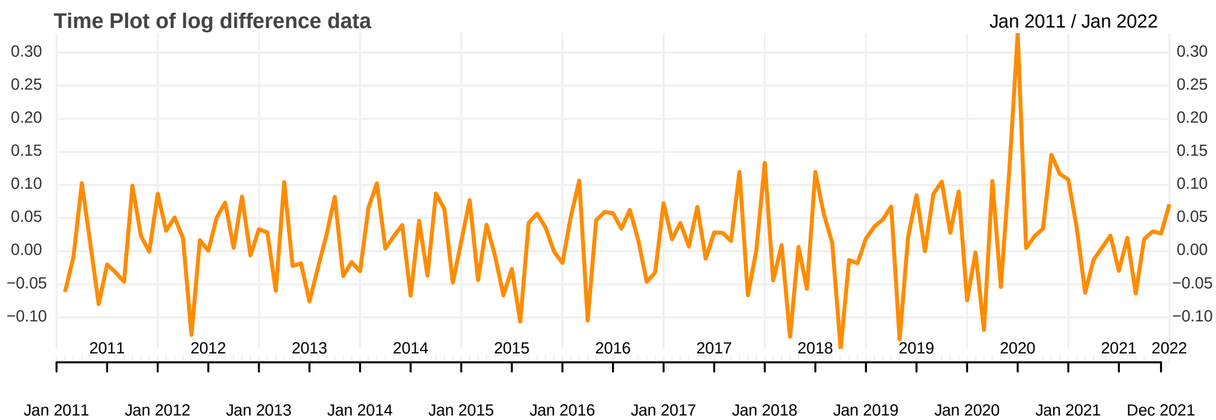




from the ACF and PACF plot,AR model would be ideal for this stock price
The trend can be removed by `differensing` the data to removes the trend

```
par(mfrow=c(2,1))

dy = diff(tsm_close,lag = 1)
chart_Series(dy,name="Time Plot of difference data")

wld = diff(log(tsm_close))
chart_Series(wld,name="Time Plot of log difference data")
```





Stationarity test of Differencing log time-series

```
adf.test(wld[!is.na(wld)])
```

```
## Warning in adf.test(wld[!is.na(wld)]): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  wld[!is.na(wld)]
## Dickey-Fuller = -4.8071, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

The p-value is less than 0.05 hence we accept null hypothesis.
This means that the difference time series is stationary.

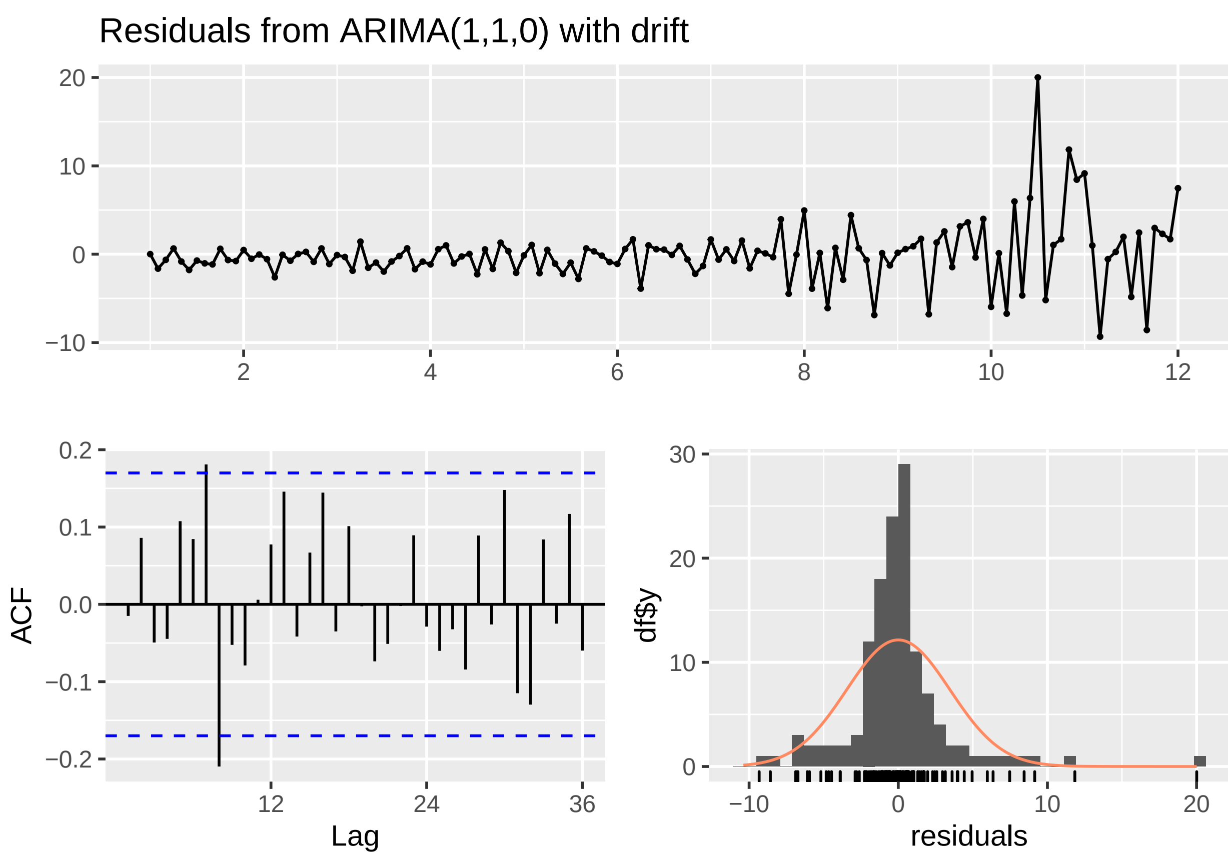
Find the best fit arima model

`auto.arima` finds the best fit ARIMA model the forecasting

```r
# differencing is set to 1 d=1
# if TRACE = TRUE prints out all models that have been tried
fit_arima = auto.arima(tsm_close,d=1,stepwise = FALSE,approximation = FALSE,trace = FALSE)
print(summary(fit_arima))
```

```
## Series: tsm_close
## ARIMA(1,1,0) with drift
##
## Coefficients:
##          ar1   drift
##       0.2193  0.8929
## s.e.  0.0863  0.3852
##
## sigma^2 = 12.17:  log likelihood = -351.25
## AIC=708.51   AICc=708.69   BIC=717.15
##
## Training set error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.003168336 3.449081 2.078593 -1.510854 5.428965 0.1884236
##                    ACF1
## Training set -0.0149737
```

```r
checkresiduals(fit_arima,plot=TRUE)
```




```
##
```

```
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0) with drift
## Q* = 29.309, df = 22, p-value = 0.1362
##
## Model df: 2.   Total lags used: 24
```

Genarate a 24-month forecast using best fit ARIMA model

```
fcst =  forecast(fit_arima, h=24,level=c(95))
autoplot(fcst)
```