The Apple Vision Pro's Stock Market Impact

My Dataset

I found this "Apple Stock Data and Key Affiliated Companies" (specifically "AAPL_daily_data") dataset after searching through Kaggle.

I will be modelling the changes in daily stock closing prices for Apple. I am specifically interested in the data spanning a few months around June 5th, 2023, the day the Apple vision Pro was announced.

In 2023, Apple announced a innovative piece of recreational technology. The goal of this study is to determine whether the promise of innovation alone can positively influence shareholder decisions.

My hypothesis is that Apple stock will see a unusual increase in share price after June 2025 for two key reasons: 1. As a publicly traded company, he Apple Vision Pro's reveal is likely to motivate individual investors to buy more shares 2. , Apple's new product announcements will attract attention from finance sector entities.

Presenting the Data

```
library(itsmr)
library(knitr)
library(dplyr)
```

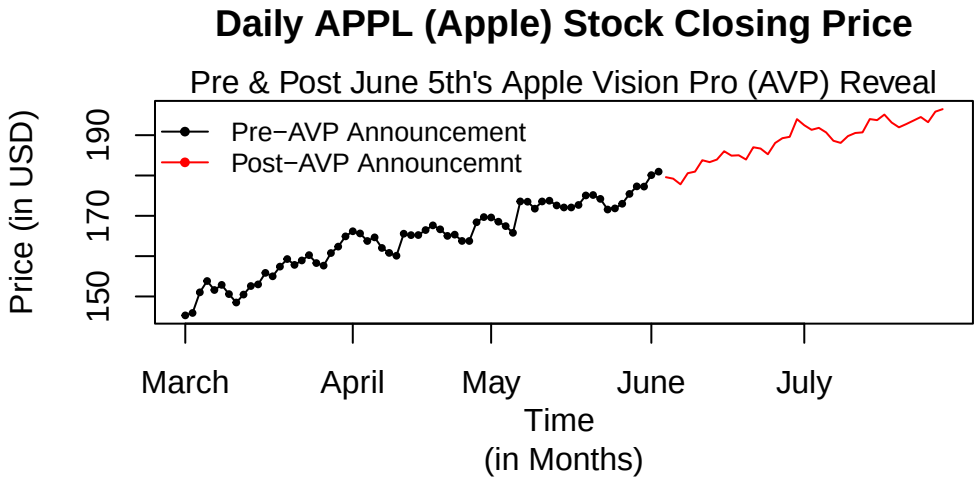
```
Attaching package: 'dplyr'
```

```
The following objects are masked from 'package:stats':

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
apple_data <- read.csv("AAPL_daily_data.csv", header = TRUE)
split_dates <- strsplit(as.character(apple_data$Date), "-")
apple_data$Month <- sapply(split_dates, function(x) as.numeric(x[2]))
clean_apple_base <- apple_data[,c("Date", "Month", "Close")]

start_date <- as.Date("2023-03-01")
end_date <- as.Date("2023-07-31")

final_apple_stock<- clean_apple_base[clean_apple_base$Date >= start_date
                          & clean_apple_base$Date <= end_date, ]
ticks <- c(min(final_apple_stock$Date[final_apple_stock$Month == 3]),
           min(final_apple_stock$Date[final_apple_stock$Month == 4]),
                 min(final_apple_stock$Date[final_apple_stock$Month == 5]),
                 min(final_apple_stock$Date[final_apple_stock$Month == 6]),
                 min(final_apple_stock$Date[final_apple_stock$Month == 7]))

month_labels <- c("March","April", "May", "June", "July")
vision_pro_release <- as.Date("2023-06-05")

pre_june_5 <- final_apple_stock[final_apple_stock$Date < vision_pro_release, ]
post_june_5 <- final_apple_stock[final_apple_stock$Date >=vision_pro_release, ]

axis.apple <- function(){tick_indices <- match(ticks, final_apple_stock$Date)
  axis(side = 1, at = tick_indices,labels = month_labels,las = 1) }

plot.apple <- function(x.range = c(1, 105),
                       y.range = range(final_apple_stock$Close))
  {plot.ts(pre_june_5$Close,
          main = "Daily APPL (Apple) Stock Closing Price",
          xlab = "Time
(in Months)",
ylab = "Price (in USD)",
type = "o",
xlim = x.range,
ylim = y.range,
pch = 20,
cex = 0.6,
xaxt = 'n')
```

```
  lines(x= 67:105, y = post_june_5$Close   , col = "red")
  axis.apple() }

plot.apple()
mtext("Pre & Post June 5th's Apple Vision Pro (AVP) Reveal")
legend("topleft",legend = c("Pre-AVP Announcement", "Post-AVP Announcemnt"),
col = c("black", "red"),pch = 20,lty = 1,bty = "n",cex = 0.8)
```



Now that we've cleaned up the data set, restricting to the dates of March 1st 2023 to July 18th, we can begin a general assessment.

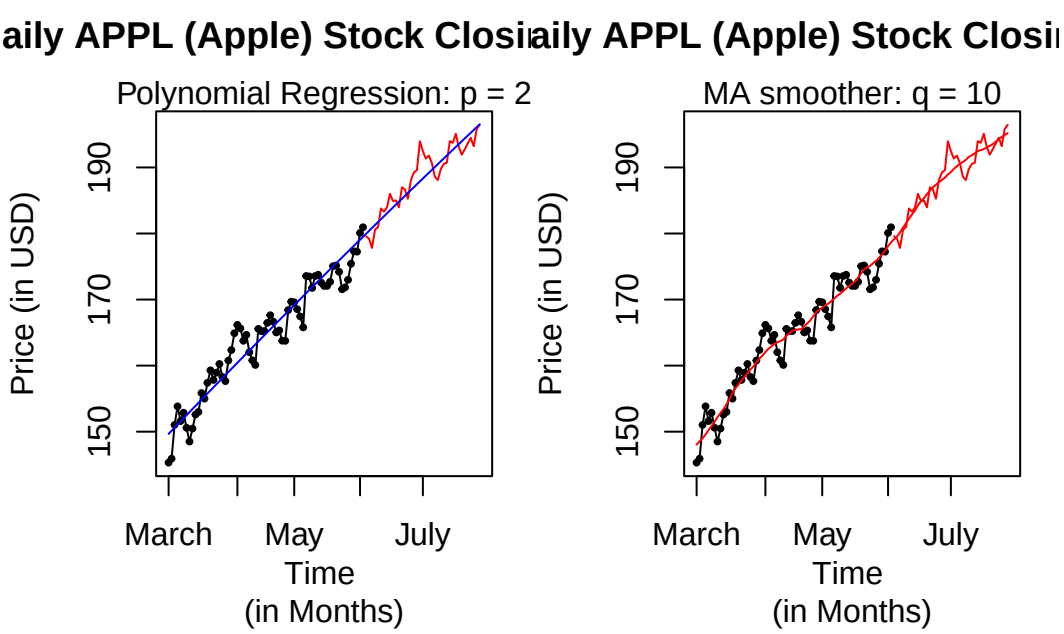
The data shows a global upward trend over time with a good bit of variance. The fluctuations in variance seem almost cyclical, suggesting the presence of seasonality.

We will now begin with trend estimation, comparing polynomial regression and MA smoothing methods to see which best fits our data.

Trend Estimation

```
m.pr <- trend(final_apple_stock$Close, p = 2)
m.ma <- smooth.ma(final_apple_stock$Close, q = 10)

par(mfrow = c(1,2), mar = c(4,4,4,1))
```

```r
plot.apple();
mtext("Polynomial Regression: p = 2")
lines(m.pr, col = "blue")

plot.apple(); mtext("MA smoother: q = 10")
lines(m.ma, col = "red")
```



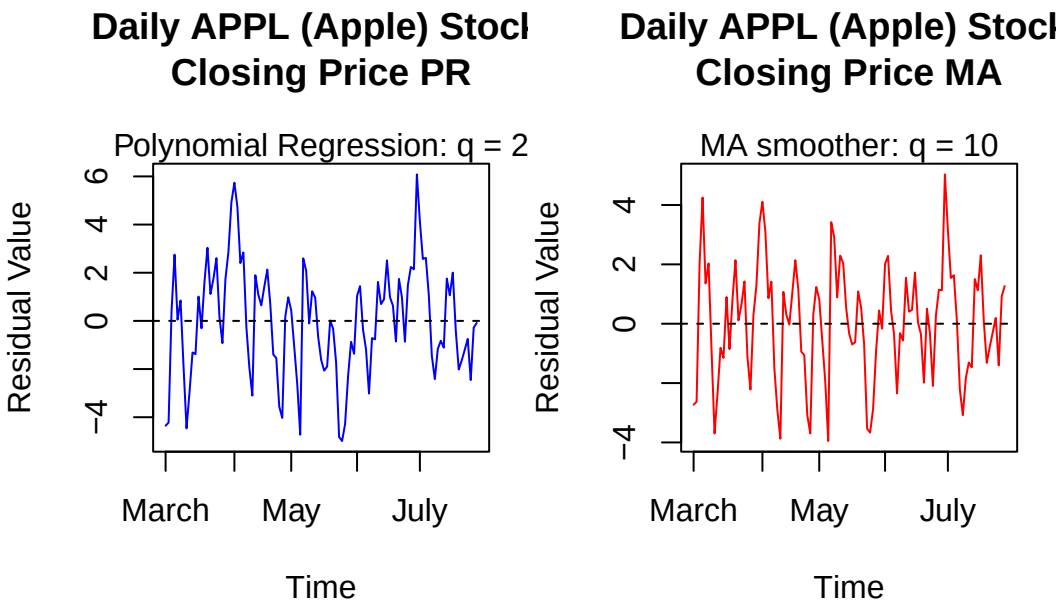
Compared to polynomial regression, MA smoothing seems to better capture the fluctuations in the data's trend over time.

Moving forward, we'll compare the data residuals for both methods.

Trend Estimation - Residuals

```r
par(mfrow = c(1,2), mar = c(4,4,6,1))

rm.pr <- final_apple_stock$Close - m.pr
plot.ts(rm.pr,
main = "Daily APPL (Apple) Stock
Closing Price PR",
ylab = "Residual Value",
```

```
xaxt = "n",
col = "blue")
axis.apple(); mtext("Polynomial Regression: q = 2")
abline(h = 0, lty = 2)

rm.ma <- final_apple_stock$Close - m.ma
plot.ts(rm.ma,
main = "Daily APPL (Apple) Stock
Closing Price MA",
ylab = "Residual Value",
xaxt = "n",
col = "red")
axis.apple(); mtext("MA smoother: q = 10")
abline(h = 0, lty = 2)
```



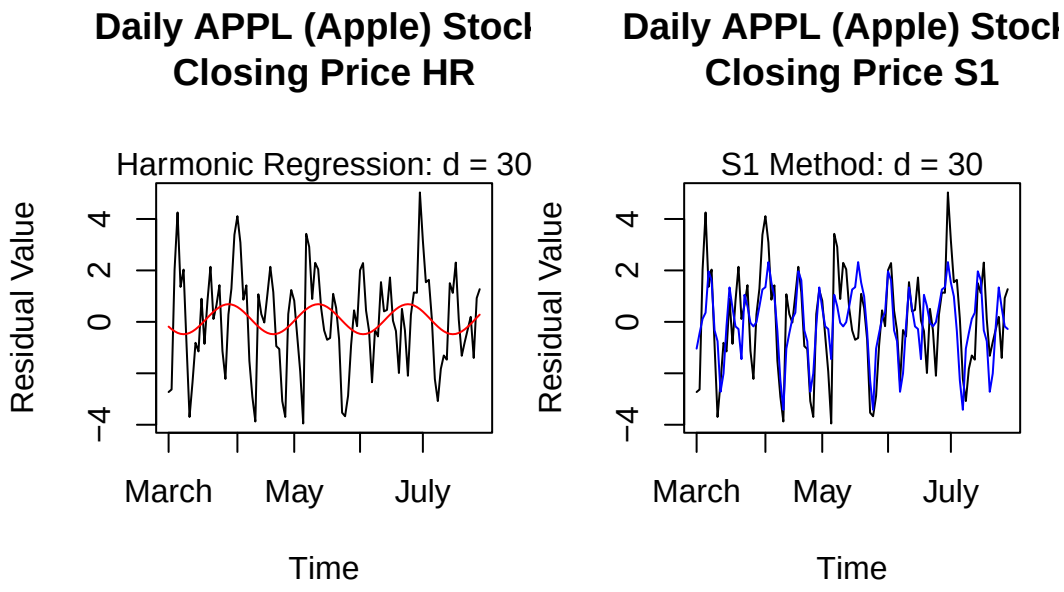
Looking at trend residual amplitudes, MA smoothing does a much better job smoothing out the data.

We will henceforth proceed with MA smoothing for our modelling.

Let's now take a look at seasonality within the data, this time comparing Harmonic Regression with the S1 Method

Seasonality Estimation - Harmonic Regression Vs. S1 Method

```
s.hr <- hr(rm.ma, d = 30)
s.s1 <- season(rm.ma, d = 30)

par(mfrow = c(1,2), mar = c(4,4,7,1))
plot.ts(rm.ma,
main = "Daily APPL (Apple) Stock
Closing Price HR",
xaxt = "n",
ylab = "Residual Value")
axis.apple(); mtext("Harmonic Regression: d = 30")
lines(s.hr, col = "red")

plot.ts(rm.ma,
main = "Daily APPL (Apple) Stock
Closing Price S1",
xaxt = "n",
ylab = "Residual Value")
axis.apple(); mtext("S1 Method: d = 30")
lines(s.s1, col = "blue")
```

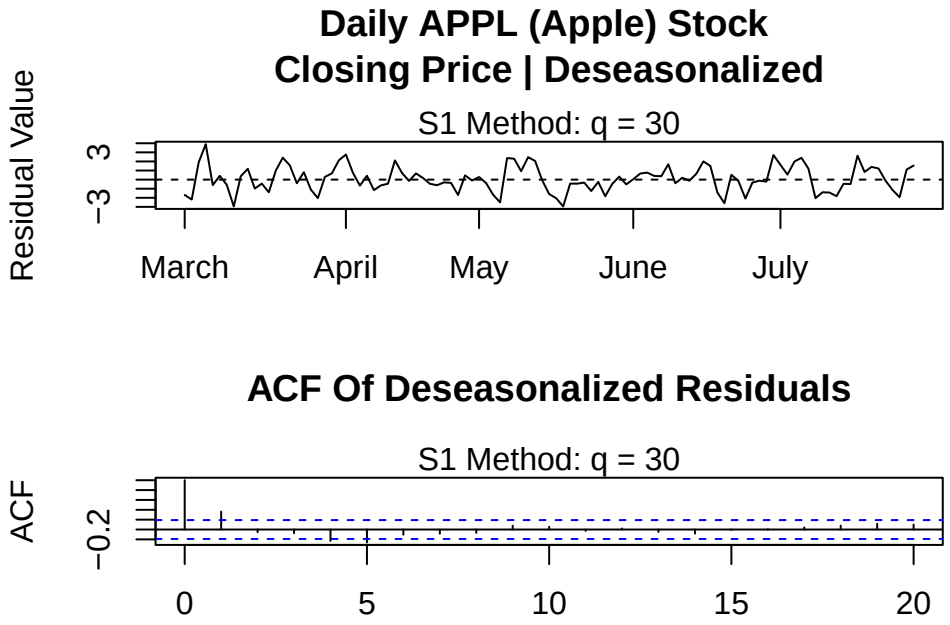


Given the number of seasonal components track to better track the data's function within a given month, its not surprise that harmonic regression does a poor job here compared to MA smoothing at estimating seasonality.

This being said, we can now look at ACF residuals for the S1 method, as well as the deseasonalized data.

Sample ACF - S1 Method

```
par(mfrow = c(2,1), mar = c(2,5,5,2))
y.s1 <- rm.ma - s.s1
plot.ts(y.s1,
main = "Daily APPL (Apple) Stock
Closing Price | Deseasonalized",
xaxt = "n",
ylab = "Residual Value")
axis.apple(); mtext("S1 Method: q = 30"); abline(h = 0, lty = 2)

acf(y.s1,
main = "ACF Of Deseasonalized Residuals",
xlab = "Lag")
mtext("S1 Method: q = 30")
```



Removing the seasonality from the data shows that there is a subtle amount of motif repetition in the data. This would suggest that the data does have a seasonal structure, if only minimal.

The ACF shows some strong correlations with time, particularly at lags 0, 1, 5, 4, and 2, in order of magnitude.

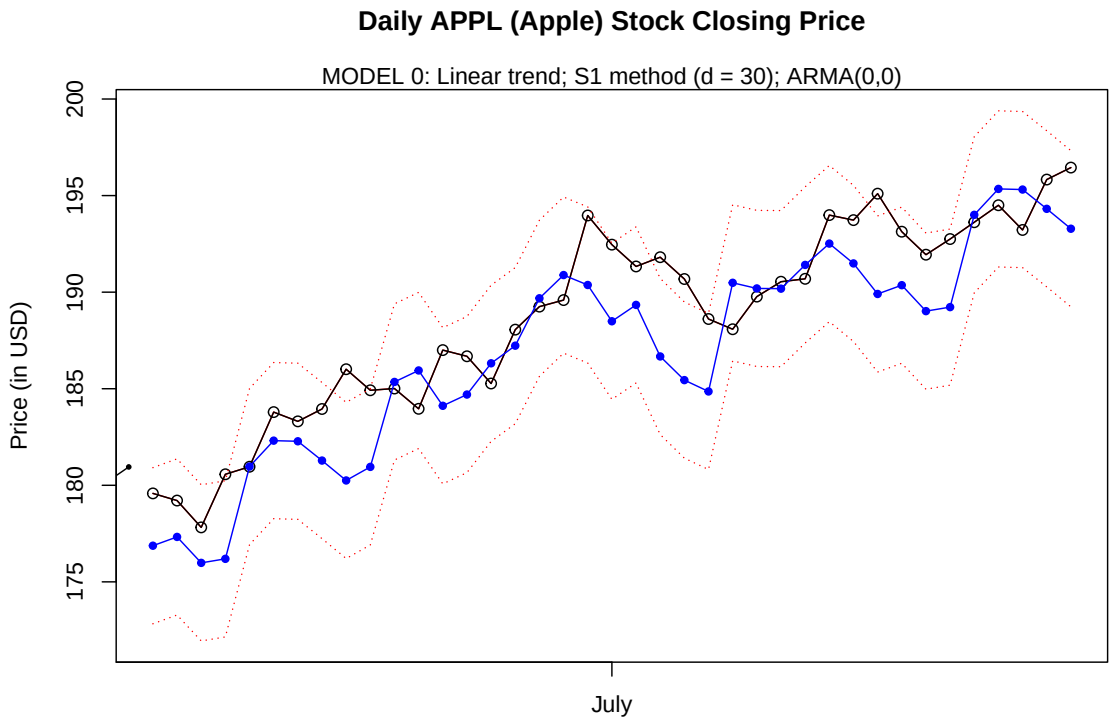
We can now move on to determining the ideal parameters for an ARMA process.

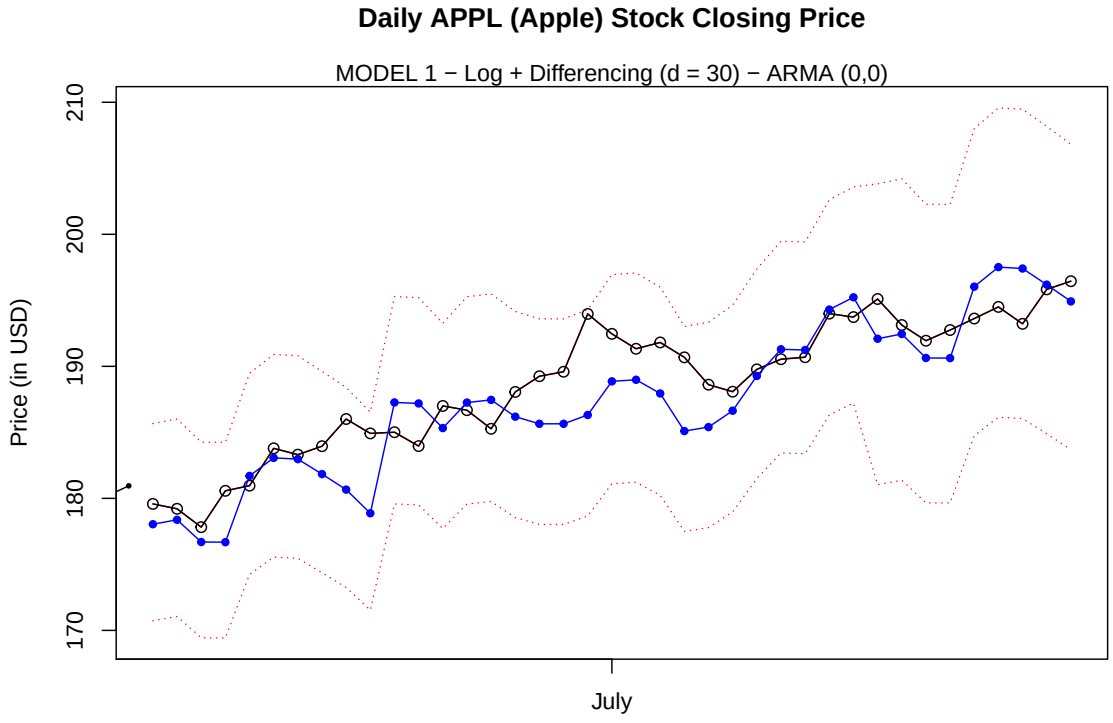
Modelling the data

```
#no success with hr keep getting upward lines
#had to include march data to set seasonality to 30
#

M.0 <- c("trend" , 1, "season", 30)
M.1 <- c("log" , "diff", 30)

R.0 <- Resid(pre_june_5$Close, M = M.0)
R.1 <- Resid(pre_june_5$Close, M = M.1)


A.0 <- arma(x = R.0, p = 0, q = 0)
A.1 <- arma(x = R.1, p = 0, q = 0)


fc.0 <- forecast(pre_june_5$Close, M = M.0, a = A.0, h = 39, opt = 0)
fc.1 <- forecast(pre_june_5$Close, M = M.1, a = A.1, h = 39, opt = 0)


plot.forecast <- function(fc){ # input prediction
  points(x = 67:105,
         y = fc$pred,
         col = "blue", type = "o", pch = 20)

lines(67:105, fc$l, col = "red", lty = 3)
lines(67:105, fc$u, col = "red", lty = 3)
}
```

```
par(mfrow = c(2,1), mar = c(2,5,5,2))

#leaves model 5 times
plot.apple(x.range = c(67,105),
```

```
          y.range = range(post_june_5$Close, fc.0$l, fc.0$u))
mtext("MODEL 0: Linear trend; S1 method (d = 30); ARMA(0,0)")
points(x = 67:105, y = post_june_5$Close, type = "o")
plot.forecast(fc.0)


# --- Model 1 - leaves model  0 times
plot.apple(x.range = c(67,105),
          y.range = range(post_june_5$Close,fc.1$l,fc.1$u))
mtext("MODEL 1 - Log + Differencing (d = 30) - ARMA (0,0)")

points(x = 67:105, y = post_june_5$Close, type = "o")
plot.forecast(fc.1)
```


Daily APPL (Apple) Stock Closing Price
MODEL 0: Linear trend; S1 method (d = 30); ARMA(0,0)
Price (in USD)
175
180
185
190
195
200
July


Daily APPL (Apple) Stock Closing Price
MODEL 1 − Log + Differencing (d = 30) − ARMA (0,0)
Price (in USD)
170
180
190
200
210
July

Here we've created 2 models with 30 seasonal components to capture the data fluctuations within a given month. Both forecasting models have follow an ARMA(0,0) process, and focus on forecasting data on and past July 5th, the Apple Vision Pro's release date.

From here we will use AIC diagnostics to determine each models ideal ARMA parameters and decide which model fits our data best.

AIC Diagnostics

```
aic.0 <- aic.1 <- matrix(data = 0, nrow = 6, ncol = 6)


for (p in 0:5) for (q in 0:5) {

  R0.fit <- arima(R.0, order = c(p,0,q))
  R1.fit <- arima(R.1, order = c(p,0,q),
                  optim.control = list(maxit = 1000))


  aic.0[p+1,q+1] <- R0.fit$aic
  aic.1[p+1,q+1] <- R1.fit$aic
}
```

```
Warning in log(s2): NaNs produced
Warning in log(s2): NaNs produced
```

```
rownames(aic.0) <- rownames(aic.1) <- paste("p =",0:5)
colnames(aic.0) <- colnames(aic.1) <- paste("q =",0:5)


kable(x = aic.0, caption = "AIC for ARMA(p,q) models; Classical Model 0")
```

Table 1: AIC for ARMA(p,q) models; Classical Model 0

	q = 0	q = 1	q = 2	q = 3	q = 4	q = 5
p = 0	286.8550	266.4188	266.2094	267.4025	267.5996	264.1259
p = 1	264.7204	265.5030	267.3086	269.0388	271.0491	266.0790
p = 2	265.8262	267.3251	269.2929	266.6895	266.2413	267.8681
p = 3	267.5503	269.2414	266.9127	267.2065	266.5881	269.5981

	q = 0	q = 1	q = 2	q = 3	q = 4	q = 5
p = 4	268.7997	270.7022	266.0238	270.3619	271.0770	270.2670
p = 5	270.1515	269.8395	267.9994	269.1956	270.8271	270.0260

```
kable(x = aic.1, caption = "AIC for ARMA(p,q) models; Classical Model 3")
```

Table 2: AIC for ARMA(p,q) models; Classical Model 3

	q = 0	q = 1	q = 2	q = 3	q = 4	q = 5
p = 0	-170.6687	-185.0930	-186.8603	-185.8302	-187.8516	-187.0254
p = 1	-190.5062	-188.7552	-187.2519	-185.5740	-183.7027	-186.1895
p = 2	-188.6800	-186.9258	-185.5827	-183.2806	-186.1895	-184.7573
p = 3	-187.0221	-185.1306	-188.5860	-185.2393	-184.2404	-187.6335
p = 4	-185.1658	-183.2139	-188.2738	-185.1587	-183.9249	-185.6335
p = 5	-183.7792	-181.6787	-183.6489	-185.6684	-187.9122	-179.9127

```
print(paste("Index of Smallest AIC (Model 0):", which.min(aic.0)))
```

```
[1] "Index of Smallest AIC (Model 0): 31"
```

```
print(paste("Index of Smallest AIC (Model 1):", which.min(aic.1)))
```

```
[1] "Index of Smallest AIC (Model 1): 2"
```

```
fit.past.0.aic  <- arima(R.0, order = c(1,0,1))

display <- cbind(fit.past.0.aic$coef,
                 sqrt(diag(fit.past.0.aic$var.coef)))

kable(display,
      col.names = c("Estimated Coefficient", "Standard Error"),
      caption = "Model 0 - ARMA cofficients (from AIC selection)")
```

Table 3: Model 0 - ARMA cofficients (from AIC selection)

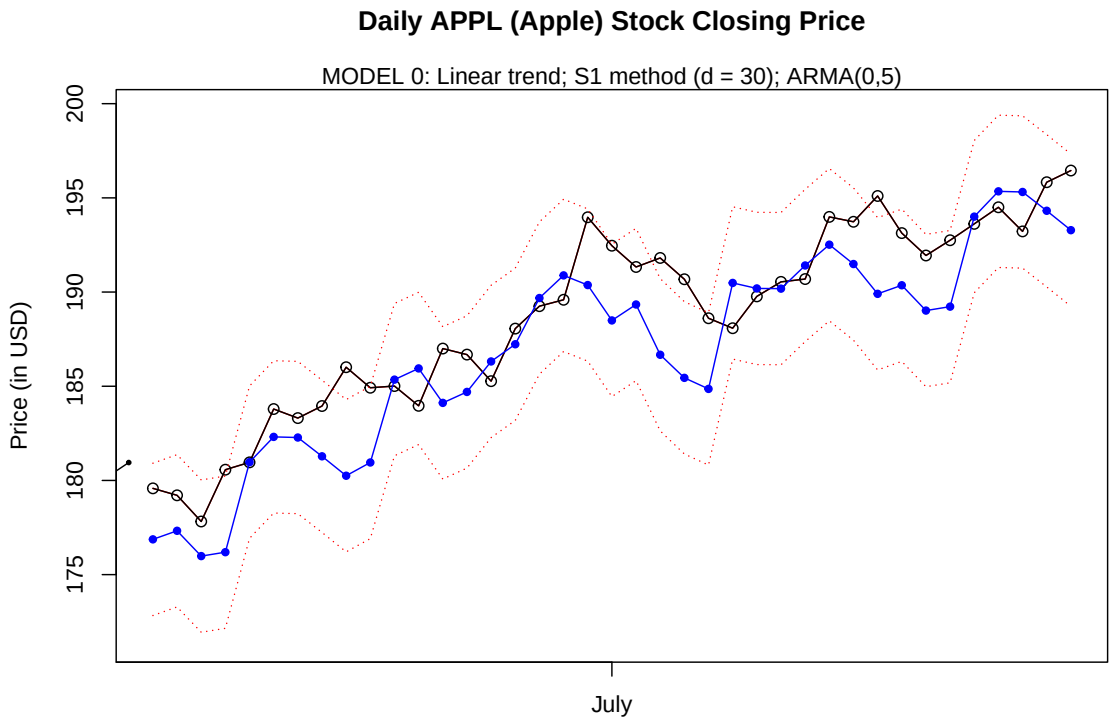
	Estimated Coefficient	Standard Error
ar1	0.3911750	0.2281635
ma1	0.2818430	0.2344537
intercept	-0.0397419	0.4347788

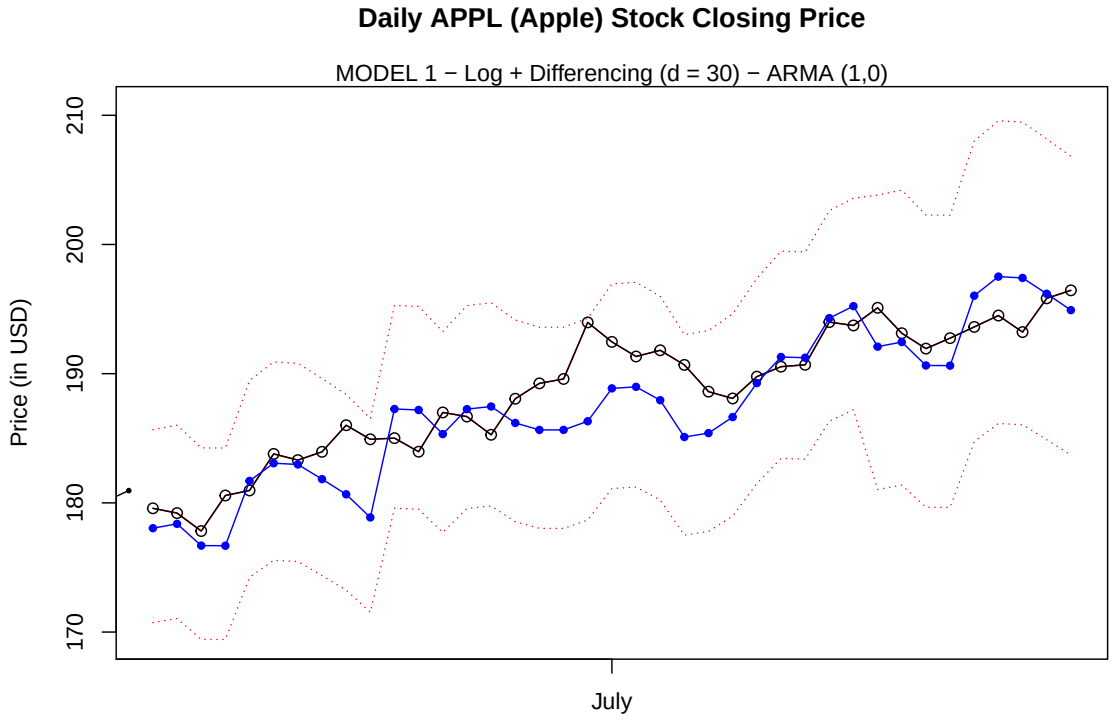
Model	p	q	AIC	Noise variance σ^2
0	0	5	264.1259	2.23
1	1	0	-190.5062	0.0001766

The table above summarizes the results from our AIC diagnostics. From here we can compare both models using the suggested parameters.

```
A.0.2 <- arma(x = R.0, p = 0, q = 5)
A.1.2 <- arma(x = R.1, p = 1, q = 0)


fc.0.2 <- forecast(pre_june_5$Close, M = M.0, a = A.0.2, h = 39, opt = 0)
fc.1.2 <- forecast(pre_june_5$Close, M = M.1, a = A.1.2, h = 39, opt = 0)
```

```
par(mfrow = c(2,1), mar = c(2,5,5,2))


plot.apple(x.range = c(67,105),
           y.range = range(post_june_5$Close, fc.0.2$l, fc.0.2$u))
mtext("MODEL 0: Linear trend; S1 method (d = 30); ARMA(0,5)")
points(x = 67:105, y = post_june_5$Close, type = "o")
plot.forecast(fc.0)


plot.apple(x.range = c(67,105),
           y.range = range(post_june_5$Close,fc.1.2$l,fc.1.2$u))
mtext("MODEL 1 - Log + Differencing (d = 30) - ARMA (1,0)")

points(x = 67:105, y = post_june_5$Close, type = "o")
plot.forecast(fc.1)
```


Daily APPL (Apple) Stock Closing Price
MODEL 0: Linear trend; S1 method (d = 30); ARMA(0,5)
Price (in USD)
175
180
185
190
195
200
July


Daily APPL (Apple) Stock Closing Price
MODEL 1 − Log + Differencing (d = 30) − ARMA (1,0)
Price (in USD)
170
180
190
200
210
July

Model 0 utilizes a linear trend and the S1 Method. Manipulating the plot for further clarity, we found that the predicted data (blue) exits our confidence interval 5 times. 0.05 x 30 = 1.5 As such, our confidence interval fails to capture the data less than the desired 95% of the time.

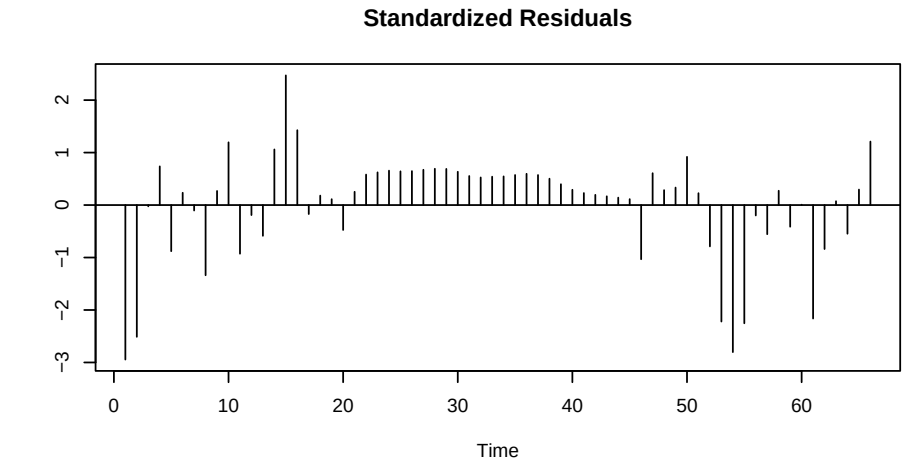
Model 1 utilizes a log and differencing. Once again through manipulating the plot, we found that the predicted data (blue) does not exit our confidence interval.

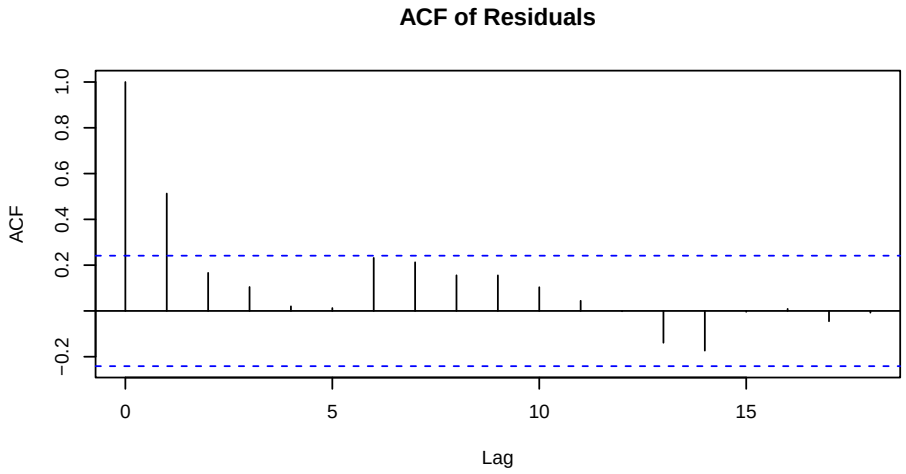
Despite this, we can see that Model 0 better constrains and captures the shape of the data. For this reason, Model 0 is preferred.

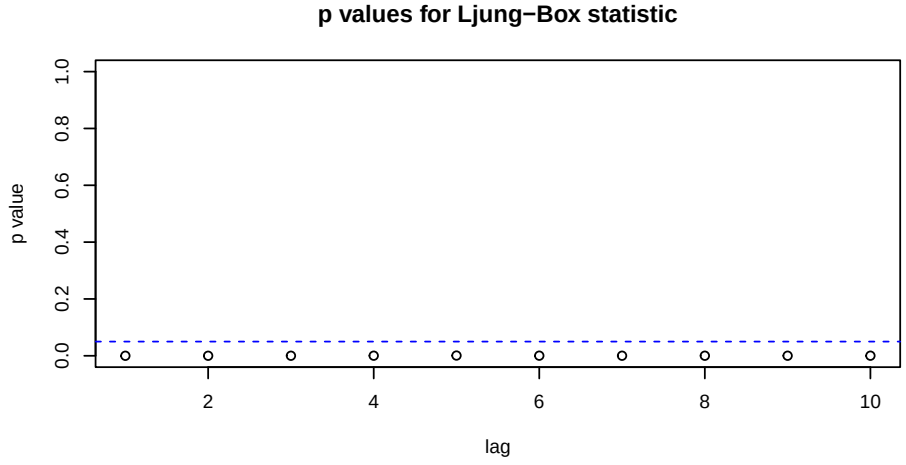
We will can now proceed with a residual analysis for Model 0.

Comparing Model 0 Residuals - ARMA (0,0) Vs. ARMA (0,5)

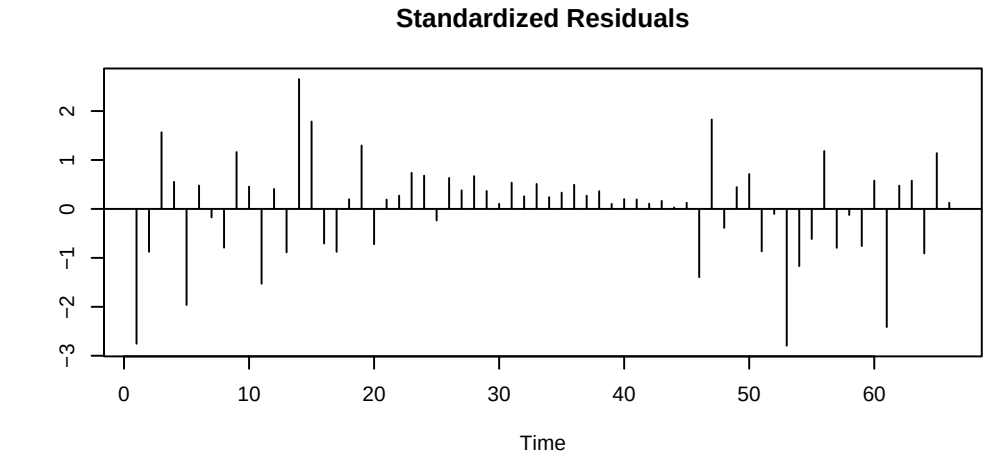
```
tsdiag(arima(R.0, order = c(0,0,0)))
```

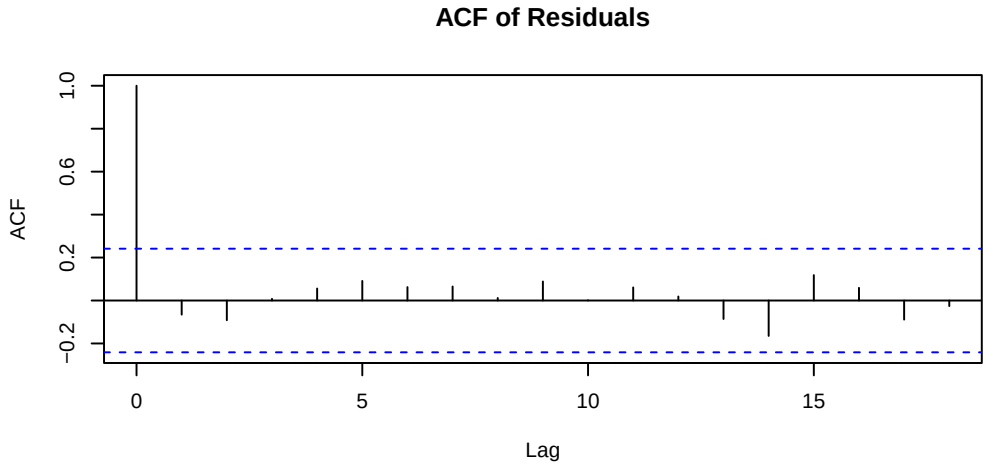

Standardized Residuals
2
1
0
−1
−2
−3
0
10
20
30
40
50
60
Time

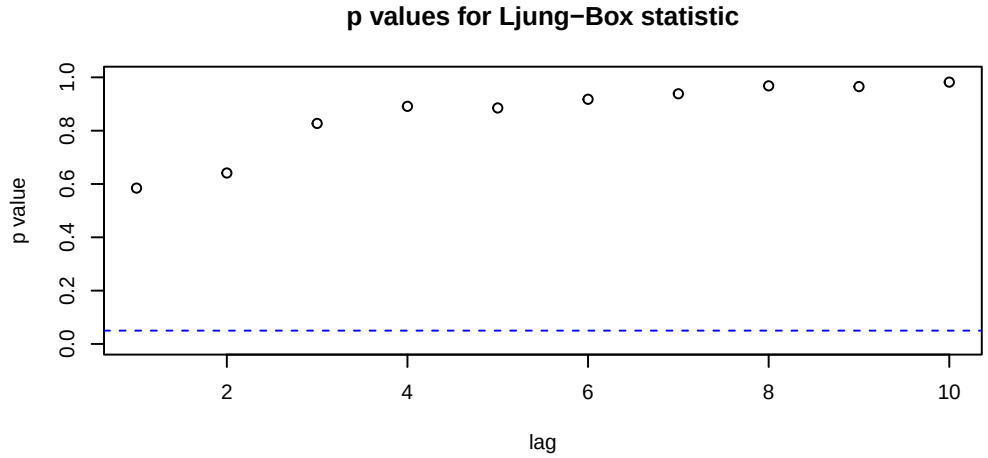

ACF of Residuals
ACF
1.0
0.8
0.6
0.4
0.2
−0.2
0
5
10
15
Lag


p values for Ljung−Box statistic
p value
1.0
0.8
0.6
0.4
0.2
0.0
2
4
6
8
10
lag

```r
tsdiag(arima(R.0, order = c(0,0,5)))
```


Standardized Residuals
−3
−2
−1
0
1
2
0
10
20
30
40
50
60
Time


ACF of Residuals
ACF
−0.2
0.2
0.6
1.0
0
5
10
15
Lag


p values for Ljung−Box statistic
p value
0.0
0.2
0.4
0.6
0.8
1.0
2
4
6
8
10
lag

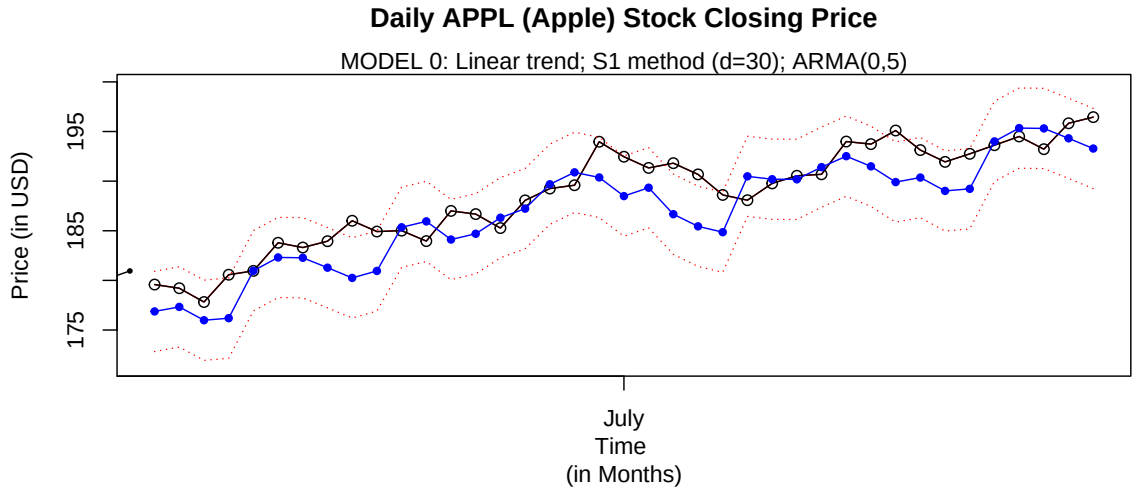
Looking at both ARMA(0,0) and ARMA (0,5) process residuals, we can see that our new ARMA(0,5) process better randomizes the data. There a no major trends or cycles present.

Summarizing our Model (Final Formula)

Model 0		
Classical Model	X_t	$X_t = m_t + s_t + Y_t$
ARMA-modelled Residuals	Y_t	$\phi(B)Y_t = \theta(B)Z_t$
Innovations (noise)	Z_t	$Z_t \sim wn(\sigma^2 = 2.23)$ |
Trend estimate	m_t	Line of best fit (linear regression)
Seasonality estimate	s_t	Periodic function derived from 30-point moving average (S1 Method)
ARMA(p,q) selection	AIC	
AR polynomial	$\phi(z)$	$1 - 0.3911750$
MA polynomial	$\theta(z)$	$1 - 0.2818430$

Conclusion

```
plot.apple(x.range = c(67,105),
           y.range = range(post_june_5$Close, fc.0.2$l, fc.0.2$u))
mtext("MODEL 0: Linear trend; S1 method (d=30); ARMA(0,5)")
points(x = 67:105, y = post_june_5$Close, type = "o")
plot.forecast(fc.0)
```



Our final plot seems to mostly capture our data's behavior.

In regard to our hypothesis, our model suggests unexpected upward trends in over time, in regard to the real data. This real data seems to somewhat be mostly above our predicted data. This would add some weight to our initial theory that the Apple Vision Pro's announcement stimulated Apple shareholders to further invest in the company's stock. It also suggests that the rise of Apple's stock was projected to rise to the real data's levels, albeit without notable price hikes showcased in the 30 days following the Vision Pro's reveal.

Some issues in our interpretation lie in the many exogenous factors that affect a companies stock price, such as the companies performance, public perception, and private dealings to name a few.

Overall, this study showcases time series' ability to predict and analysis data fluctuations over time, with the predicted data mostly being coherent with the real data based on 4 months worth of prior data.