MMA 867 Predictive Modelling
ASSIGNMENT 3

Jacqueline Mak
Student #: 20311028
Date of submission: September 16, 2021

Use auto.arima on the original time series to make forecasting

```
#Question 1 - throwing raw data to the model
model.raw.CAN <- auto.arima( CANNewcase, stepwise=FALSE, seasonal=TRUE)
model.raw.CAN
fit.raw.CAN <- arima(CANNewcase, order=c(0,1,5))
autoplot( forecast(fit.raw.CAN,16))
```

Output of the model:

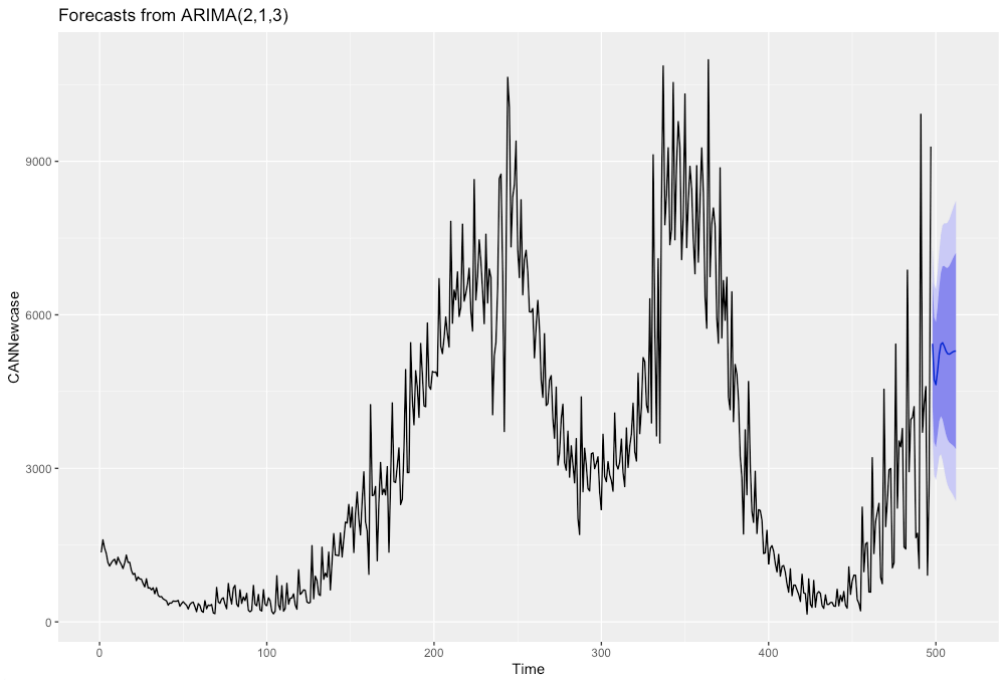
```
Series: CANNewcase
ARIMA(2,1,3)

Coefficients:
         ar1     ar2      ma1     ma2      ma3
      1.0313  -0.524  -1.9721  1.4739  -0.3312
s.e.  0.1119   0.076   0.1194  0.1974   0.1036

sigma^2 estimated as 908781:  log likelihood=-4104.99
AIC=8221.97   AICc=8222.15   BIC=8247.21
```

Number of daily cases forecasted in Canada from September 14 to September 28, 2021:

```
fc.CAN.raw<-forecast(fit.raw.CAN,16) #interested to see the fc mean
fc.CAN.raw$mean
```

```
 [1] 5436.004 4733.640 4634.215 4899.691 5225.584 5422.592 5455.016 5385.231 5296.269 5241.084 5230.782 5249.073 5273.335
[14] 5288.773 5291.983
```

Autoplot of the model:



Above is the auto.arima model's plot on the original time series without preprocessing the data. As you can you see, the prediction is a fairly straight blue line which clearly means it is not a good prediction. There are two properties that I noticed in the plot above. Firstly, the variance increases with the mean which means there is heteroskedasticity. As COVID cases go up, the variances tend to be higher. Secondly, the data has a cyclic pattern which means there is seasonality. We cannot directly apply the ARIMA model when either of the properties shows up in the dataset. Hence, it is important to stabilize the variance, remove seasonality through seasonal differencing and log the data, which will be further explained in details shortly.

Forecasting by first preprocessing the time series before applying auto.arima.

I begin my model first removing the first 100 days when the case counts are small.

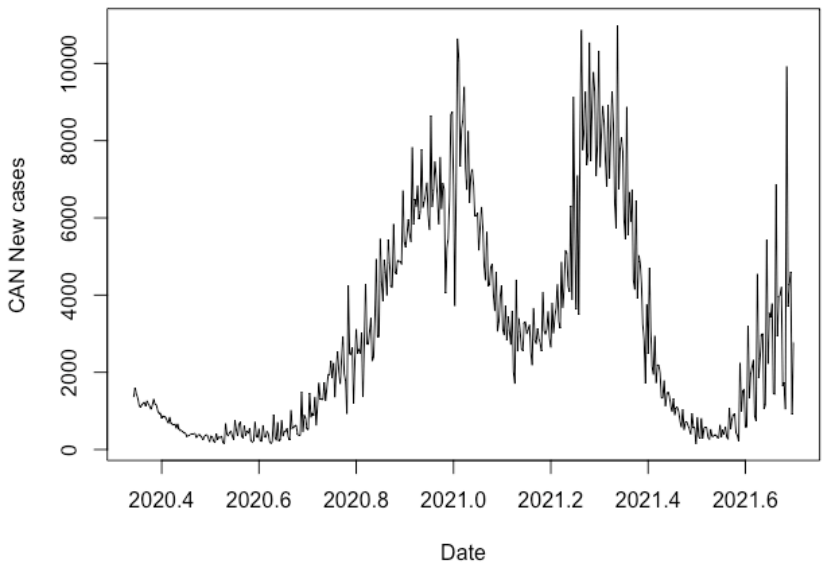
```
# Preprocessing
CANNewcase<- Canada.cases.data$new_cases
CANNewcase<-tail(CANNewcase,-100) # Remove the first 100 days when the case numbers are small
```

Since there is no time stamp on the dataset, I have to change it to a time series object.

```
CANNewcase <- ts(CANNewcase, frequency=365, start=c(2020, 126))
```

I can now plot the time series object.

```
plot.ts(CANNewcase, xlab="Date", ylab="CAN New cases")
```



Looking at the magnitude of the fluctuation in the plot above, the most recent data is more volatile when comparing to the dataset before. It could be due to the variance or the reporting data system in Canada works more accurately after April 2021. This time series also does not have a constant variance, hence there is heteroskedasticity. If we apply the arima model to this time series, it will not make great forecasting cases.

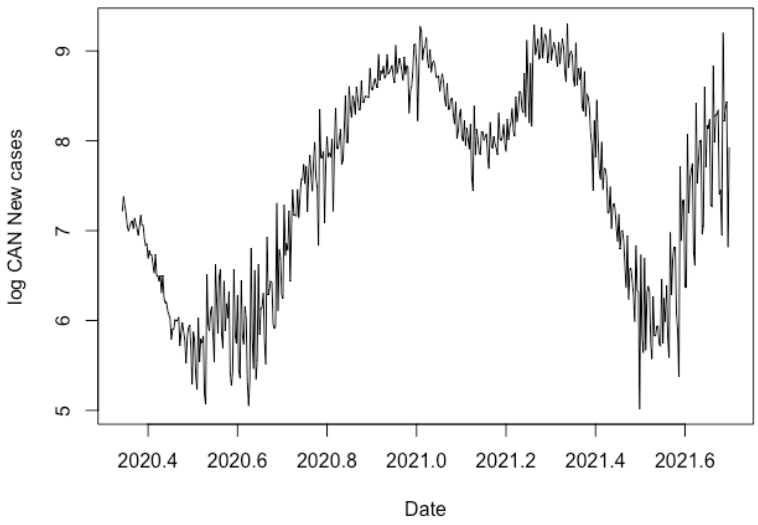
Stabilizing the Variance

The first approach is to stabilize the variance is by transforming the original data using Box-Cox Transform which is also known as Power transform. This will make the data more Gaussian in other others more normal.

```
CANNewcase.lambda <- BoxCox.lambda(CANNewcase)
CANNewcase.BoxCox<-BoxCox(CANNewcase, CANNewcase.lambda)
plot.ts(CANNewcase.BoxCox, xlab="Date", ylab="CAN New cases")
```

The second method is to use log transform.

```
logCANNewcase<-log(CANNewcase) # We use log transform for simplicity
plot.ts(logCANNewcase, xlab="Date", ylab="log CAN New cases")
```

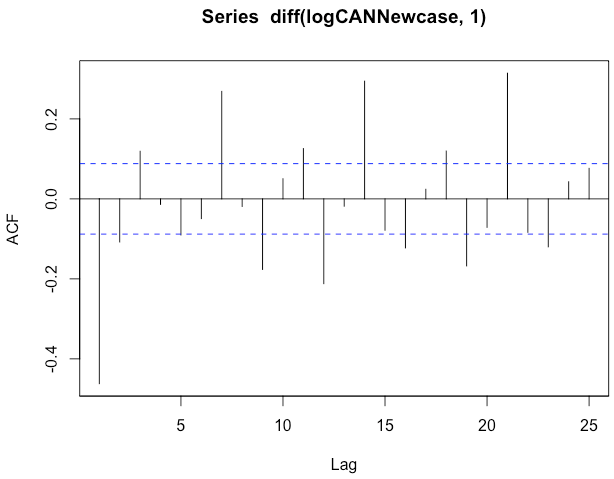


Referring to the plot above, although you can still see more variance starting April 2021, the variance overall is not changing with respect to the level of y and it seems to be fairly stable. Hence, log is an important technique for preprocessing the data if we have unstable variances, otherwise we would get wrong predictions.

Remove Seasonality through Seasonal Differencing

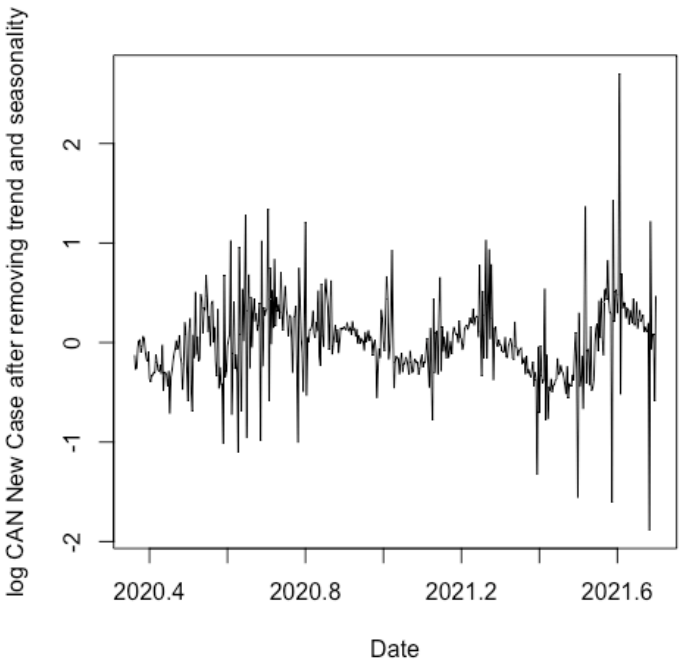
The next step is to remove seasonality through seasonal differencing. If we don't remove it, the predictions will not be accurate because ARIMA cannot handle seasonality. In order to check the period of cyclic pattern, we can use the autocorrelation function which will tell us how often the pattern is repeating.

```
Acf(diff(logCANNewcase,1),lag.max =25)
```

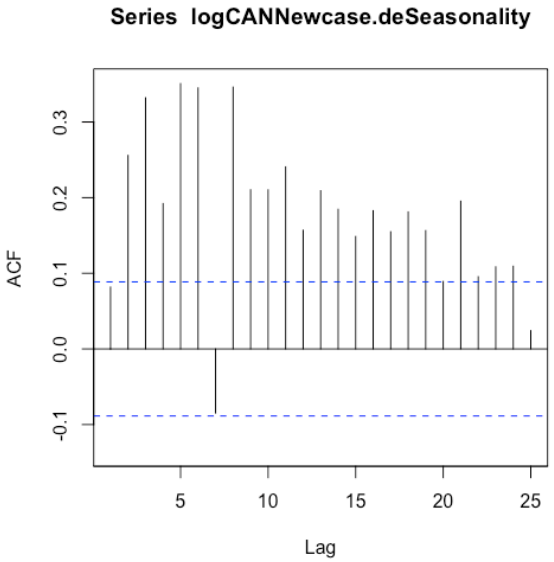


Referring to the plot above, we can see spikes at lag 7, 14, 21 which means a weekly pattern with a multiple of 7. When there is a big spike at a fixed time interval, there is seasonality in the time series data and we would need to remove it before doing a time series analysis.

```
logCANNewcase.deSeasonality <- diff(logCANNewcase,7)
plot.ts(logCANNewcase.deSeasonality, xlab="Date", ylab="log CAN New Case after removing trend and seasonality")
```



```
Acf(logCANNewcase.deSeasonality,lag.max =25)
```



Referring to the plot above, the big spikes are gone, which means seasonality has been removed. I can now use the dataset as I have logged and de-seasoned it.

Automatic ARIMA Modeling

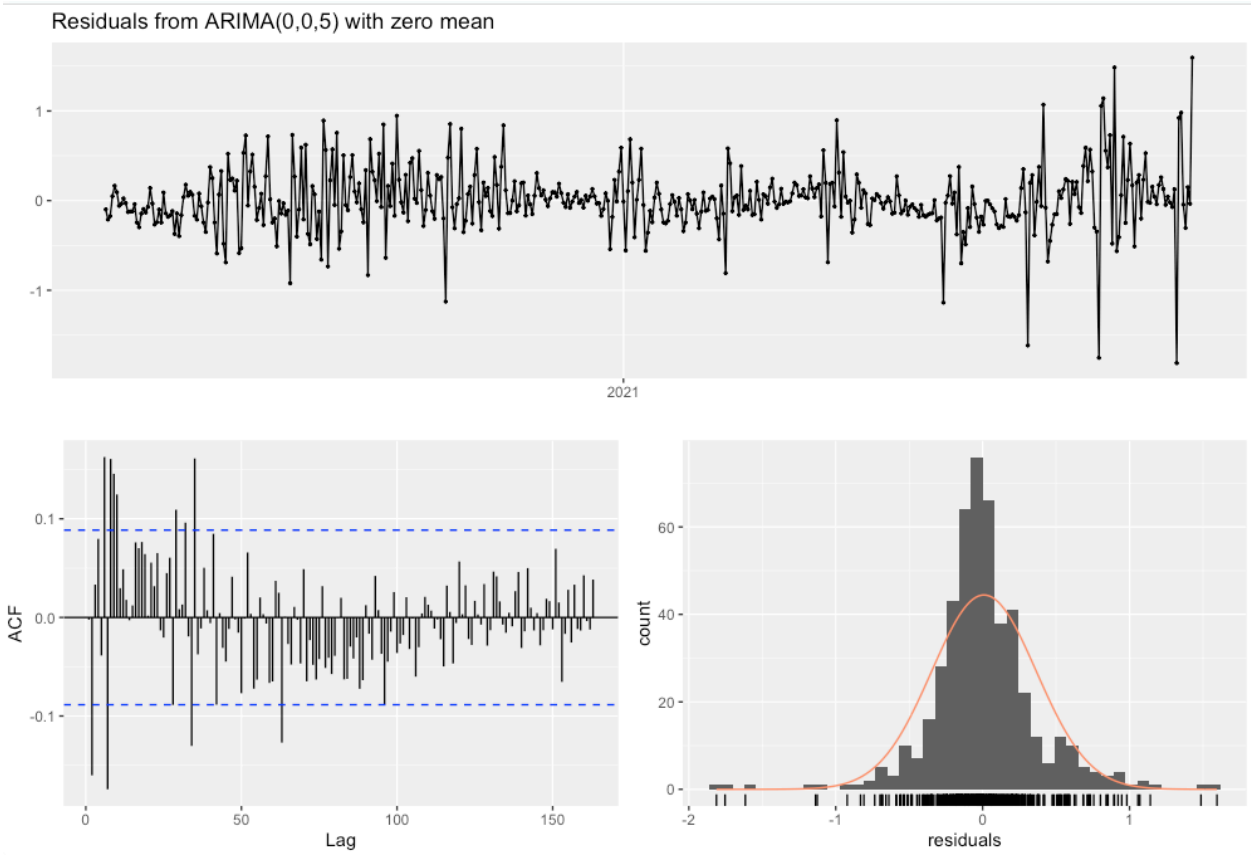
As a starting point, I begin to use an automated algorithm to find a good model.

```
model.auto.CAN <- auto.arima( logCANNewcase.deSeasonality, stepwise=FALSE, seasonal= FALSE)
model.auto.CAN
```

Output of the model:

```
Series: logCANNewcase.deSeasonality
ARIMA(0,0,5) with zero mean

Coefficients:
          ma1     ma2     ma3     ma4     ma5
      -0.1202  0.5117  0.1247  0.1659  0.5668
s.e.   0.0334  0.0385  0.0417  0.0348  0.0351

sigma^2 estimated as 0.1285:  log likelihood=-191.54
AIC=395.07   AICc=395.25   BIC=420.24
```

The automated algorithm suggests a ARIMA(0,0,5) model. In order to determine if this a good model, we would need to rely on the AIC, which is a measure of how good your model is, the lower the better. AIC depends heavily on log transformation. If we don't use the log transformation, the AIC will be very different. When we compared the AIC/AICc/BIC from this model to the previous model with raw data, the AIC has improved from 8221 to 395, AICc from 8222 to 395 and lastly BIC from 8247 to 420. We can also check the quality of fit by checking the residuals.



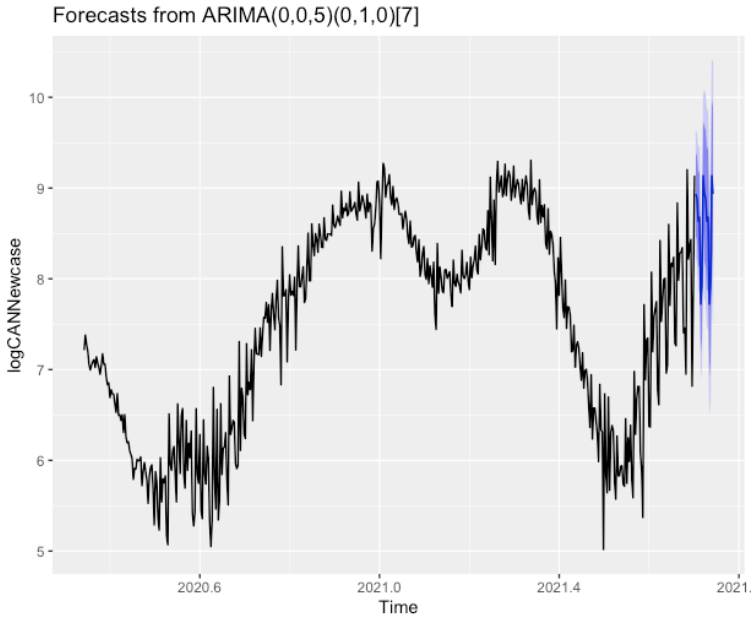
The residuals should have the following characteristics:

- No significant autocorrection is displayed
- Follow a normal distribution
- Have stable variance over time

The ACF is most important plot to examine as we should not see many sticks out, which means the residuals are not significantly correlated with each other. We would need to worry if we see many sticks sticking out. The residuals above look good, so we can now fit the model using (0,0,5) that auto.arima suggested and make forecasting for the cases.

Code:

```
fit.yourself.CAN <- Arima(logCANNewcase, order=c(0,0,5), seasonal=list(order=c(0,1,0),period=7))
fit.yourself.CAN #the coefficients are same as before
autoplot( forecast(fit.yourself.CAN,15) )
```

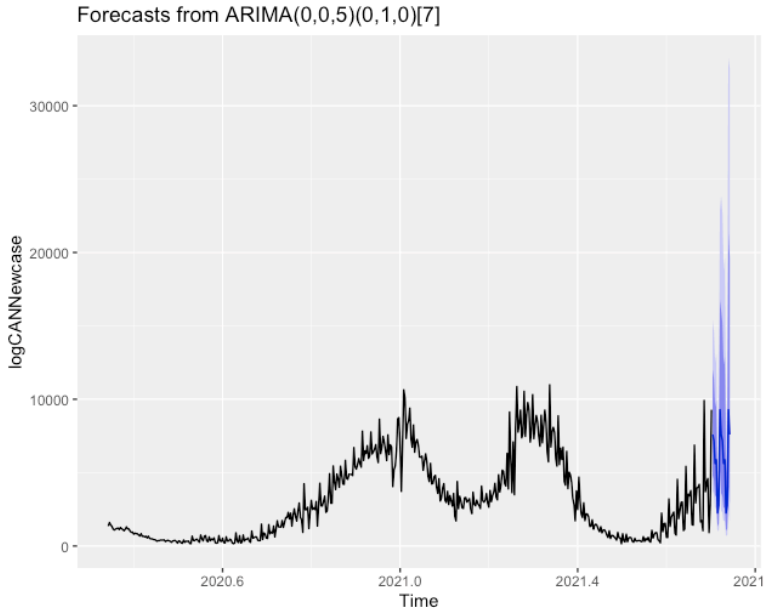


Referring to the plot above, we are forecasting the number of cases from September 14 to September 28, 2021 (15 days in advance). In comparison to the raw data model (without preprocessing), this autoplot can predict those detailed fluctuations versus a straight blue line in the raw data model. As you can see, it predicted lower number on weekends and higher in the weekdays, but overall, the number of cases suggested by the model will be stable. The shaded parts of the predictions are the marginal errors. As we move further to the future, the marginal error is getting bigger, however this is expected.

We can now plot the forecasting in the original scale to see the forecasted means.

```
 [1] 7597.378 7194.550 5625.651 5873.035 2254.661 2774.000 9288.000 7597.378 7194.550 5625.651 5873.035
[12] 2254.661 2774.000 9288.000 7597.378
```

Below is the actual prediction curve after we take the exponential value of the numbers.



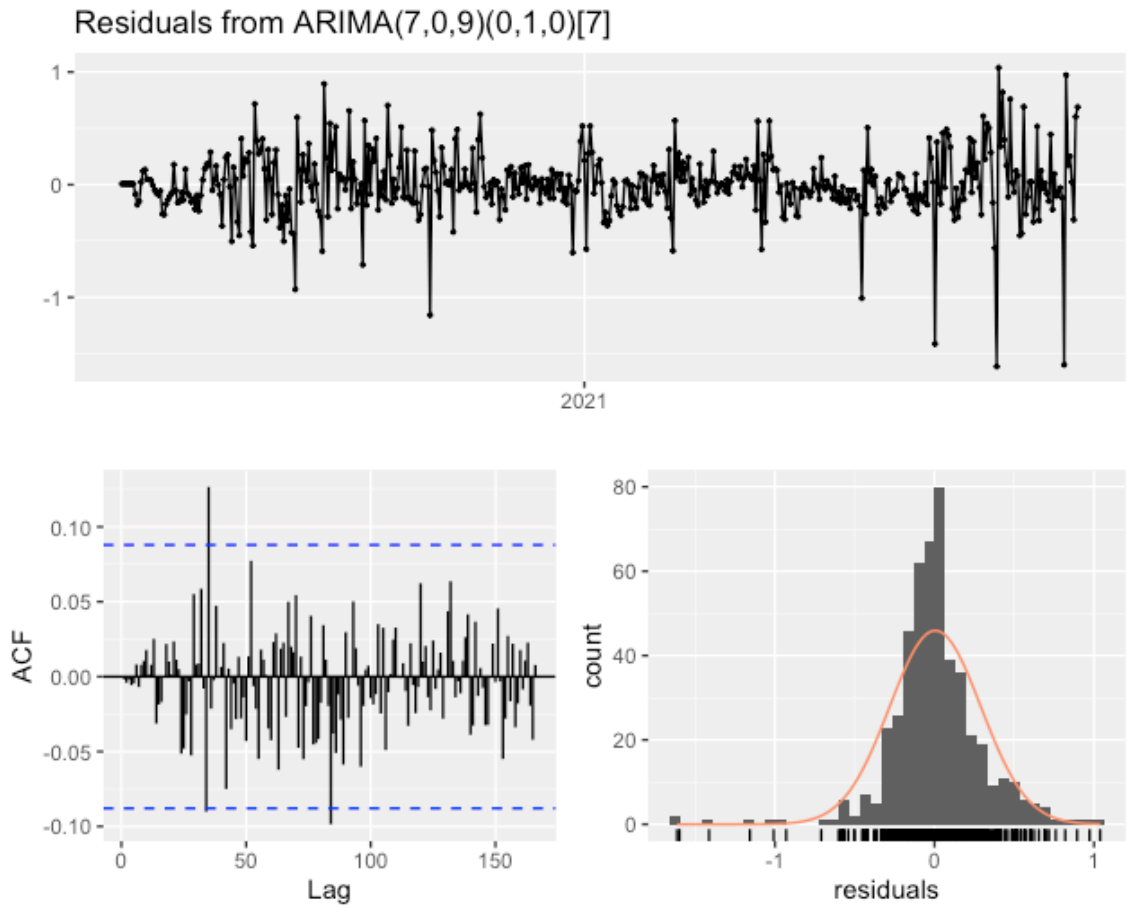
Improve your model by exploring alternative orders

As mentioned previously, the auto.arima function does not always find the model with the lowest AIC, AICc or BIC. In order to improve the automatically selected model, I explored other models manually to see if one could produce a lower AIC/AICc/BIC than the existing model.

```
Series: logCANNewcase
ARIMA(7,0,9)(0,1,0)[7]

Coefficients:
         ar1     ar2     ar3      ar4     ar5      ar6      ar7      ma1      ma2      ma3     ma4      ma5
      0.4840  0.5422  0.1776  -0.0772  0.1078  -0.0311  -0.2352  -0.4476  -0.4312  -0.0085  0.0788  -0.0512
s.e.  0.3751  0.2969  0.0985   0.0847  0.0937   0.0789   0.0769   0.3793   0.2855   0.0595  0.0496   0.0629
         ma6      ma7     ma8     ma9
      0.1378  -0.7696  0.3878  0.2608
s.e.  0.0419   0.0701  0.3254  0.2495

sigma^2 estimated as 0.08588:  log likelihood=-92.35
AIC=218.7   AICc=220   BIC=290
```

After exploring other models manually, my final model is ARIMA(7,0,9). I selected this as my final model because it has lowest AIC, AICc and BIC (refer to the output of the model above) when compared to other models that I tested manually.

I also checked the quality of fit by checking the residuals.

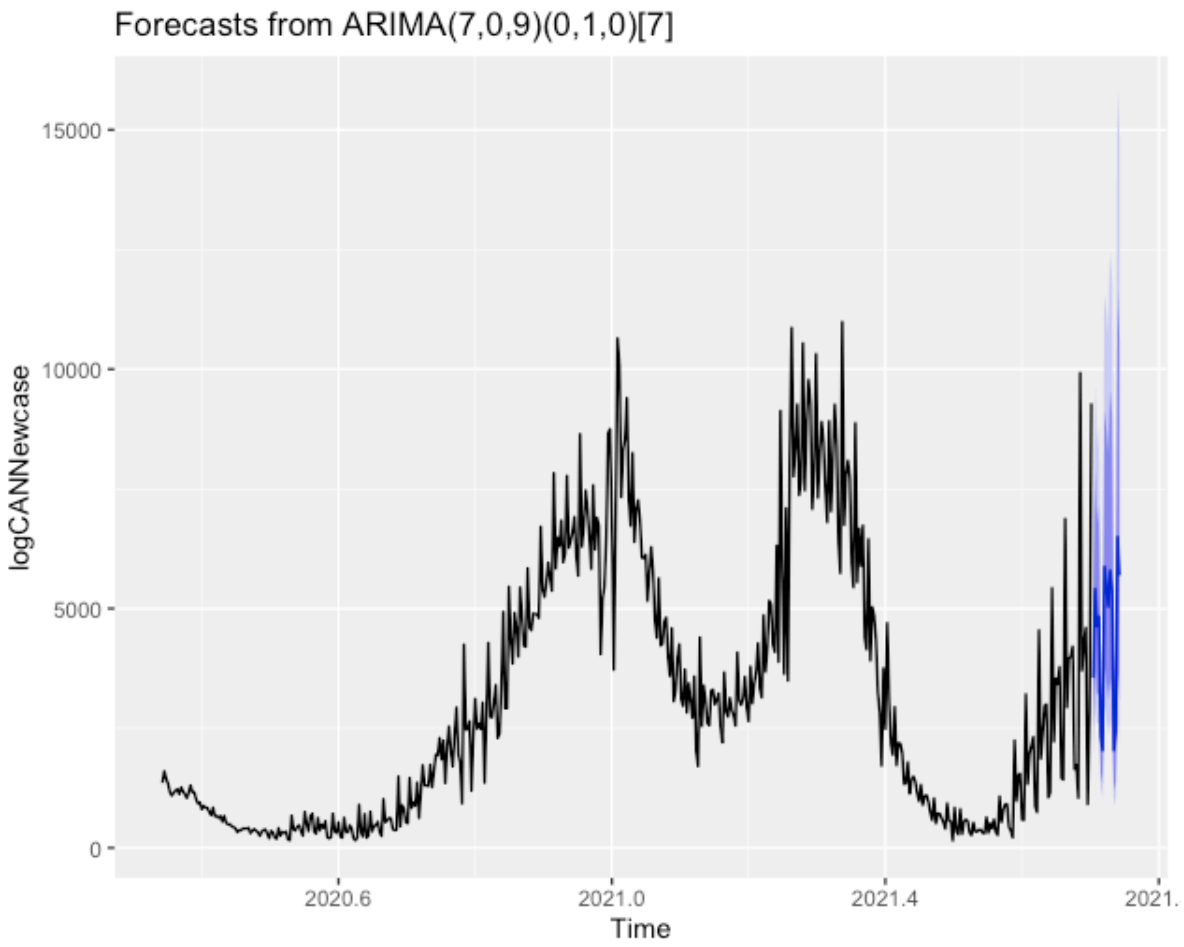


The residuals above look good as it has no significant autocorrection and it follows a normal distribution. Regarding the ACF plot, we don't want to see more than 5% of the sticks sticking out. Since the quality of the fit looks good, we can now fit the model using the order arima(7,0,9) and make forecasting for the cases.

We can now plot the forecasting in the original scale to see the forecasted means.

```
 [1] 3553.763 5409.149 4630.263 4834.497 2326.691 2036.515 5873.771 5495.022 5036.506 5792.782 5387.909
[12] 2038.396 2423.471 6497.831 5691.765
```

Below is the actual prediction curve after we take the exponential value of the numbers.



Referring to the plot above, we are forecasting the number of cases from September 14 to September 28, 2021 (15 days in advance) using ARIMA(7,0,9). In comparison to the raw data model (without preprocessing), this autoplot shows that it makes better predictions as you can see the detailed fluctuations versus a straight blue line in the raw data model. Similarly to the previous model using ARIMA(0,0,5), as we move further to the future, the marginal error is getting bigger, however this is expected.

ARIMA with Covariates - Dynamic Regression

On top of the time series model, I added the stringency index as a predictor (X) and incorporated it into the model through dynamic regression. There are 3 lagged predictors that have been tested for the stringency index.

```
stringency<- cbind(Canada.cases.data1[,46],
                   c(NA,Canada.cases.data1[1:496,46]),
                   c(NA,NA,Canada.cases.data1[1:495,46]),
                   c(NA,NA,NA,Canada.cases.data1[1:494,46]))
colnames(stringency) <- paste("AdLag",0:3,sep="")
stringency
```

I chose the optimal lag length for stringency index based on the AIC and BIC.

```
fit1 <- auto.arima(logCANNewcase[4:497], xreg=stringency[4:497,1], d=0) #
fit2 <- auto.arima(logCANNewcase[4:497], xreg=stringency[4:497,1:2], d=0)
fit3 <- auto.arima(logCANNewcase[4:497], xreg=stringency[4:497,1:3], d=0)
fit4 <- auto.arima(logCANNewcase[4:497], xreg=stringency[4:497,1:4], d=0)
```

```
# Compute Akaike Information Criteria
AIC(fit1) #lowest AIC (443.8593), without lag
AIC(fit2)
AIC(fit3)
AIC(fit4)
```

```
BIC(fit1) #strong evidence that first model is good
BIC(fit2)
BIC(fit3)
BIC(fit4)
```

```
> AIC(fit1) #lowest AIC (443), without lag
[1] 443.8593
> AIC(fit2)
[1] 445.8233
> AIC(fit3)
[1] 446.988
> AIC(fit4)
[1] 446.7444
```

```
> BIC(fit1) #strong evidence that first model is good
[1] 469.0379
> BIC(fit2)
[1] 475.1984
> BIC(fit3)
[1] 480.5596
> BIC(fit4)
[1] 484.5124
```

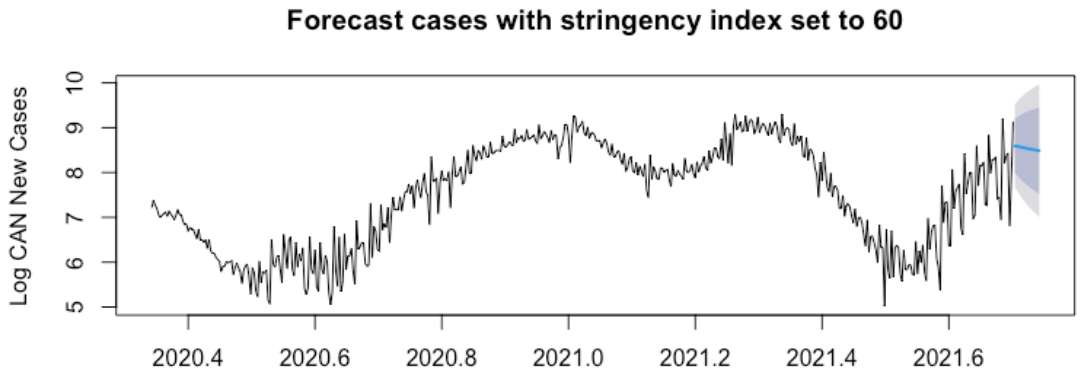
As per the lowest AIC and BIC above, there is strong evidence that the first model (fit1) is the best fit (using no lagged predictors).

```
fit <- auto.arima(logCANNewcase, xreg=stringency[,1], d=0)
fit
```

```
Series: logCANNewcase
Regression with ARIMA(3,0,0) errors

Coefficients:
         ar1     ar2     ar3  intercept     xreg
      0.3402  0.2251  0.4084    10.3401  -0.0402
s.e.  0.0410  0.0426  0.0409     1.2838   0.0166

sigma^2 estimated as 0.1402:  log likelihood=-216.03
AIC=444.06   AICc=444.23   BIC=469.28
```

I then forecasted the number of cases with the stringency index = 60 for 15 days. As you can see from the graph below, the number of new cases is gradually going down as restriction measures become stricter.

```
fc60 <- forecast(fit, xreg=rep(60,15), h=15)
plot(fc60, main="Forecast cases with stringency index set to 60", ylab="Log CAN New Cases")
```



Another scenario is I forecasted the number of cases with the stringency index = 5 for 15 days. As you can see, the number of new cases has gone up drastically when I relaxed the striction.

```
fc5 <- forecast(fit, xreg=rep(5,15), h=15)
plot(fc5, main="Forecast cases with stringency index set to 5", ylab="Log CAN New Cases")
```