STAT 153 Project: COVID-19 Forecasting

Group: Andre Sha, Alice Young

May 2, 2020

https://medium.com/@aliceyoung/covid-19-in-timeseria-558af175cfee

Executive Summary:

For our report, we chose to forecast the next 10 days of COVID-19 cases in fictional Timeseria. We then sought to make predictions using quintic parametric AR(3), MA(2), and ARIMA(1,2,0) models. Our forecast uses the first one and is based upon the assumption that there will be no changes to public health policies in Timeseria within the next 10 days.

1 Exploratory Data Analysis

Our dataset *covid.csv* contains the total detected COVID-19 counts in Timeseria from the first day of the first diagnosis until present.

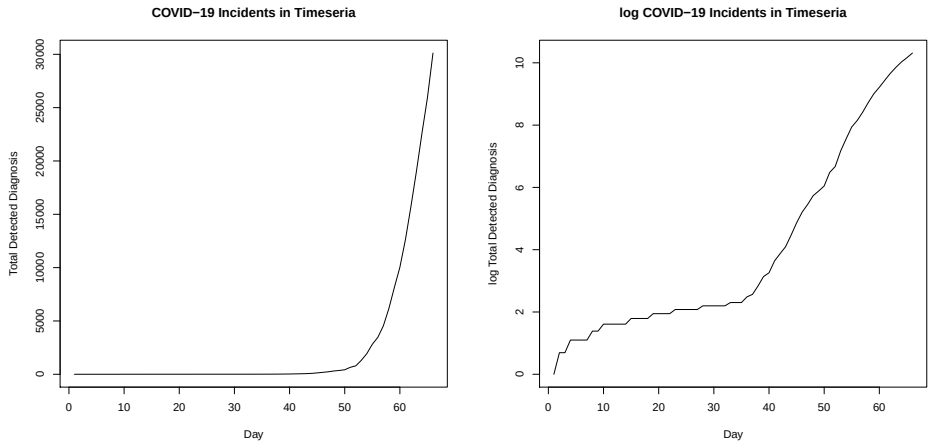


Figure 1: As we can see from this raw data (L) , the plot is not homoskedastic. Thus, we applied a log variance stabilizing function to account for the seemingly exponential increases in variance (R). While the log transform creates relative homoskedasticity, it is clear that stationarity does not hold due to the increasing mean over time.

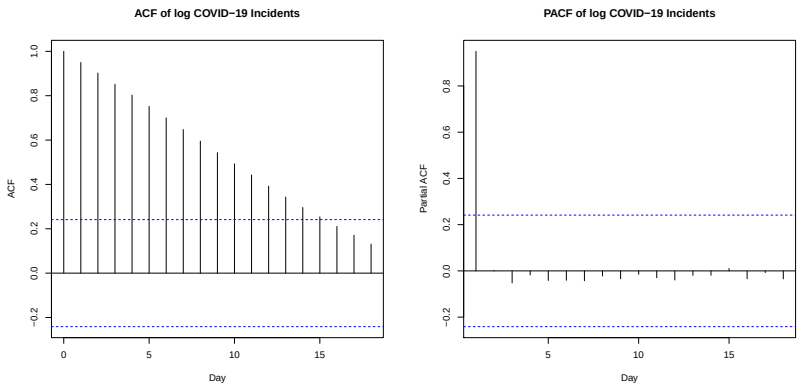


Figure 2: Corresponding ACF and PACF of log COVID-19 Incidents

2 Models Considered

After our data exploratory analysis, we decided upon two methods to model our to time series, with 1) differencing our model to eliminate the trend and 2) smooth the series using polynomial regression. Afterwards, we regrouped and decided upon trying out an ARIMA model as we noted through our independent work that the series displays autoregressive properties after differencing the data.

2.1 Andre's Model

To eliminate the trend from our log graph, I took second differences.

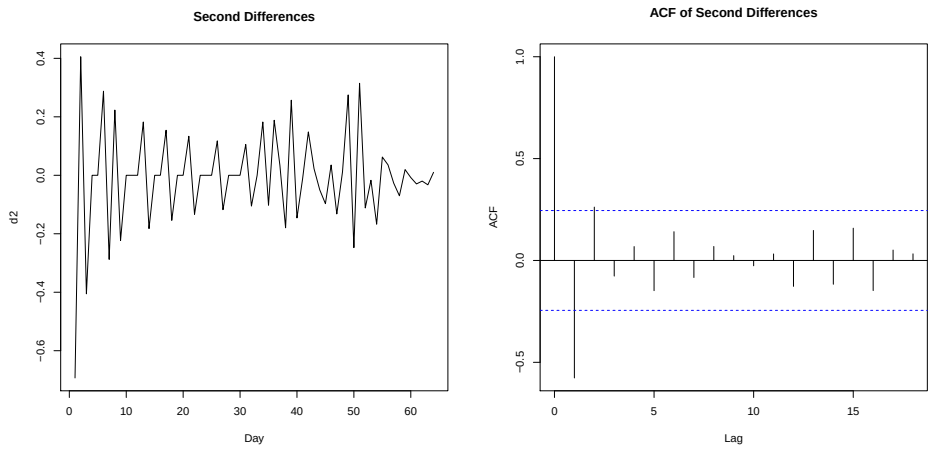


Figure 3: Second difference of the log of Covid-19 Cases (L) and its corresponding ACF (R)

As we can see from the ACF of our second differencing, the series is roughly stationary. I also noted that since there are two statistically significant spikes at lags 1 and 2 followed by non-significant values for future lags, I tried testing out an MA(2) model.

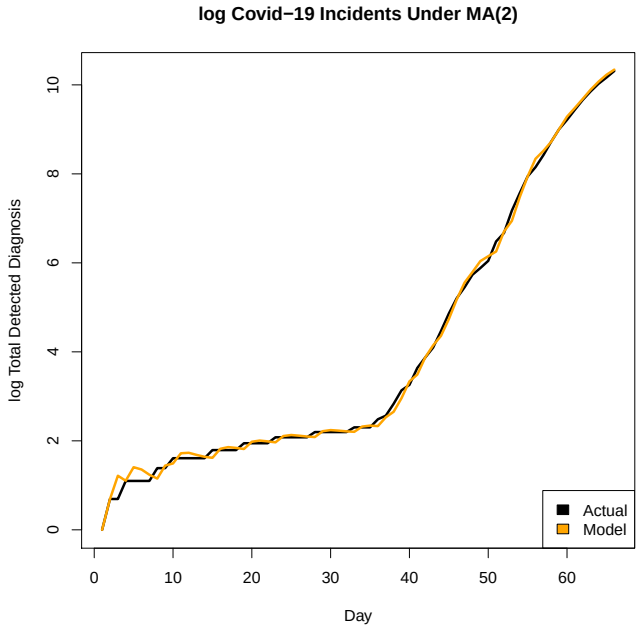


Figure 4: Andre's 2nd Difference and MA(2)Model

By taking a look at the residuals its ACF, since they were relatively stable and looked like white noise, I thought my model was a good fit.

2.2 Alice's Model

The trend of the log of corona counts looks like an odd-degree polynomial, so I decide to use a degree-5 polynomial. Since the residuals do not look like white noise, I need ARMA. To determine the parameters of the ARMA model, I look at the ACF and PACF plots of the residuals.

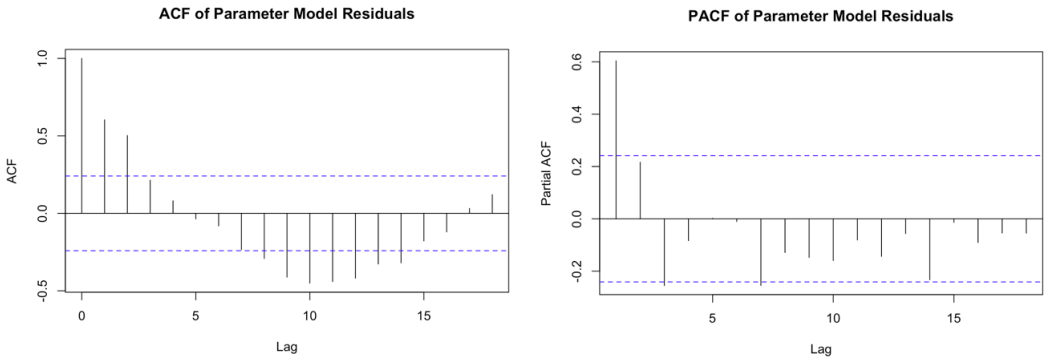


Figure 5: In the ACF plot on the left, there is a "rollercoaster" (exponential decrease in the heights of the bars). In the PACF plot on the right, there are 3 main spikes, followed by a majority of the bars falling within the blue confidence bands.

The exponential decrease in the ACF indicates that there is autoregressive properties. Because there are 3 spikes in the PACF, I conclude that p=3. There does not seem to be MA because the PACF is never 0 for lags greater than a certain value. So, I fit an AR(3) model on top of the degree-5 polynomial.

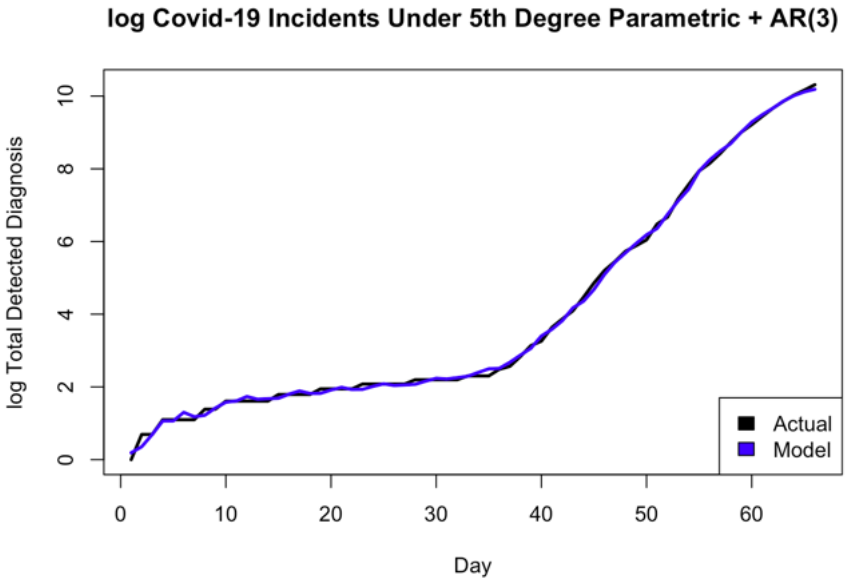


Figure 6: Alice's Polynomial Regression + AR(3) Model.

I then look at the residuals and ACF of this new model - the residuals appear relatively stable and the ACF looks like white noise, so I am done.

2.3 The Group Model

As a group, we opted to use an ARIMA(1,2,0) model because the ACF of the log of the counts is exponentially decreasing (showing autoregressive properties), and the PACF has only 1 spike.

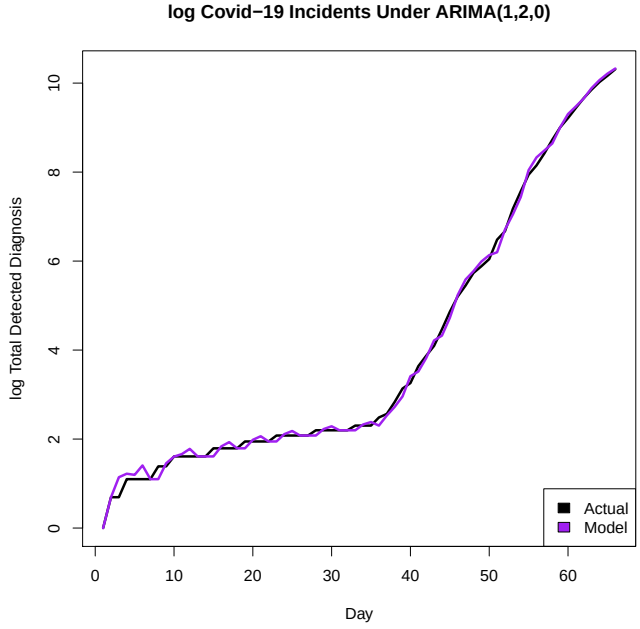


Figure 7: Our group model

3 Model Comparison and Selection

We compared our models using RMSE, AIC, and BIC, shown below in Table 1.

Model Name	Description	RMSE	AIC	BIC
Alice	Quintic polynomial and AR(3)	0.096	-111.4	-100.4
Andre	Second differences and MA(2)	0.125	-75.4	-68.9
Group	ARIMA(1,2,0)	0.123	-79.8	-75.4

Table 1: These are the in-sample RMSE's, AIC's, and BIC's for our models of interest.

We want a model that has low RMSE, AIC, and BIC. Since Alice's model has the lowest RMSE while also having lost the least amount of information (AIC, BIC), we choose this to be our final model.

4 Results

Our model of choice is AR(3) defined in equation (1).

$$(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3) X_t = W_t \tag{1}$$

4.1 Estimation of model parameters

Our model parameters are as follows (Table 2).

Parameter	Estimate	(s.e)
ϕ_1	0.5644	(0.1295)
ϕ_2	0.4524	(0.1583)
θ_1	-0.4127	(0.1334)

Table 2: These are our parameter estimates and corresponding standard errors for the AR model in equation 1.

So, our final model is defined in equation (2).

$$(1 - 0.5644B - 0.4524B^2 + 0.4127B^3) X_t = W_t \tag{2}$$

4.2 Prediction

Because Timeseria is struggling with the novel coronavirus, we forecast the total COVID-19 case count for the next 10 days to help the country prepare appropriately and at the right scale. We use their total detected COVID-19 counts from the first day of the first diagnosis until present to do this. Our projection is depicted below in Figure 5.

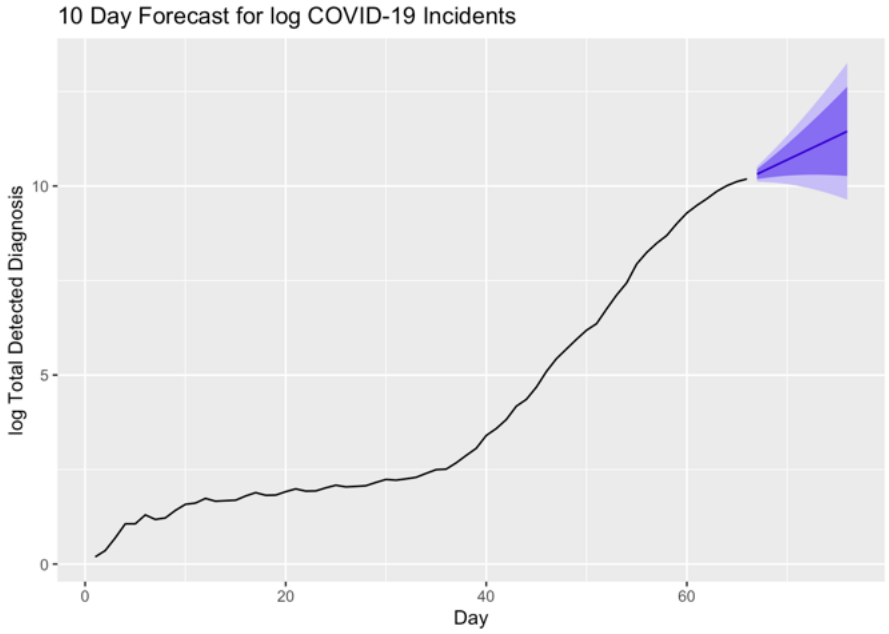


Figure 8: This is our forecast of the log of the number of COVID-19 incidents for the next 10 days in Timeseria. The dark blue line is our forecast, and the shaded dark and light blue regions are the 80 and 95 percent prediction intervals, respectively.

Even though we included prediction intervals to allow for a range of forecasts, there is still uncertainty regarding the growth of COVID-19 in Timeseria. We only looked at data from 66 time points, which is a relatively small sample size. Also, the number of COVID-19 cases relies heavily on external factors, such as the amount of testing available and the strictness of social distancing policies that Timeseria has. Thus, only if the trend continues as it did without any external factors will our predictions be pretty accurate.

Stat 153 Project

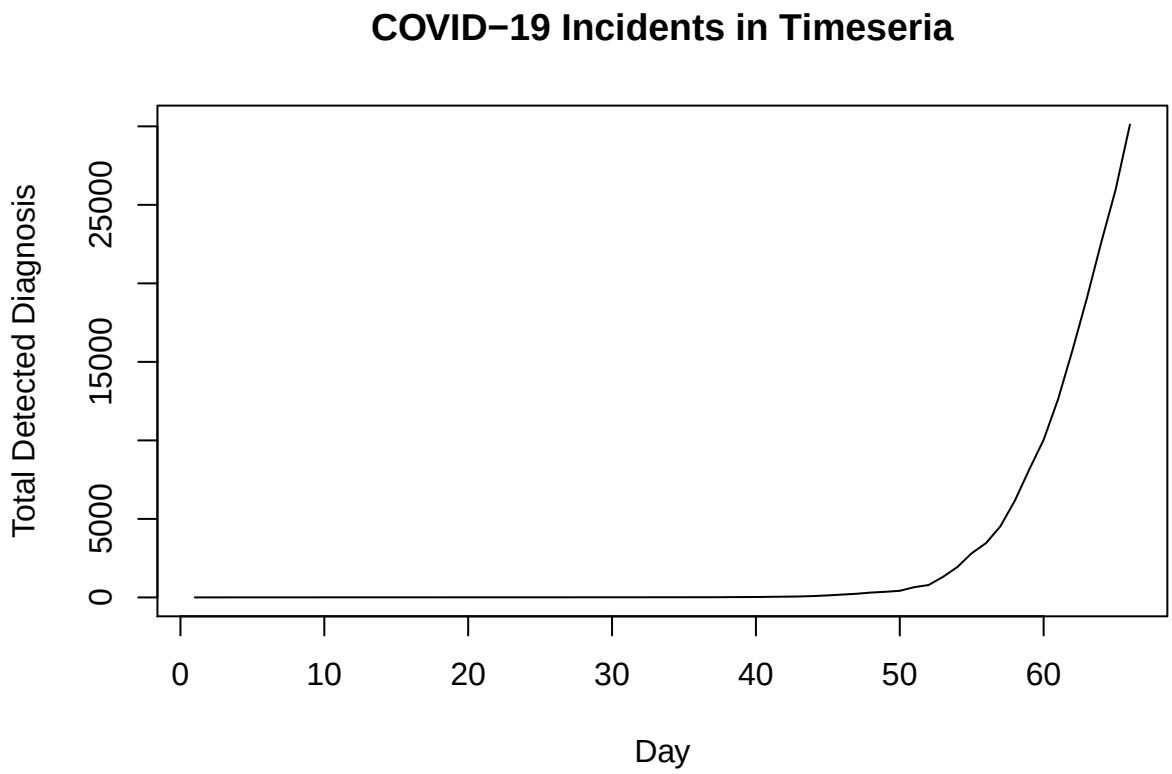
Alice and Andre

4/25/2020

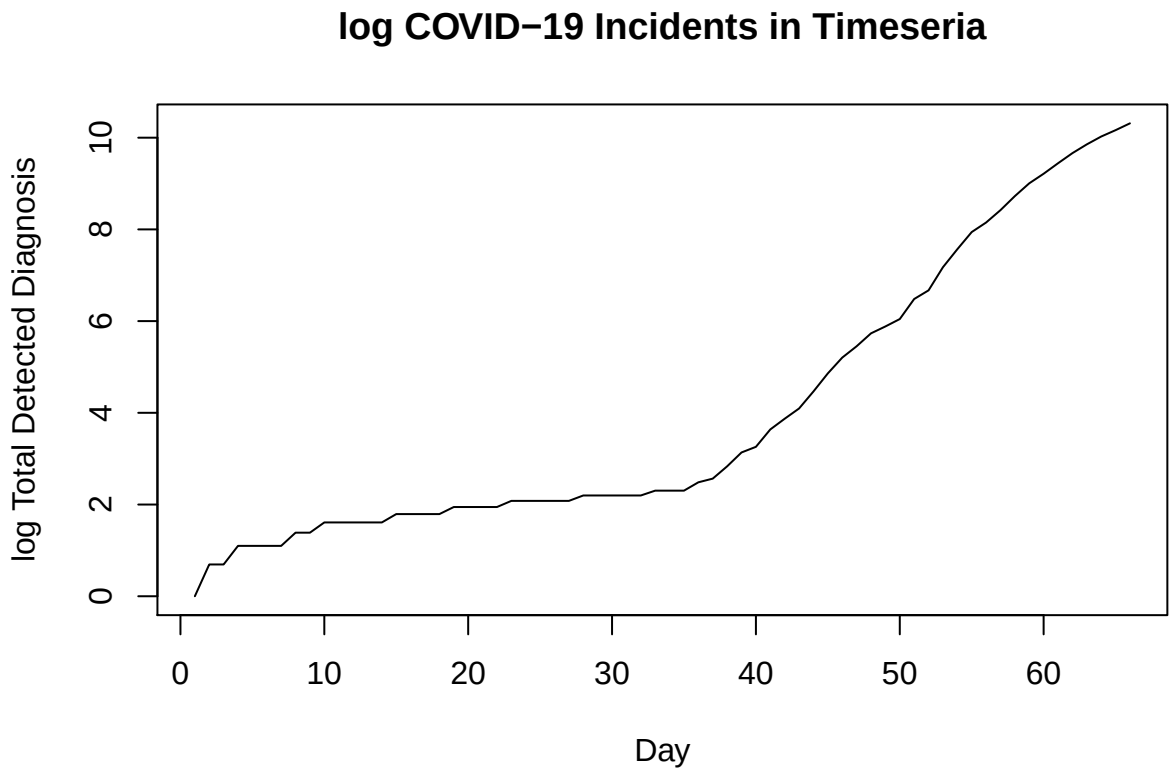
APPENDIX.

1. Exploratory Data Analysis

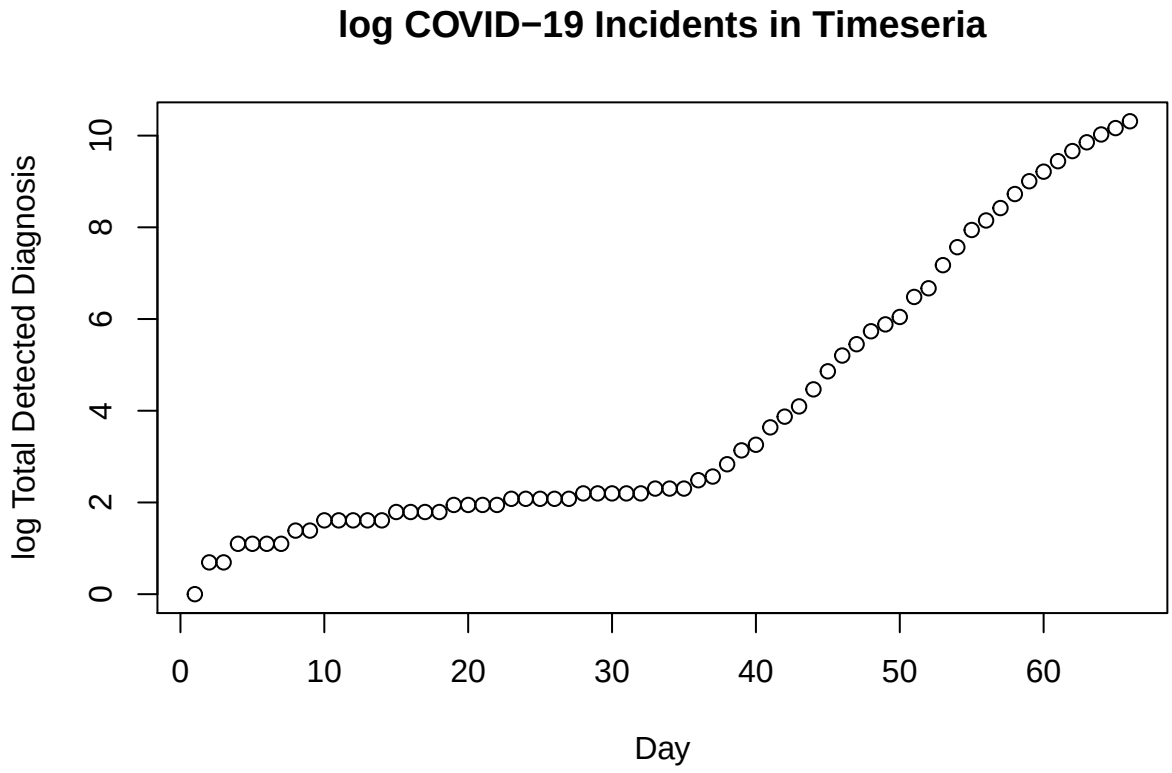
```
corona <- read.csv("covid19.csv")

plot(corona, main = 'COVID-19 Incidents in Timeseria',
     ylab = 'Total Detected Diagnosis',
     xlab = 'Day',type='l')
```




```
#As we can see from this raw data, the plot is not homoskedastic. Thus we will use a log transform.

log_corona = log(corona$Count)

#We used the log() transform for the exponential increases in variance.


plot(log_corona, main = 'log COVID-19 Incidents in Timeseria',
     ylab = 'log Total Detected Diagnosis',
     xlab = 'Day',type='l')
```



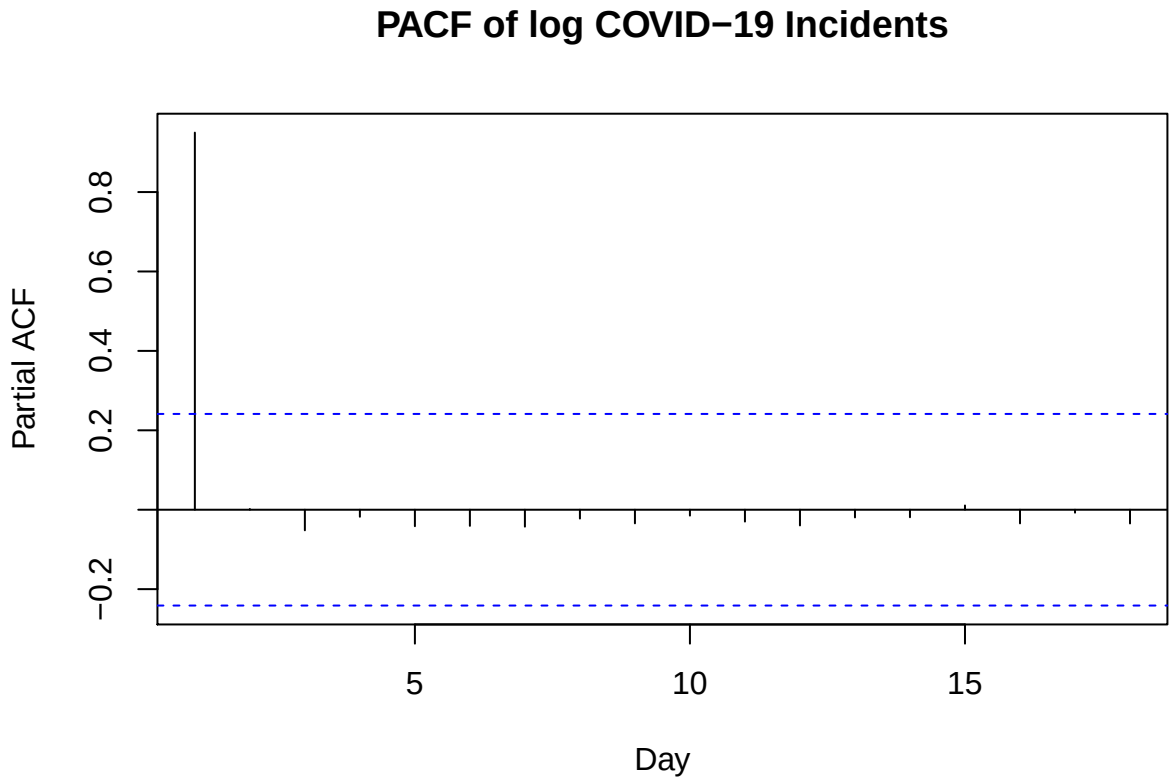

```
plot(log_corona, main = 'log COVID-19 Incidents in Timeseria',
     ylab = 'log Total Detected Diagnosis',
     xlab = 'Day')
```



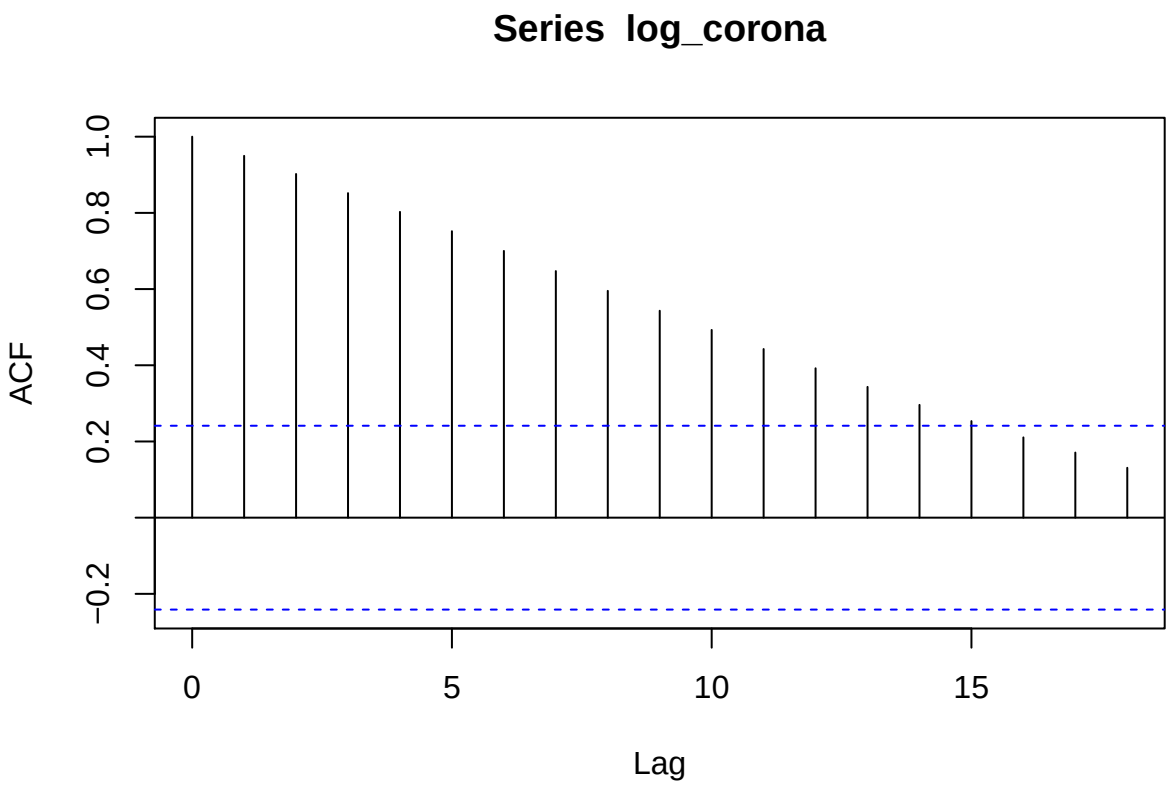
2. Models Considered

2.1: Alice's Model: Parametric Degree-5 Polynomial + AR(3)

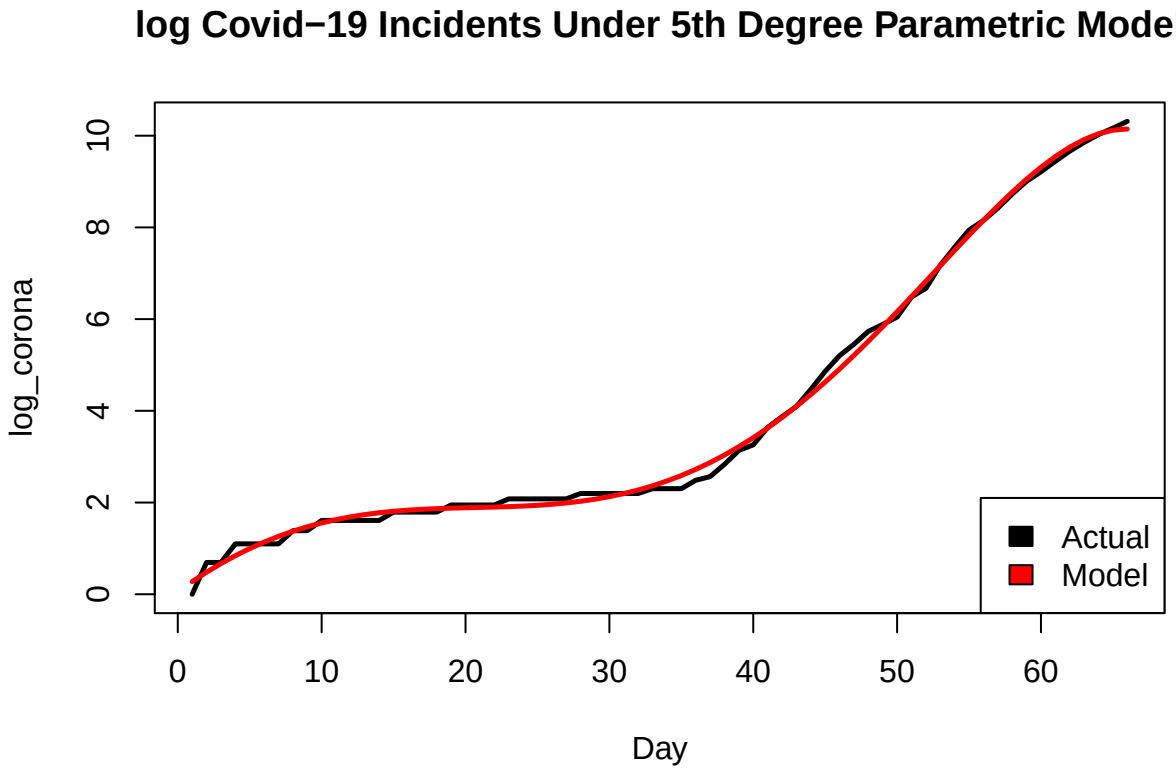
```
pacf(log_corona, main = "PACF of log COVID-19 Incidents", xlab = "Day")
```



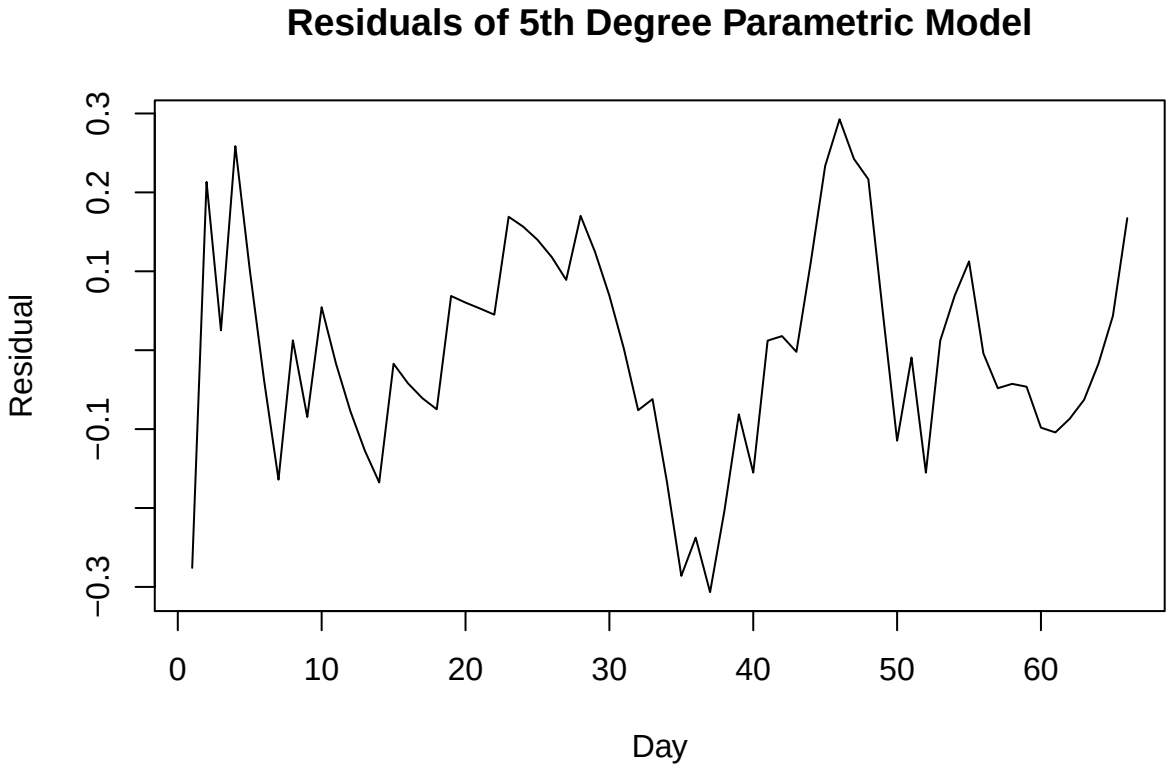

```
acf(log_corona)
```




```
# alice

param_model <- lm(log(corona$Count) ~ poly((corona$Day), 5))

plot(log_corona, type = 'l', main = 'log Covid-19 Incidents Under 5th Degree Parametric Model',
     xlab = 'Day', lwd = 2.5)
points(corona$Day, param_model$fitted.values, col= 'red', type = 'l', lwd = 2.5)
legend("bottomright", c("Actual", "Model"), fill = c("black", "red"))
```



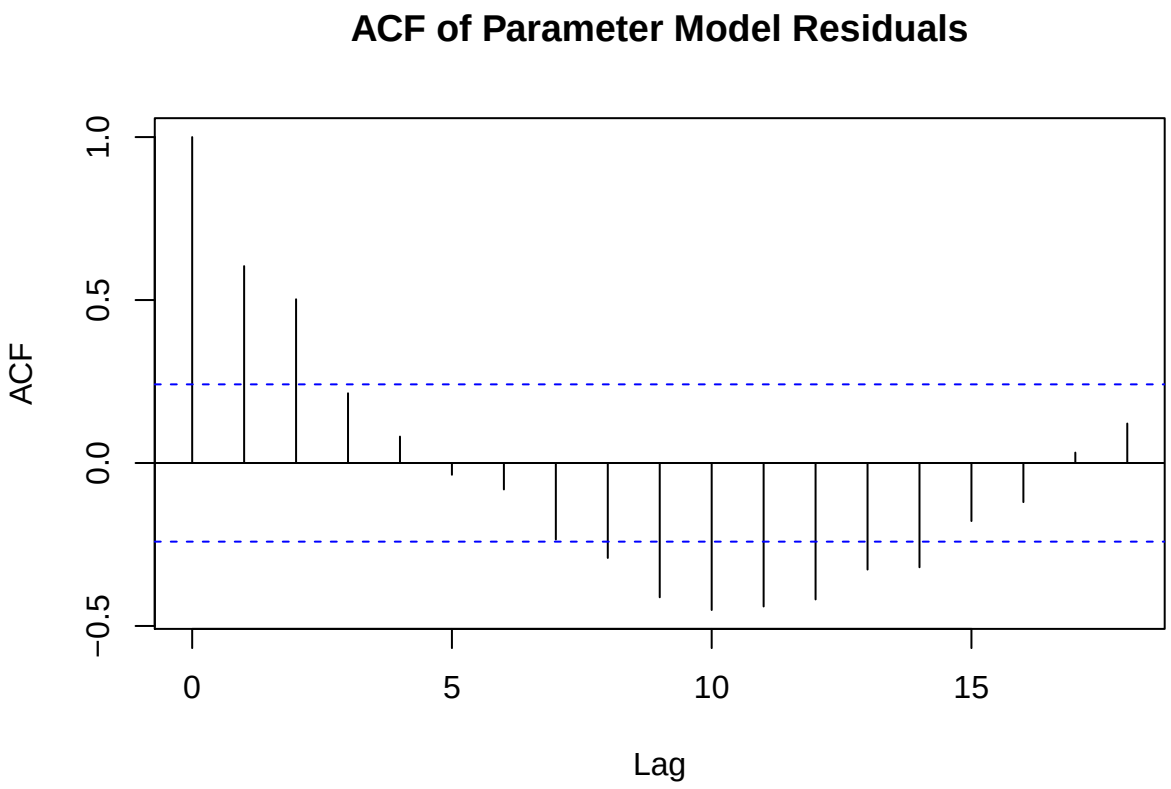

```
plot(param_model$residuals, type='l',
     main = 'Residuals of 5th Degree Parametric Model',
     ylab = "Residual", xlab = 'Day')
```



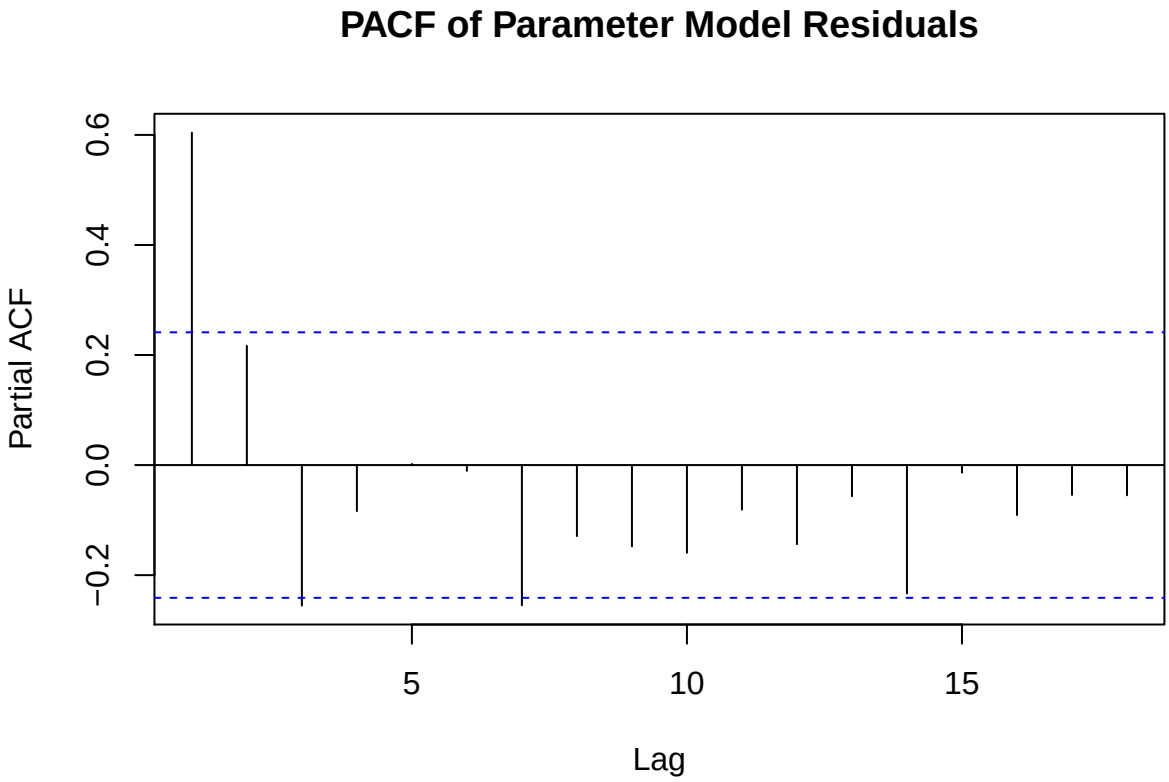
```r
# residuals do not look like white noise so need ARMA
param_model$fitted.values
```

```
##         1         2         3         4         5         6
##  0.2758427  0.4798951  0.6679275  0.8399802  0.9962284  1.1369740
##         7         8         9        10        11        12
##  1.2626385  1.3737547  1.4709598  1.5549872  1.6266595  1.6868802
##        13        14        15        16        17        18
##  1.7366264  1.7769414  1.8089266  1.8337343  1.8525599  1.8666342
##        19        20        21        22        23        24
##  1.8772160  1.8855843  1.8930307  1.9008518  1.9103417  1.9227842
##        25        26        27        28        29        30
##  1.9394453  1.9615654  1.9903520  2.0269718  2.0725431  2.1281284
##        31        32        33        34        35        36
##  2.1947265  2.2732652  2.3645934  2.4694733  2.5885735  2.7224608
##        37        38        39        40        41        42
##  2.8715924  3.0363090  3.2168266  3.4132290  3.6254604  3.8533173
##        43        44        45        46        47        48
##  4.0964417  4.3543124  4.6262383  4.9113504  5.2085942  5.5167219
##        49        50        51        52        53        54
##  5.8342853  6.1596275  6.4908760  6.8259344  7.1624751  7.4979320
##        55        56        57        58        59        60
##  7.8294923  8.1540890  8.4683937  8.7688085  9.0514586  9.3121848
##        61        62        63        64        65        66
##  9.5465355  9.7497595  9.9167980 10.0422774 10.1205012 10.1454429
```

```r
acf(param_model$residuals, main = 'ACF of Parameter Model Residuals')
```




```r
pacf(param_model$residuals, main = 'PACF of Parameter Model Residuals')
```

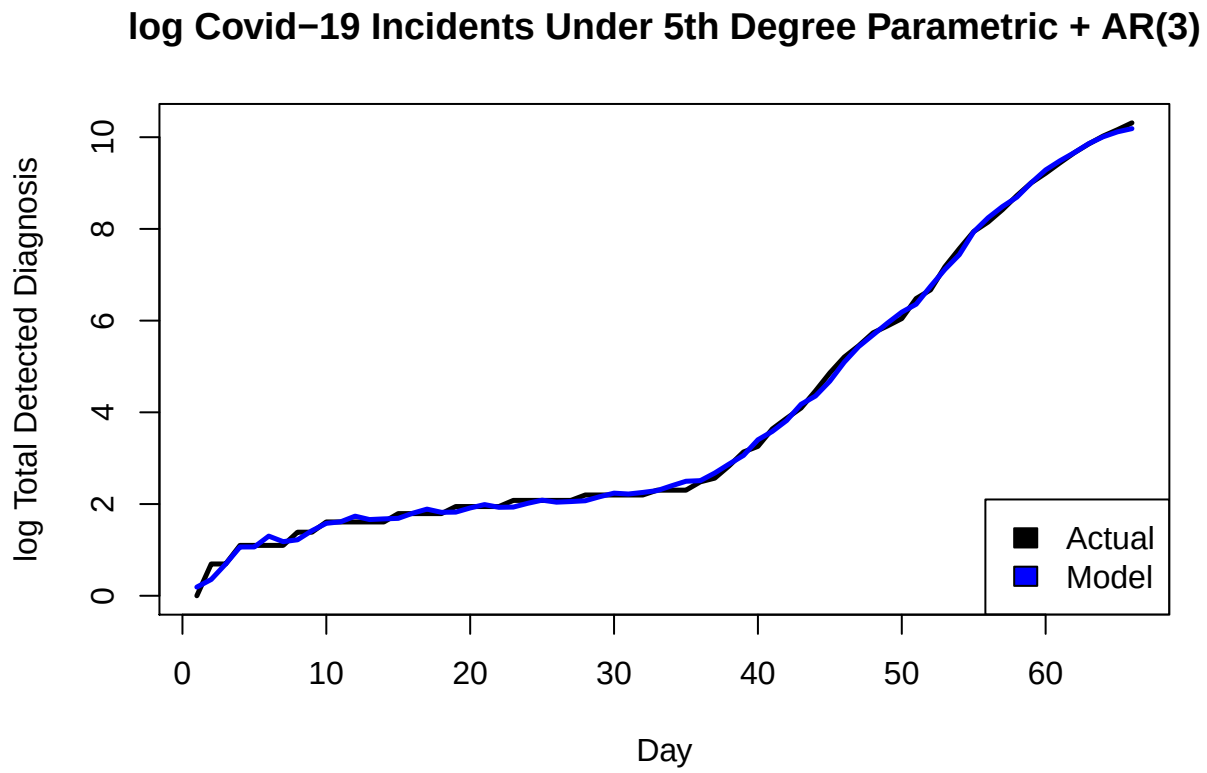



```
# Try ARIMA(3,0,0) becuase AR(3): ACF rollercoaster (exponential decrease)
#-> there is AR in here. PACF shows 3 main spikes, followed by majority of bars
#falling within blue confidence bands -> p=3. Also, no MA because
#PACF # never 0 at lags > a value of p

# get ar coefficients

ar3 <- arima(param_model$residuals, order = c(3,0,0),method='ML')
ar3
```

```
##
## Call:
## arima(x = param_model$residuals, order = c(3, 0, 0), method = "ML")
##
## Coefficients:
##          ar1     ar2      ar3  intercept
##       0.5644  0.4524  -0.4127    -0.0022
## s.e.  0.1295  0.1583   0.1334     0.0295
##
## sigma^2 estimated as 0.009144:  log likelihood = 60.68,  aic = -111.36
```

```
# find fitted values of model

ar_resid <- residuals(ar3)

ar_fitted <- log_corona - ar_resid

plot(log_corona, main = 'log Covid-19 Incidents Under 5th Degree Parametric + AR(3)',
     type = 'l', lwd = 2.5, xlab = 'Day', ylab = 'log Total Detected Diagnosis')
lines(ar_fitted, type = 'l', col = 'Blue', lwd = 2.5)
```

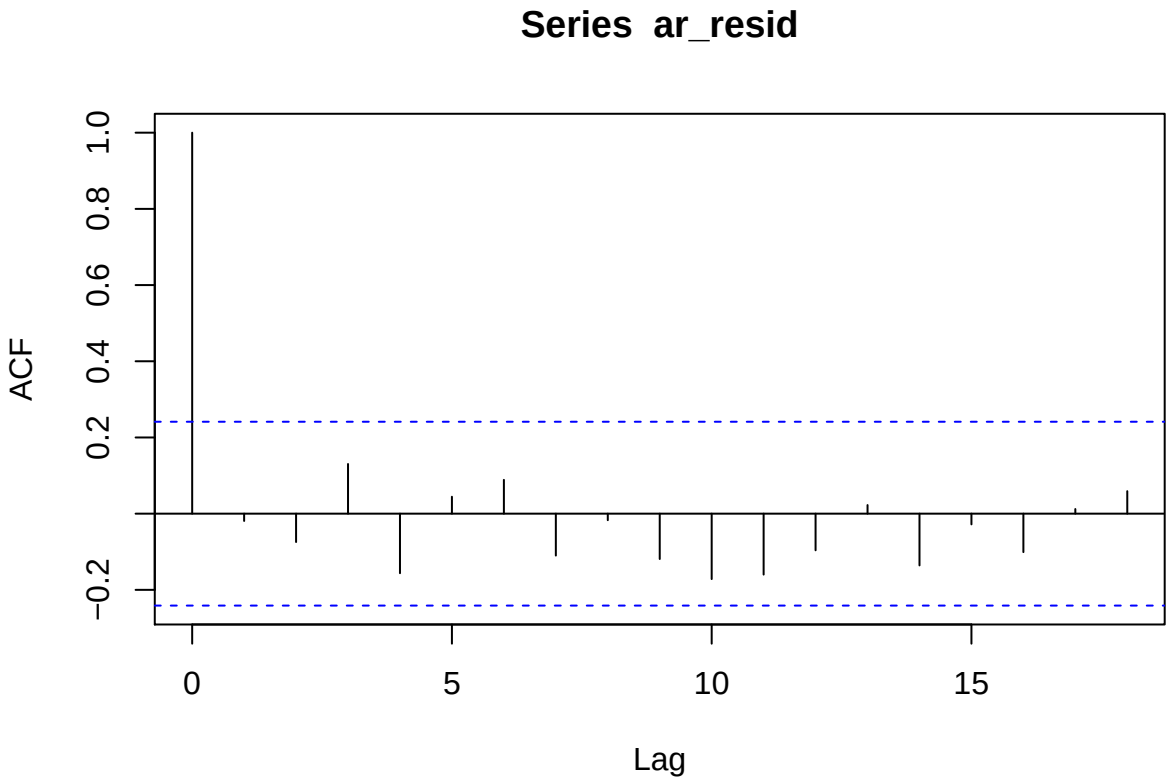
```
legend("bottomright", c("Actual", "Model"), fill = c("black", "Blue"))
```



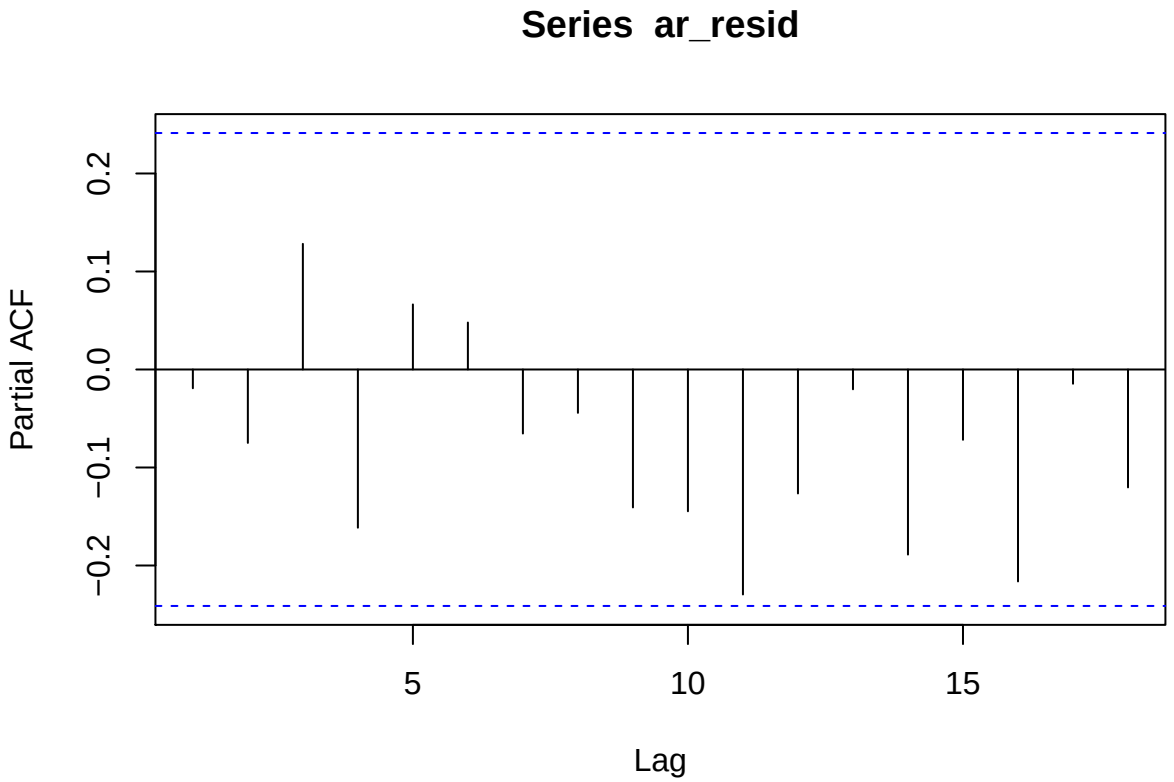

```
ar_fitted
```

```
## Time Series:
## Start = 1
## End = 66
## Frequency = 1
##  [1]  0.1887734  0.3550819  0.6915881  1.0636293  1.0647275  1.3004650
##  [7]  1.1796877  1.2206878  1.4187860  1.5796780  1.6130352  1.7358465
## [13]  1.6617726  1.6763455  1.6879242  1.7998765  1.8893428  1.8195305
## [19]  1.8238900  1.9146933  1.9881691  1.9287555  1.9339159  2.0159046
## [25]  2.0848817  2.0407814  2.0546809  2.0719222  2.1594100  2.2378683
## [31]  2.2189849  2.2535958  2.2934072  2.3981627  2.4968249  2.5102556
## [37]  2.6761292  2.8729094  3.0606308  3.4011101  3.5840371  3.8226657
## [43]  4.1751556  4.3553311  4.6800129  5.0936472  5.4324985  5.6886753
## [49]  5.9445674  6.1843619  6.3580878  6.7477160  7.1171066  7.4376080
## [55]  7.9370693  8.2428219  8.4876060  8.6924166  9.0064523  9.2857462
## [61]  9.4869109  9.6647641  9.8602773 10.0097017 10.1174740 10.1873369
```

```
acf(ar_resid)
```




```
pacf(ar_resid)
```

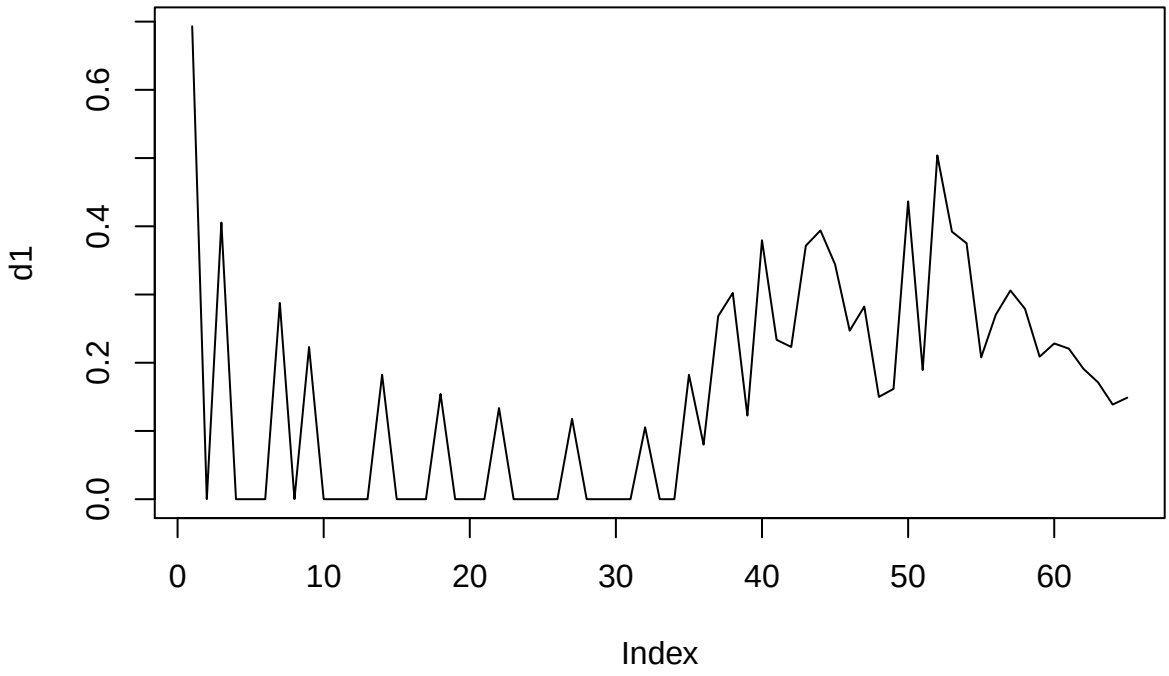



```
# acf looks like white noise! We are done

ar3$coef
```

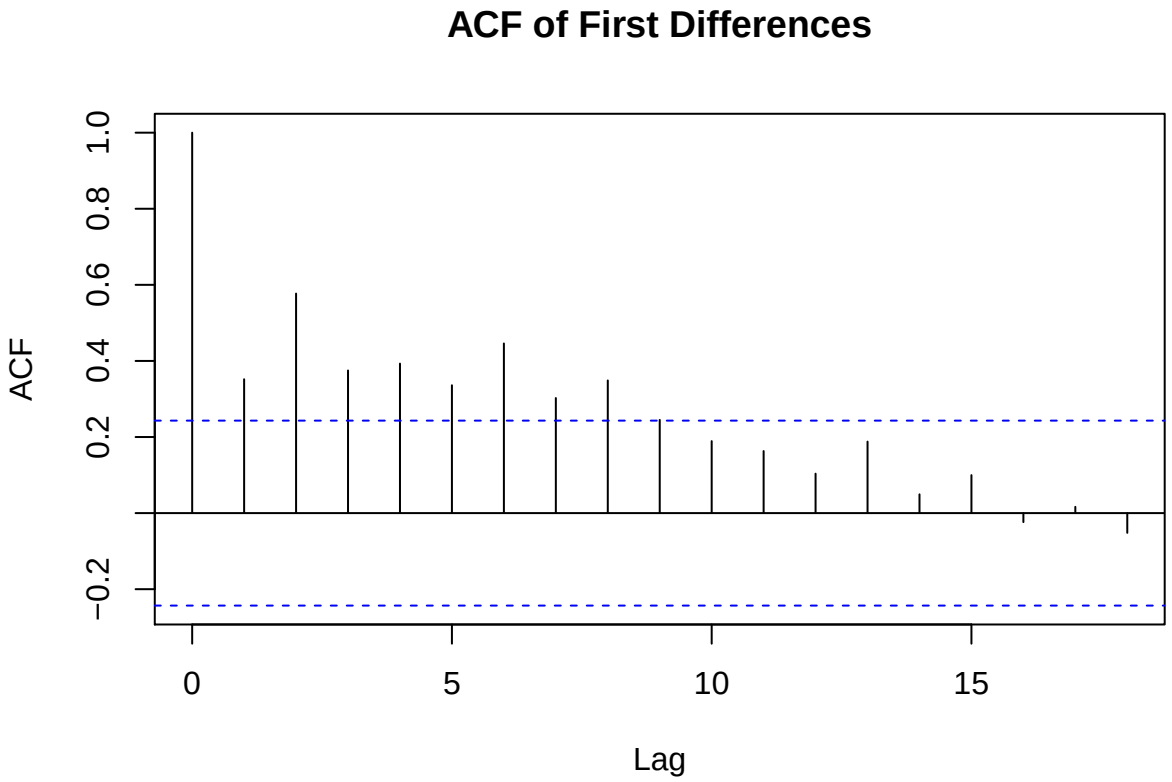
```
##          ar1          ar2          ar3    intercept
##  0.564418650  0.452358571 -0.412669092 -0.002231619
```

2.2: Andre's Model: ARIMA(0,2,2)

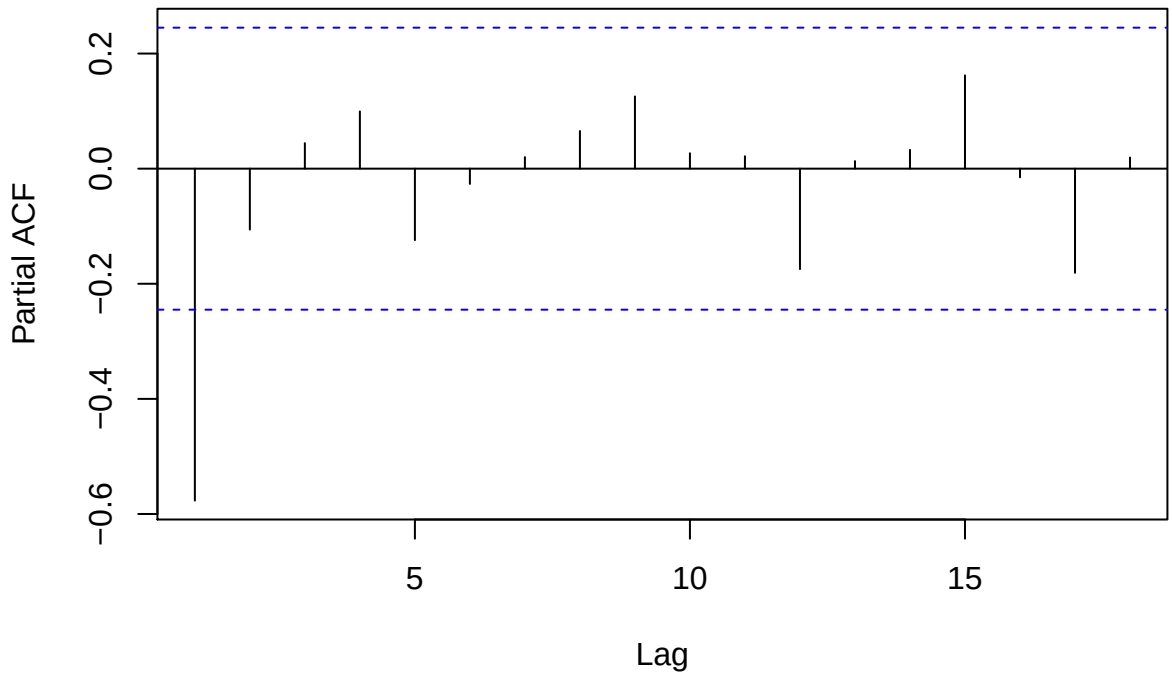
```
# andre

d1 <- diff(log_corona)
plot(d1, type = 'l')
```




```
acf(d1, main = "ACF of First Differences")
```



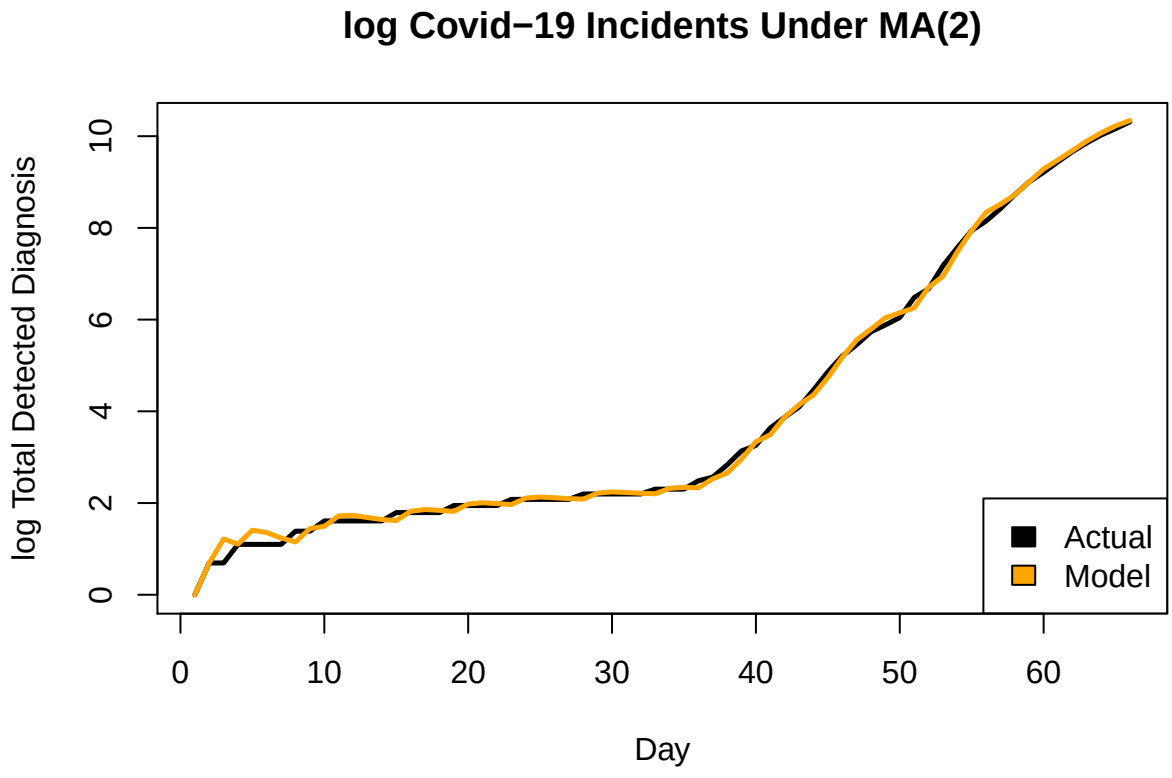

```r
d2 <- diff(d1)
#plot(d2, type = 'l', main = "Second Differences", xlab = "Day")
#acf(d2, main = "ACF of Second Differences")
pacf(d2, main = "PACF of Second Differences")
```



```r
#As we can see from the acf and pacf plot of our second differencing,
#the series is roughly stationary. We also note that since there
#are two statistically significant spikes at lags 1 and 2 followed by
#non-significant values for future lags, we propose testing out an MA(2) model.

ma2 <- arima(log_corona, order = c(0,2,2))
ma2
```

```
##
## Call:
## arima(x = log_corona, order = c(0, 2, 2))
##
## Coefficients:
##           ma1     ma2
##       -0.8353  0.2475
## s.e.   0.1404  0.1053
##
## sigma^2 estimated as 0.01623:  log likelihood = 40.7,  aic = -75.4
```

```r
ma2_residuals <- residuals(ma2)
ma2_fitted <- log_corona - ma2_residuals


plot(log_corona, type = 'l', main = 'log Covid-19 Incidents Under MA(2)',
     lwd = 2.5, xlab = 'Day', ylab = 'log Total Detected Diagnosis')
points(ma2_fitted, type = 'l', col = 'Orange', lwd = 2.5)
legend("bottomright", c("Actual", "Model"), fill = c("black", "orange"))
```



2.3: Group Model: ARIMA(1,2,0)

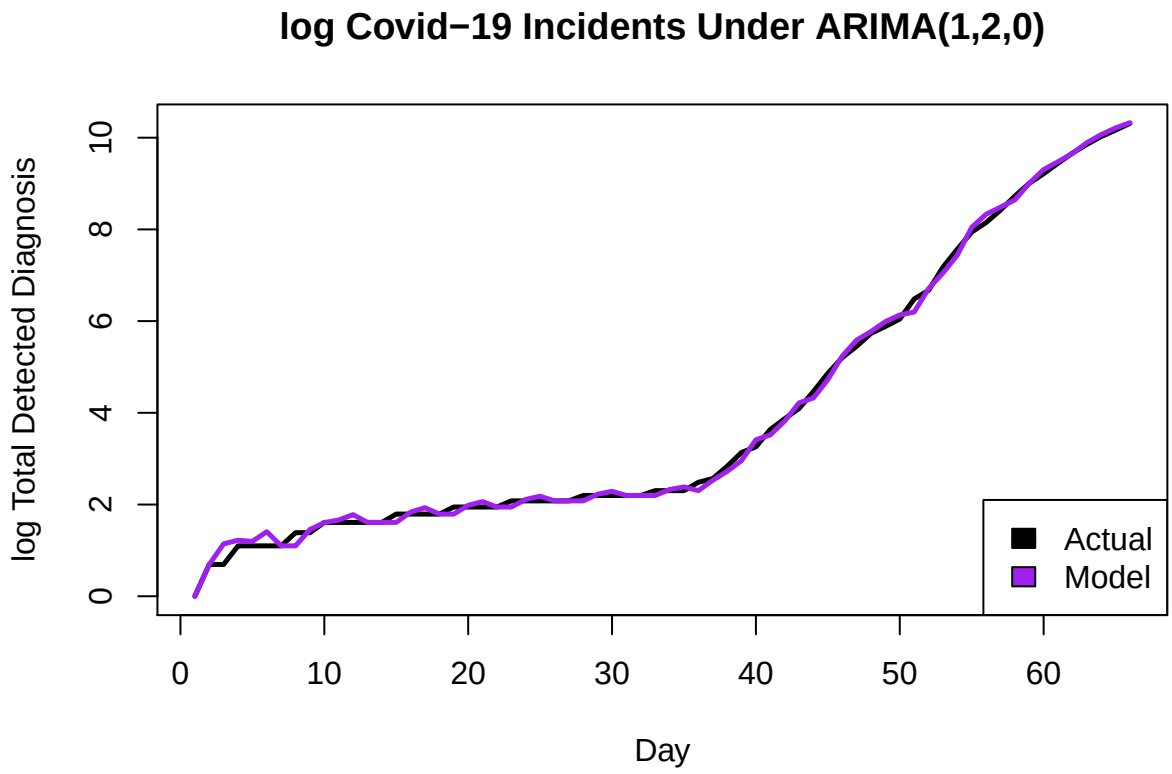
```
auto.arima(log_corona)
```

```
## Series: log_corona
## ARIMA(1,2,0)
##
## Coefficients:
##           ar1
##       -0.7617
## s.e.   0.1029
##
## sigma^2 estimated as 0.01585:  log likelihood=41.88
## AIC=-79.76   AICc=-79.56   BIC=-75.44
```

```
arima12 <- arima(log_corona, order = c(1,2,0))
arima12
```

```
##
## Call:
## arima(x = log_corona, order = c(1, 2, 0))
##
## Coefficients:
##           ar1
##       -0.7617
## s.e.   0.1029
##
## sigma^2 estimated as 0.01561:  log likelihood = 41.88,  aic = -79.76
```

```
arima_residuals <- residuals(arima12)
arima_fitted <- log_corona - arima_residuals


plot(log_corona, type = 'l', main = 'log Covid-19 Incidents Under ARIMA(1,2,0)',
     xlab = 'Day', ylab = 'log Total Detected Diagnosis', lwd = 2.5)
points(arima_fitted, type = 'l', col = 'purple', lwd = 2.5)
legend("bottomright", c("Actual", "Model"), fill = c("black", "purple"))
```




```
arima_fitted
```

```
## Time Series:
## Start = 1
## End = 66
## Frequency = 1
##  [1]  0.0000000  0.6915973  1.1422670  1.2211079  1.1952402  1.4074495
##  [7]  1.0986123  1.0986123  1.4548529  1.6054179  1.6626161  1.7794033
## [13]  1.6094379  1.6094379  1.6094379  1.8352092  1.9306313  1.7917595
## [19]  1.7917595  1.9826464  2.0633246  1.9459101  1.9459101  2.1112639
## [25]  2.1811506  2.0794415  2.0794415  2.0794415  2.2252939  2.2869383
## [31]  2.1972246  2.1972246  2.1972246  2.3276939  2.3828368  2.3025851
## [37]  2.5283564  2.7228965  2.9581118  3.4118649  3.5175576  3.8214082
## [43]  4.2159267  4.3254640  4.7244222  5.2367001  5.5860644  5.7720776
## [49]  5.9887786  6.1340911  6.1977751  6.7087698  7.0483835  7.4389272
## [55]  8.0441302  8.3300773  8.4852003  8.6430122  9.0049832  9.3046610
## [61]  9.4763237  9.6558261  9.8894062 10.0676166 10.2116749 10.3275131
```

3. Model Comparison and Selection

```
#RMSE:

sqrt(mean((ar_fitted - log_corona)^2))
```

```
## [1] 0.09562367
```

```
sqrt(mean((ma2_fitted - log_corona)^2))
```

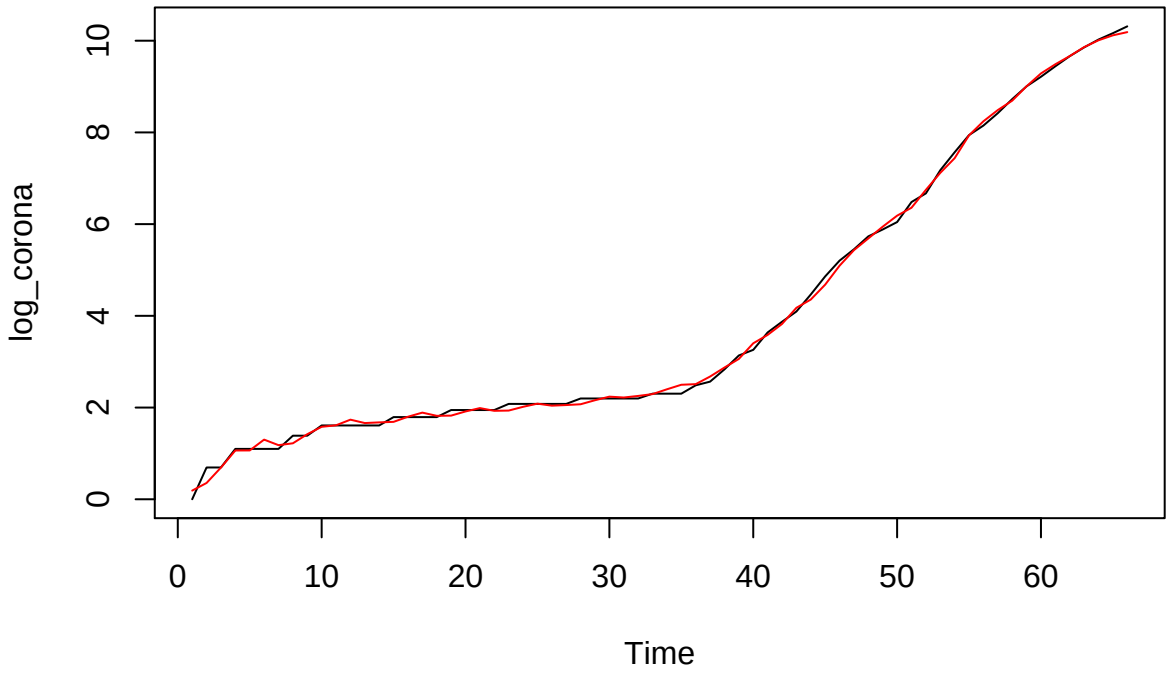
```
## [1] 0.1254401
```

```r
sqrt(mean((arima_fitted - log_corona)^2))
```

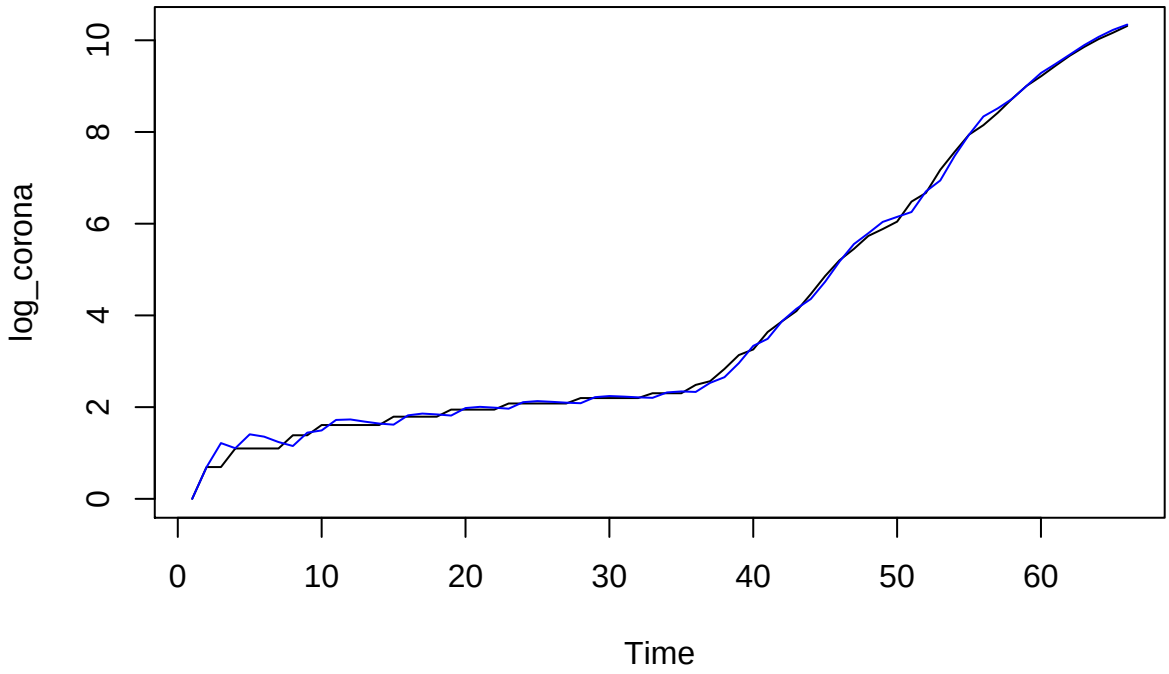
```
## [1] 0.1230138
```

```r
#can also use accuracy function
```

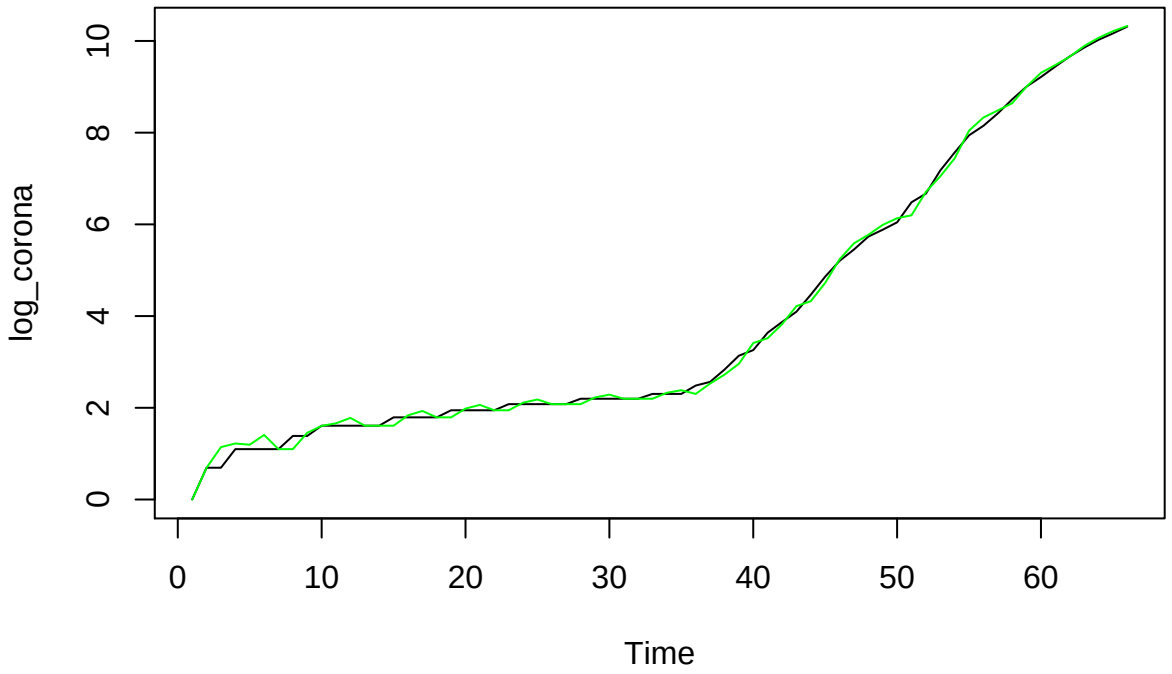
```r
ts.plot(log_corona)
lines(ar_fitted, col = "Red")
```




```r
ts.plot(log_corona)
lines(ma2_fitted, col = "Blue")
```



```r
ts.plot(log_corona)
lines(arima_fitted, col = 'Green')
```




```r
#When comparing models fitted by MLE to same data,
#the smaller the AIC or BIC, the better the fit.

#AIC
AIC(ar3)
```

```
## [1] -111.3601
```

```r
AIC(ma2)
```

```
## [1] -75.40414
```

```r
AIC(arima12)
```

```
## [1] -79.75749
```

```r
#BIC
BIC(ar3)
```

```
## [1] -100.4118
```

```r
BIC(ma2)
```

```
## [1] -68.92749
```

```r
BIC(arima12)
```

```
## [1] -75.43973
```

```
# Model 3 has both lowest AIC and BIC - We choose this as our final model.
```

4. Results

Final Model = (1 - 0.5644B - 0.4524Bˆ2 + 0.4127Bˆ3) X_t = W_t

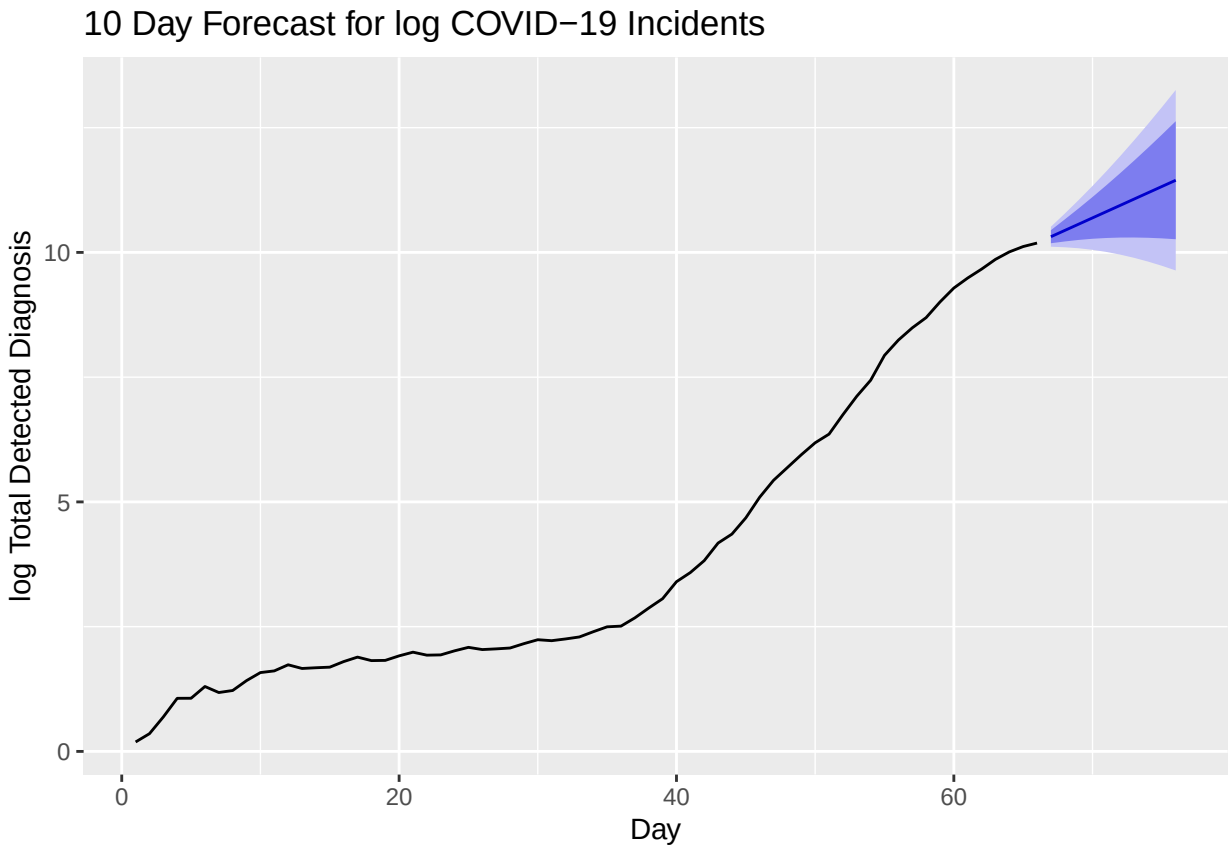
4.1 Estimation of Model Parameters

```
ar3
```

```
##
## Call:
## arima(x = param_model$residuals, order = c(3, 0, 0), method = "ML")
##
## Coefficients:
##          ar1     ar2      ar3  intercept
##       0.5644  0.4524  -0.4127    -0.0022
## s.e.  0.1295  0.1583   0.1334     0.0295
##
## sigma^2 estimated as 0.009144:  log likelihood = 60.68,  aic = -111.36
```

4.2 Prediction

```
autoplot((forecast(ar_fitted)), main = '10 Day Forecast for log COVID-19 Incidents',
         ylab = 'log Total Detected Diagnosis', xlab = 'Day', lwd = 2.5)
```




```
fcast <- as.data.frame(forecast(ar_fitted))
fcast
```

```
##    Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
## 67       10.31436 10.18340 10.44532 10.114077 10.51464
## 68       10.44011 10.21950 10.66073 10.102712 10.77752
## 69       10.56587 10.25007 10.88167 10.082893 11.04884
## 70       10.69162 10.27327 11.10997 10.051812 11.33143
## 71       10.81737 10.28896 11.34579 10.009240 11.62551
## 72       10.94313 10.29735 11.58890  9.955500 11.93076
## 73       11.06888 10.29873 11.83903  9.891042 12.24672
## 74       11.19464 10.29341 12.09586  9.816325 12.57295
## 75       11.32039 10.28165 12.35912  9.731780 12.90900
## 76       11.44614 10.26373 12.62855  9.637803 13.25448
```

```
fcast$`Point Forecast`
```

```
##  [1] 10.31436 10.44011 10.56587 10.69162 10.81737 10.94313 11.06888
##  [8] 11.19464 11.32039 11.44614
```

```
exp(fcast$`Point Forecast`)
```

```
##  [1] 30162.69 34204.57 38788.07 43985.77 49879.97 56564.02 64143.75
##  [8] 72739.17 82486.41 93539.81
```

```
corona$Count
```

```
##  [1]     1     2     2     3     3     3     3     4     4     5     5
## [12]     5     5     5     6     6     6     6     7     7     7     7
## [23]     8     8     8     8     8     9     9     9     9     9    10
```

```
## [34]    10    10    12    13    17    23    26    38    48    60    87
## [45]   129   182   233   309   359   422   653   789  1306  1933  2813
## [56]  3463  4538  6162  8145 10037 12611 15724 19035 22591 25952 30116
```

```
append(corona$Count, exp(fcast$`Point Forecast`) )
```

```
##  [1]     1.00     2.00     2.00     3.00     3.00     3.00     3.00
##  [8]     4.00     4.00     5.00     5.00     5.00     5.00     5.00
## [15]     6.00     6.00     6.00     6.00     7.00     7.00     7.00
## [22]     7.00     8.00     8.00     8.00     8.00     8.00     9.00
## [29]     9.00     9.00     9.00     9.00    10.00    10.00    10.00
## [36]    12.00    13.00    17.00    23.00    26.00    38.00    48.00
## [43]    60.00    87.00   129.00   182.00   233.00   309.00   359.00
## [50]   422.00   653.00   789.00  1306.00  1933.00  2813.00  3463.00
## [57]  4538.00  6162.00  8145.00 10037.00 12611.00 15724.00 19035.00
## [64] 22591.00 25952.00 30116.00 30162.69 34204.57 38788.07 43985.77
## [71] 49879.97 56564.02 64143.75 72739.17 82486.41 93539.81
```

```
plot(append(corona$Count, exp(fcast$`Point Forecast`)),
     main = "10 Day Forecast for COVID-19 Incidents", ylab = "Total Detected Incidents",
     xlab = "Day", type = 'l', lwd = 2.5, col = "Red")
```