Time Series Forecasting with ARMA Model

YEN CHEN HSU

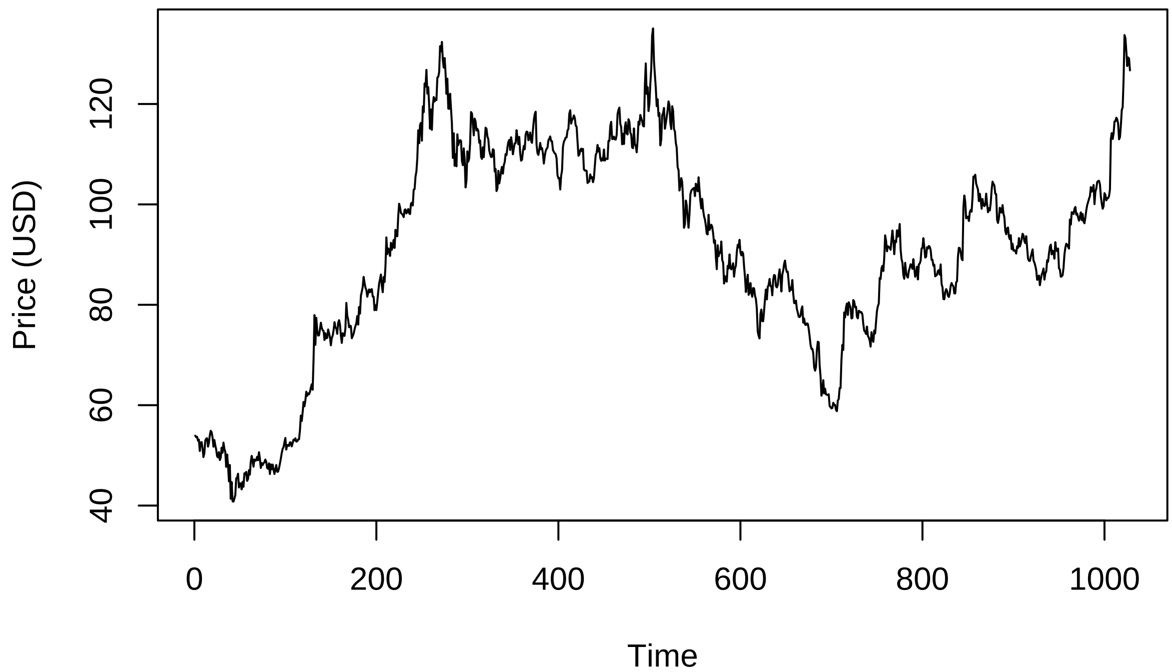
2024-05-18

```r
suppressPackageStartupMessages({
  library(TSA)
  library(ggplot2)
  library(dplyr)
  library(forecast)
  library(tseries)
  library(Metrics)
  library(lmtest)
  library(readr)
})
```

This data set contains the Adjusted Closing Stock Price of Taiwan Semiconductor Manufacturing Company Limited (TSMC) from 2020/01/17 to 2024/02/16, we are interested in knowing the trend for the following 2 weeks (14 days) data source: https://finance.yahoo.com/quote/TSM/

```r
TSMC <- read_csv("TSMC.csv",show_col_types = FALSE)
```

```r
# plot the data
ts.plot(TSMC$Price, xlab="Time", ylab = "Price (USD)")
```




```r
# provide sample size
set.seed(42) # For reproducibility
sample_size <- 100
sample_indices <- sample(1:length(TSMC$Price), sample_size)

# Create a Sampled Time Series
sampled_prices <- TSMC$Price[sample_indices]
TSMC_sample <- ts(sampled_prices, start=start(TSMC$Price), frequency=frequency(TSMC$Price))
print(TSMC_sample)
```

```
## Time Series:
## Start = 1
## End = 100
## Frequency = 1
##   [1]  96.98549 115.04890  83.70407  43.59749  63.49279 112.21228  51.17901
##   [8]  89.61135 111.96563  89.83604  73.80183  78.22403 106.80381 114.71884
##  [15]  96.89807 103.75995  92.15515 108.99618 118.79000 116.14816  86.49239
##  [22] 100.17000 103.15000  73.28713 115.07119  90.07921  85.03924 103.54211
##  [29]  98.43284 115.06125 112.74355 111.32676 103.37038 133.73000  76.35803
##  [36] 104.78396  93.84834 112.47131  99.13000  88.12691 112.80238  82.26535
##  [43] 100.47648  73.41795 101.67000 113.37919  81.60870 119.20969  99.42501
##  [50] 135.06270  92.47420 113.32227  79.61258  89.17973 109.26757  88.83137
##  [57]  78.84476 119.71135 113.08080  63.08081 114.81406  53.72000 117.99488
##  [64] 111.45230  92.74730 102.13607  59.78259  87.24428  41.33724 101.24000
##  [71]  51.30789  96.85355  52.06420  98.09798  53.14631 110.83667  76.43937
##  [78]  48.44888  89.08841 117.20283 102.96748  78.52988  52.49553 113.44270
```

```
## [85]  78.45184 115.87408 109.99233  83.55859  85.76030  48.42101  90.32333
## [92] 102.04695  93.67011  90.03569 113.39000 108.88100 111.16871  73.53966
## [99]  95.52859  53.48616
```

```r
# leave the last 2 weeks for testing data prediction
train_tsmc <- ts(TSMC$Price[1:1014], start = 1, end = 1014)
test_tsmc <- ts(TSMC$Price[1015:1028], start = 1015, end = 1028)
```

Stationarity Check with ADF test & ARMA Model Selection

Use the `ADF test` to check for stationarity. Remove trend if necessary, and check the residuals for spurious regression (proof of random walk)

Check ACF, PACF, and EACF for the order of the ARMA model (after differencing, if it has a random walk). Use AIC or BIC to select a final model from your candidate models.

```r
print(adf.test(train_tsmc)) # the p-value of the training data indicates non-stationary
```

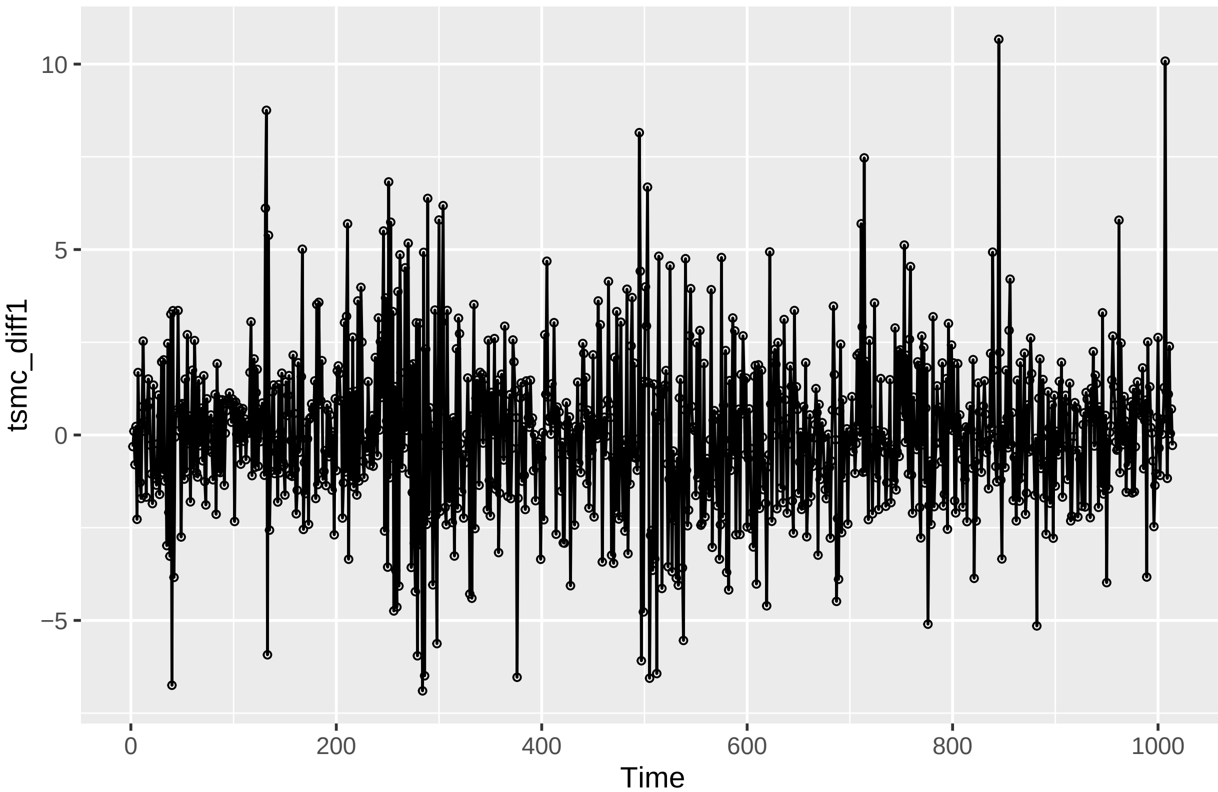
```
##
##  Augmented Dickey-Fuller Test
##
## data:  train_tsmc
## Dickey-Fuller = -1.8549, Lag order = 10, p-value = 0.6397
## alternative hypothesis: stationary
```

```r
tsmc_diff1=diff(train_tsmc,differences = 1) # conduct differencing to address non-stationary problem
print(adf.test(tsmc_diff1)) # after differencing, we obtain a stationary result
```

```
## Warning in adf.test(tsmc_diff1): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  tsmc_diff1
## Dickey-Fuller = -8.8665, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
```

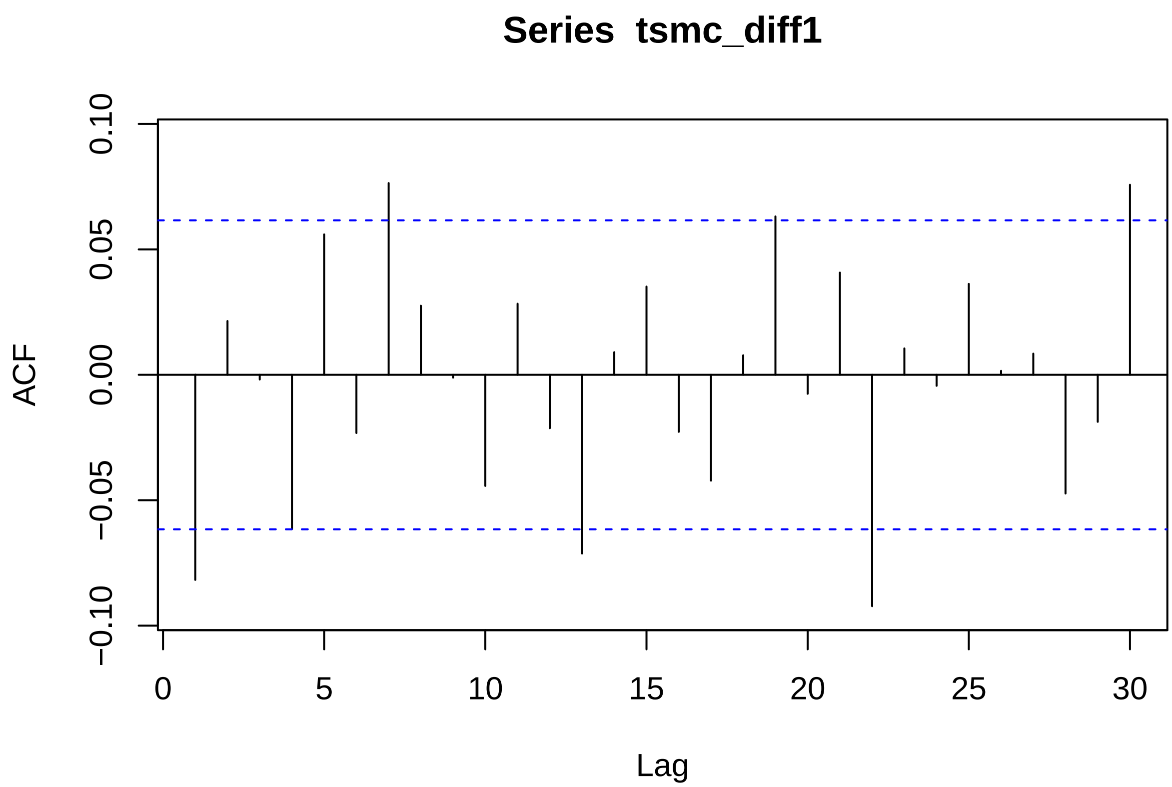
```r
autoplot(tsmc_diff1) + geom_point(shape = 1, size = 1)
```



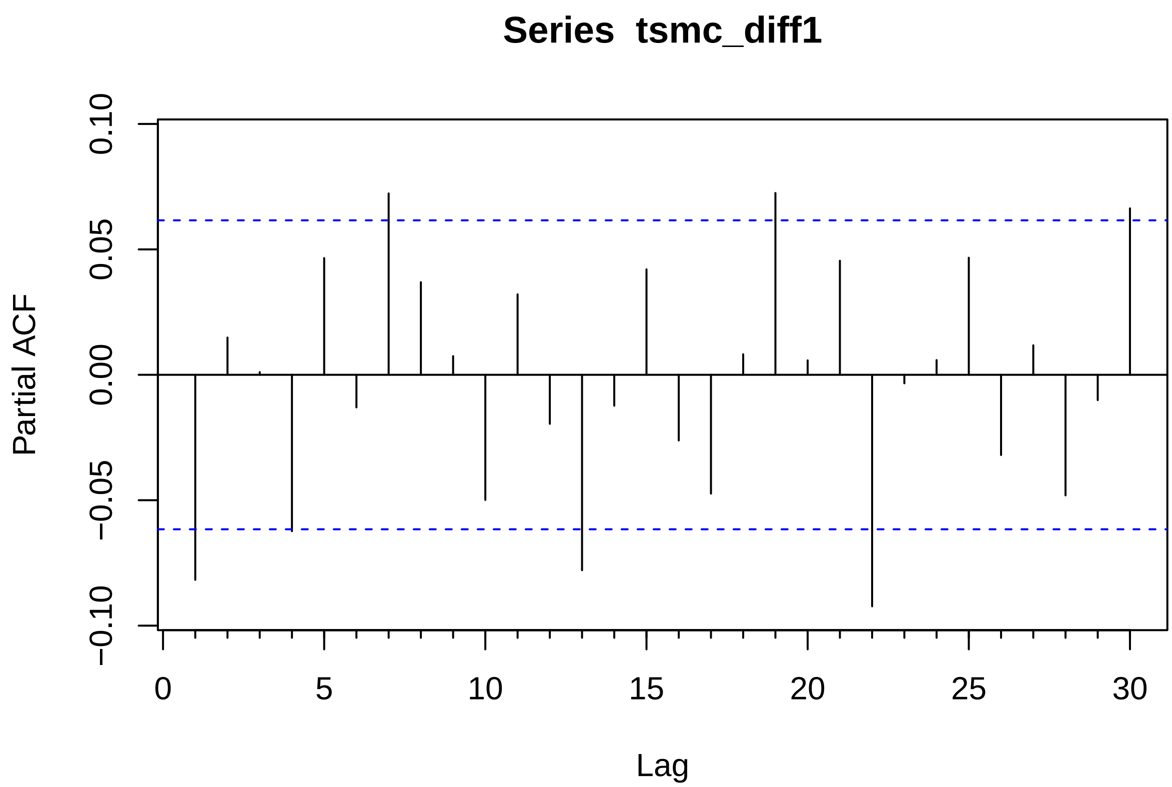

```
#Final Model: ARIMA(p,d,q)(P,D,Q)[S]
```

Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots are used to identify potential orders for an ARIMA model. The Extended ACF (EACF) is also used to refine the model selection.

```
Acf(tsmc_diff1)
```




```
Pacf(tsmc_diff1)
```



Judging from Acf and Pacf result, we are not able to identify p and q order for ARMA model, hence, we rely on eacf to obtain the result. If we look at the top left, the choice would be ARMA(3,0)

```
eacf(tsmc_diff1)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x o o o o o x o o o o  o  x  o
## 1 x o o o o o x o o o o  o  x  o
## 2 x o o o o o x o o o o  o  o  o
## 3 o x x o o o o o o o o  o  o  o
## 4 x x x x o o o x o o o  o  o  o
## 5 x x x x x o x o o o o  o  o  o
## 6 x x x x o o o o o o o  o  o  o
## 7 x x x x x o o o o o o  o  o  o
```

```
Arima(train_tsmc,order=c(3,1,0)) #from eacf result
```

```
## Series: train_tsmc
## ARIMA(3,1,0)
##
## Coefficients:
##           ar1     ar2     ar3
##       -0.0795  0.0160  0.0021
## s.e.   0.0314  0.0315  0.0314
##
```

```
## sigma^2 = 4.319:  log likelihood = -2176.9
## AIC=4361.8   AICc=4361.84   BIC=4381.48
```

```
auto.arima(train_tsmc)
```

```
## Series: train_tsmc
## ARIMA(0,1,1)
##
## Coefficients:
##           ma1
##       -0.0778
## s.e.   0.0307
##
## sigma^2 = 4.313:  log likelihood = -2177.15
## AIC=4358.3   AICc=4358.31   BIC=4368.14
```

Based on the EACF results, an ARIMA(3,1,0) model is initially fitted. However, automated model selection using auto.arima suggests an ARIMA(0,1,1) model based on lower AIC and BIC values.

Fitting our final model, write down the model

$$Y_t = -0.077848 \cdot e_{t-1} + e_t$$

Fitting with ARIMA Model

```
#arima_fit
tsmc_fit=Arima(train_tsmc,order=c(0,1,1))
coeftest(tsmc_fit)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ma1 -0.077848   0.030694 -2.5363   0.0112 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Based on the calculation, the magnitude of the ma1 coefficient is approximately 2.54 times larger than its standard error, therefore, the ma1 coefficient is considered significant.

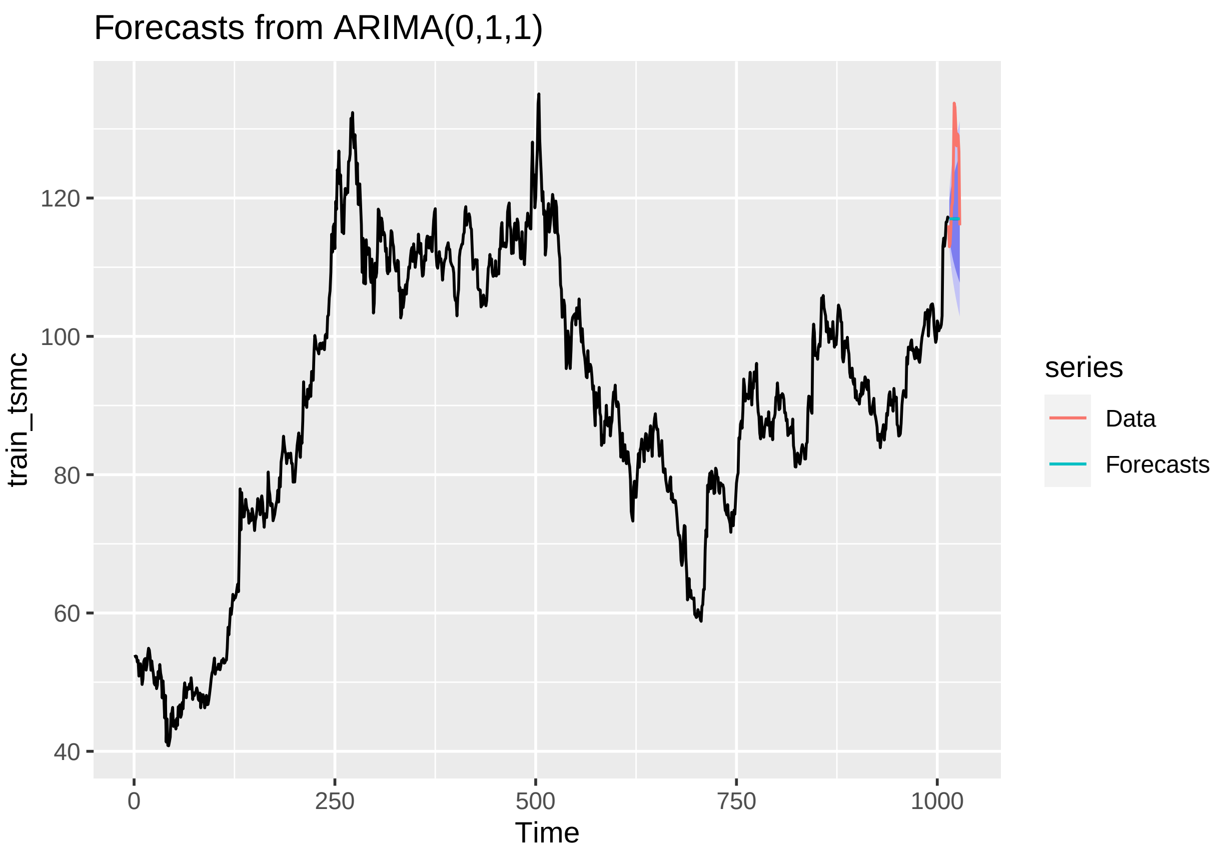
Forecast on the testing set. Report RMSE.

Plot the fitted value, as well as 80% and 95% prediction intervals, superimposed on the raw data.

```
rw2=Arima(train_tsmc,order=c(0,1,1))

rw2_pred <- forecast(rw2,h=14)
```

```r
autoplot(rw2_pred) +
  autolayer(ts(test_tsmc,start=1014,end=1028), series="Data") +
  autolayer(rw2_pred$mean, series="Forecasts")
```




```r
#Check the RMSE
rmse(test_tsmc,rw2_pred$mean)
```

```
## [1] 9.731735
```