Workbook 2 - Time Series

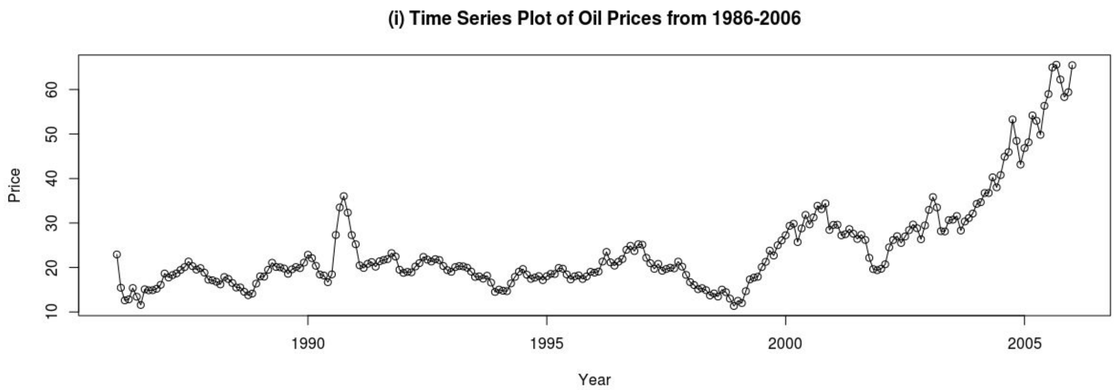
MAM5220 - Statistical Techniques for Computational Biology

Note to see code/images: https://github.com/sap218/R/tree/master/mam5220/w2

Exercise 1

Autocorrelation and Stationarity - application to oil prices

a)



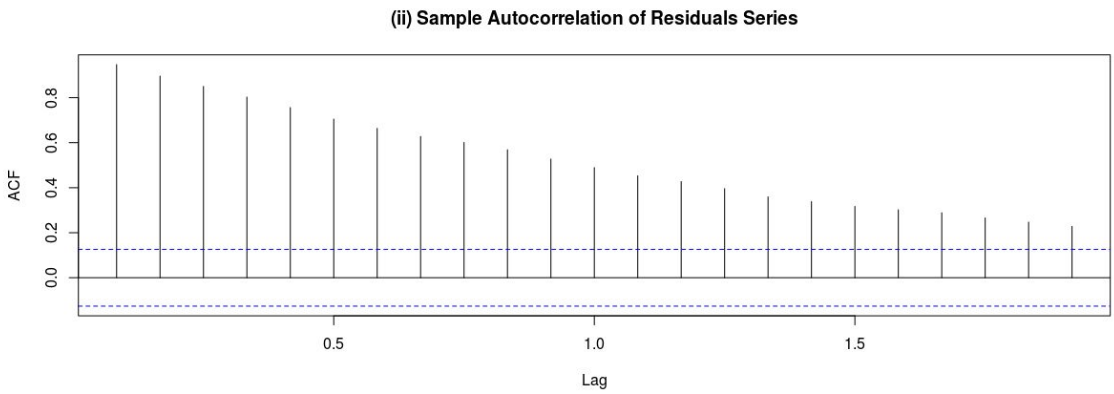


Figure 1: As we can see from the above graphs, the oil prices time series data isn't stationary: values from plot (i) don't depend on time and doesn't have an obvious strong increasing trend, it is quite skewed then the growth is somewhat exponential. If the time series was from 1986-1995, it could be considered stationary with an outlier during 1991. The Autocorrelation Function (acf) plot (ii) shows large positive correlations for several lags which quickly decay towards zero, clearly showing that the time series of Oil Prices isn't stationary.

b)

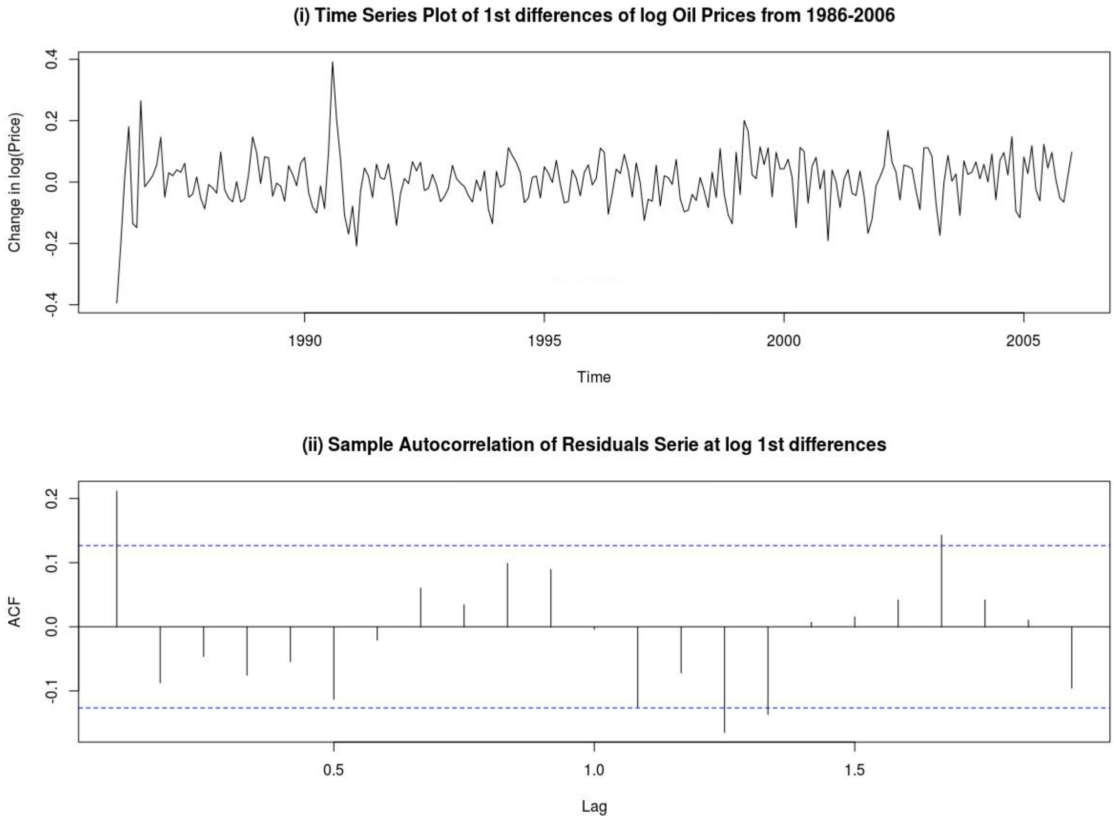


Figure 2: When we constructed the series of 1st differences of log prices, we can see that both the time series and ACF plots become stationary since differencing is often a convenient way to reduce non-stationary series to stationarity. Log reduces effect of inflations since plotting a time series does not consider external factors; such as: currency inflations.

Exercise 2

ARMA and ARIMA Models

2.1

a)

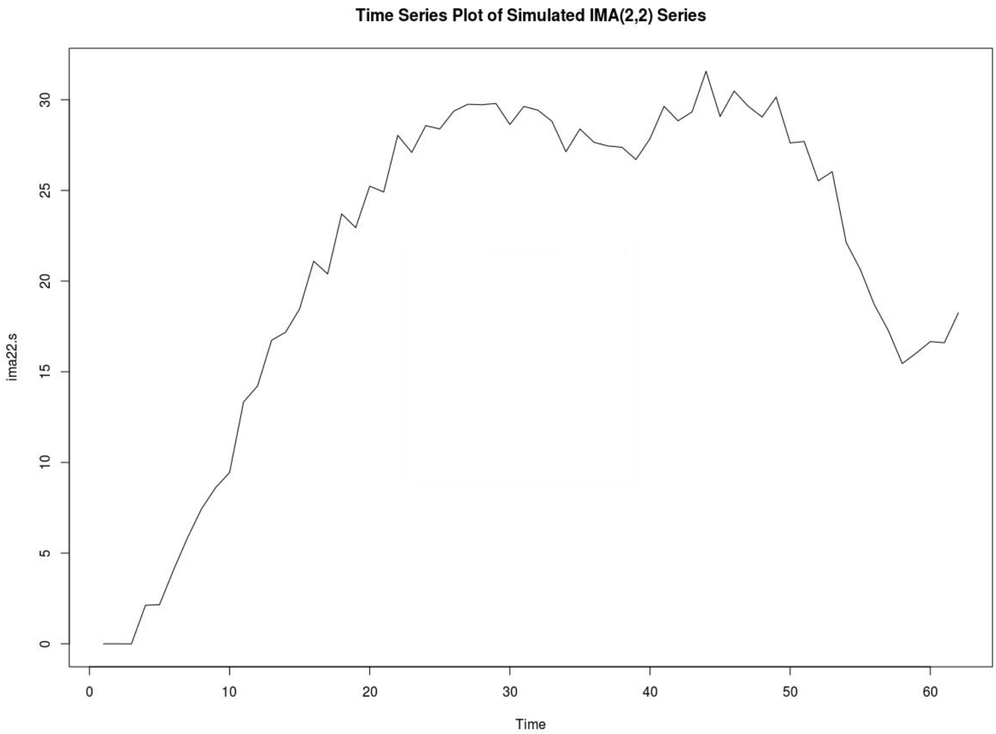


Figure 3: This is a Time Series Plot that states how values are not dependent on time, the count increases steadily, becomes somewhat stationary between timestamp 25-50, then decreases again until we see an increase again at timestamp 60.

b)

(i)

$$\nabla^2 Y_t = e_t - \theta_1 e_{t-1} - \theta_2 e_{t-2}$$
$$\nabla^2 Y_t = e_t - e_{t-1} + 0.6 e_{t-2}$$

(ii)

$$Y_t = 2Y_{t-1} + Y_{t-2} - \theta_1 e_{t-1} - \theta_2 e_{t-2}$$
$$Y_t = 2Y_{t-1} + Y_{t-2} - e_{t-1} + 0.6 e_{t-2}$$

c)

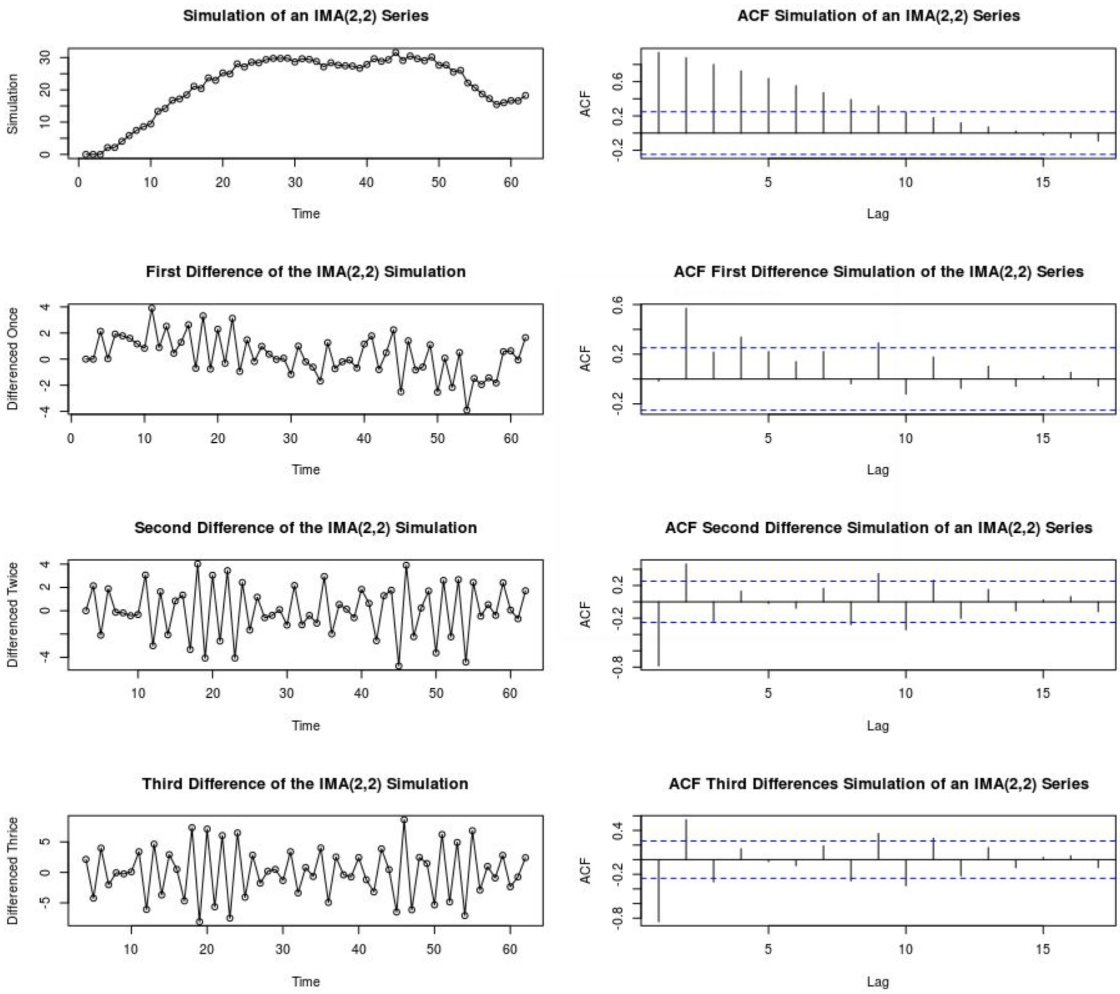


Figure 4: As you can see above plots, we can confirm that through time plots and the ACFs that 1st differenced “ima22.s” is still non-stationary due to the ACF slightly decreasing. On the other hand, the 2nd differenced series is stationary. Differencing a 3rd time is very similar to 2nd backing up our claim that the series becomes stationary at 2nd difference.

2.2

a)

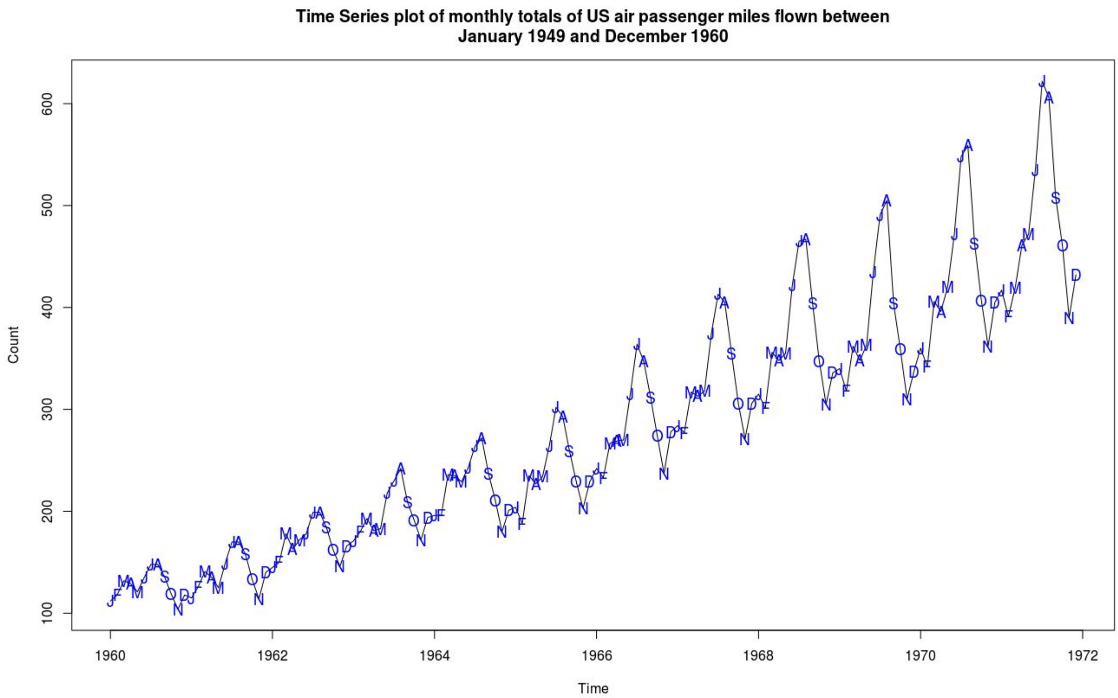


Figure 5: This time series plot seems to depend on time since there is a trend: the plot does not seem stationary however if logged, it would most likely become stationary. We can see a seasonal trend as I have plotted the month labels on the plots which conclude that for each year, July and August are always a spike (Summer) and there's always a very small spike during December/January for Winter.

b)

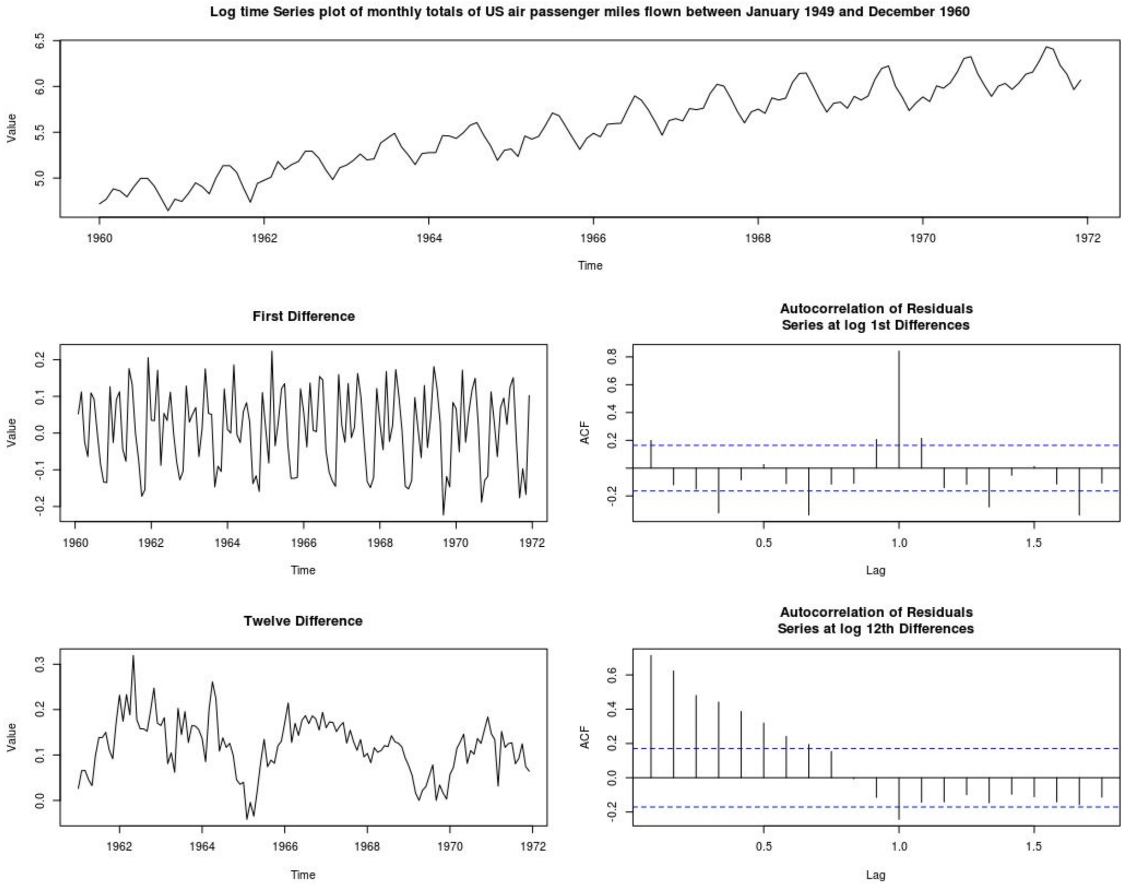


Figure 6: As you can see from the above set of plots, I calculated the logs of the "Airpass" data and displayed the updated time series plot. Also plotted includes the 1st difference and 12th difference with their ACFs. The first difference makes the data stationary showing the seasonal change excluding the count increase over years: it is only displaying the count as a constant rate. The 12th difference makes the plot appear yearly: 12 months in a year for the monthly data - this plot differs, especially the ACF plot: it becomes less stationary showing the positive correlation lags decreasing towards zero.

c)

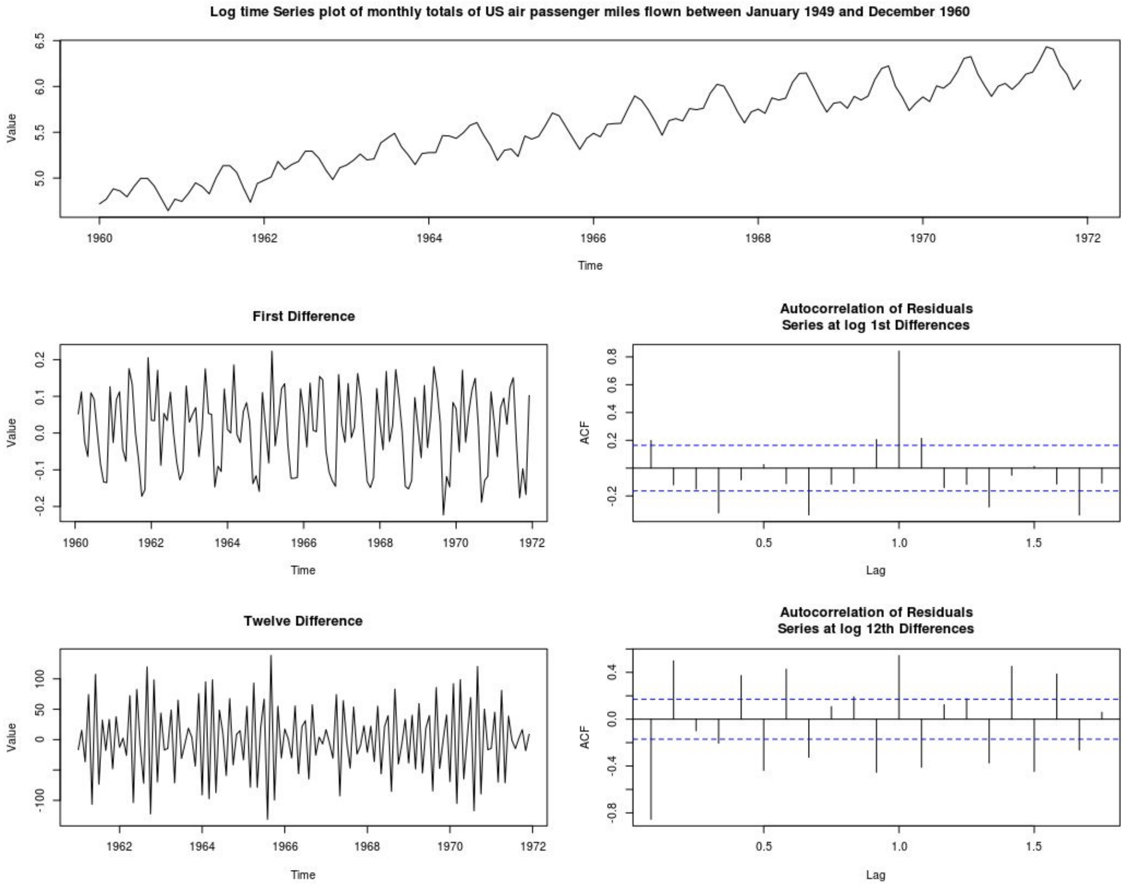


Figure 7: These plots continue from figure 6 however this time the time series and ACF plots are the log of the “Airpass” data. This time, for the log 12th difference plots, they now appear more stationary - logging has made the data vector normal and fits the model better.

Exercise 3

Hares and Roots

```
install.packages("forecast")
auto.arima()
```

By installing the package, “forecast” and using that library, I was able to use the automatic ARMA function model, the automatic ARMA function generate a set of optimal (p, d, q) - it searches through combinations of order parameters and picks the set that optimizes model fit criteria. I compared AR(1), AR(2), and AR(3) with the automatic results.

```
data(hare)
arima(hare, order=c(1,0,0))
arima(hare, order=c(2,0,0))
arima(hare, order=c(3,0,0))
auto.arima(hare)
```

```
> data(hare)
> arima(hare, order=c(1,0,0))

Call:
arima(x = hare, order = c(1, 0, 0))

Coefficients:
         ar1  intercept
      0.6878    38.9820
s.e.  0.1240    10.5377

sigma^2 estimated as 383:  log likelihood = -136.5,  aic = 277
> arima(hare, order=c(2,0,0))

Call:
arima(x = hare, order = c(2, 0, 0))

Coefficients:
         ar1      ar2  intercept
      1.3021  -0.7971    38.2165
s.e.  0.1260   0.1203     4.9255

sigma^2 estimated as 176.6:  log likelihood = -125.57,  aic = 257.13
> arima(hare, order=c(3,0,0))

Call:
arima(x = hare, order = c(3, 0, 0))

Coefficients:
         ar1      ar2      ar3  intercept
      1.0956  -0.4321  -0.2778    38.0405
s.e.  0.1884   0.2846   0.1951     3.8841

sigma^2 estimated as 164.5:  log likelihood = -124.6,  aic = 257.21
> auto.arima(hare)
Series: hare
ARIMA(2,0,0) with non-zero mean

Coefficients:
         ar1      ar2     mean
      1.3021  -0.7971  38.2165
s.e.  0.1260   0.1203   4.9255

sigma^2 estimated as 195.6:  log likelihood=-125.57
AIC=259.13   AICc=260.67   BIC=264.87
>
```

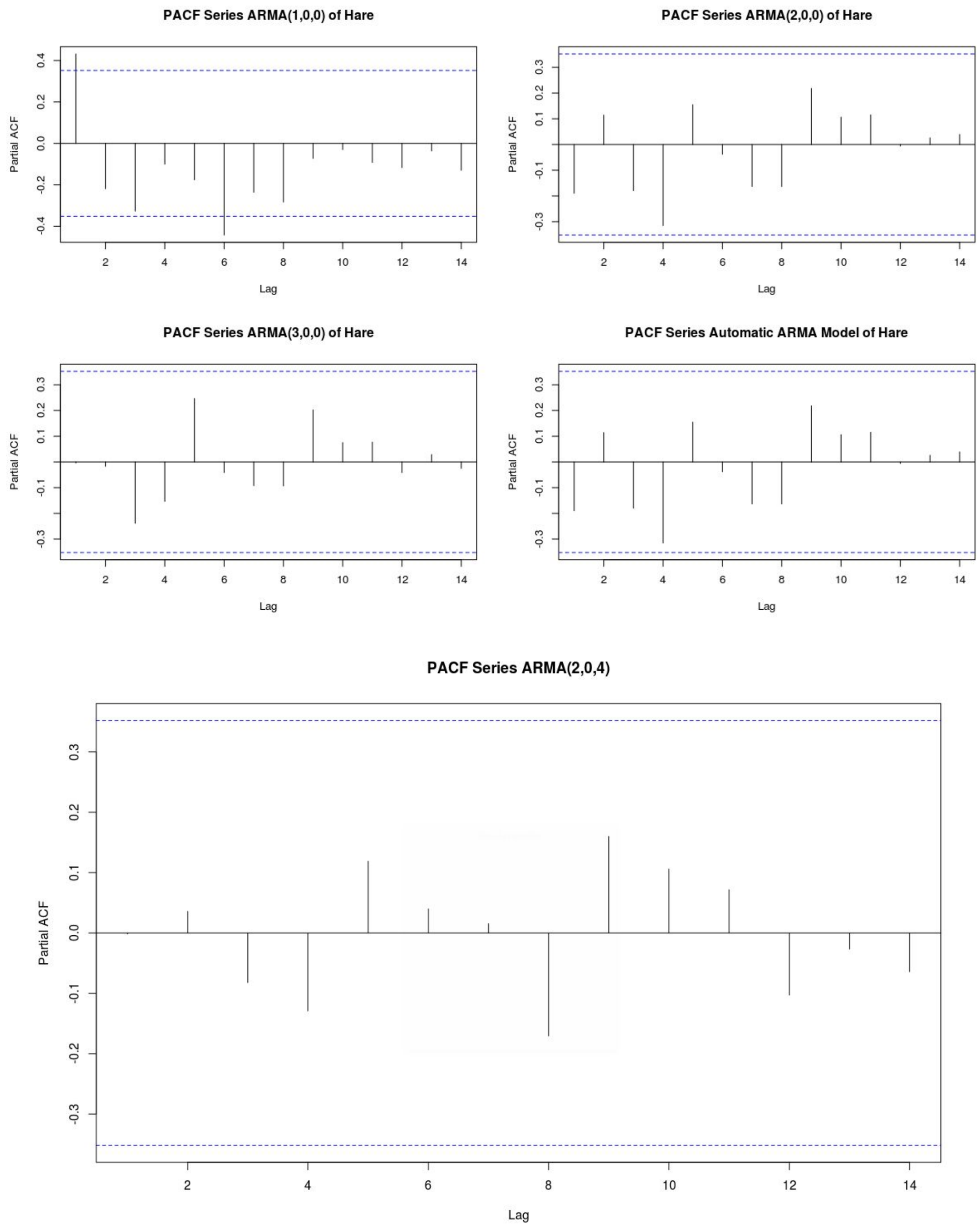
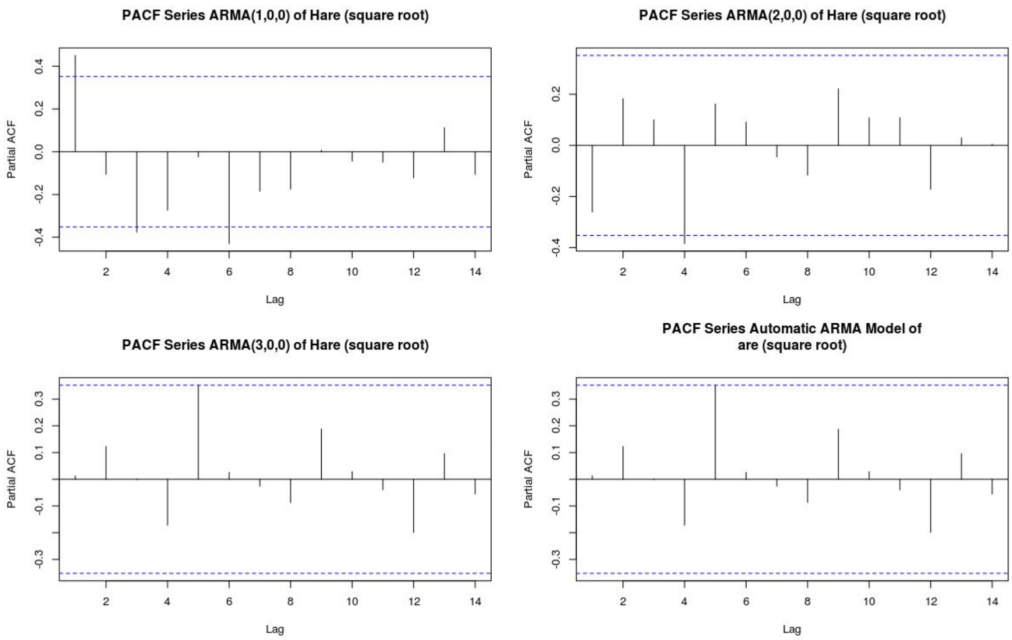


Figure 8: From the above tables, for the square root data of Hare, `ARMA(3,0,0)` were the best fitting values. When plotting the `PACF()` graphs, the plots at lag 4 suggests we can alter the ARMA model further, as see I created a `PACF` plot of `ARMA(2,0,4)` to create a plot much more reliable.

```
sqrt.hare <- sqrt(hare)
arima(sqrt.hare, order=c(1,0,0))
arima(sqrt.hare, order=c(2,0,0))
arima(sqrt.hare, order=c(3,0,0))
auto.arima(sqrt.hare)
```

```
> sqrt.hare <- sqrt(hare)
> arima(sqrt.hare, order=c(1,0,0))

Call:
arima(x = sqrt.hare, order = c(1, 0, 0))

Coefficients:
         ar1  intercept
      0.7275     5.8120
s.e.  0.1157     0.9723

sigma^2 estimated as 2.55:  log likelihood = -58.87,  aic = 121.75
> arima(sqrt.hare, order=c(2,0,0))

Call:
arima(x = sqrt.hare, order = c(2, 0, 0))

Coefficients:
         ar1      ar2  intercept
      1.3514  -0.7763     5.7134
s.e.  0.1286   0.1242     0.4753

sigma^2 estimated as 1.223:  log likelihood = -48.46,  aic = 102.91
> arima(sqrt.hare, order=c(3,0,0))

Call:
arima(x = sqrt.hare, order = c(3, 0, 0))

Coefficients:
         ar1      ar2      ar3  intercept
      1.0519  -0.2292  -0.3931     5.6923
s.e.  0.1877   0.2942   0.1915     0.3371

sigma^2 estimated as 1.066:  log likelihood = -46.54,  aic = 101.08
> auto.arima(sqrt.hare)
Series: sqrt.hare
ARIMA(3,0,0) with non-zero mean

Coefficients:
         ar1      ar2      ar3    mean
      1.0519  -0.2292  -0.3931  5.6923
s.e.  0.1877   0.2942   0.1915  0.3371

sigma^2 estimated as 1.224:  log likelihood=-46.54
AIC=103.08   AICc=105.48   BIC=110.25
>
```



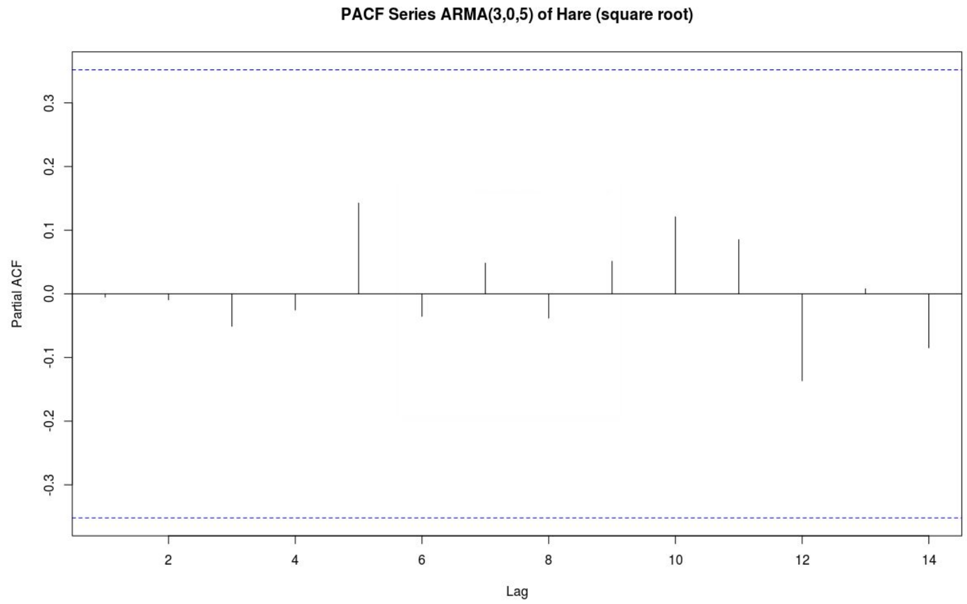
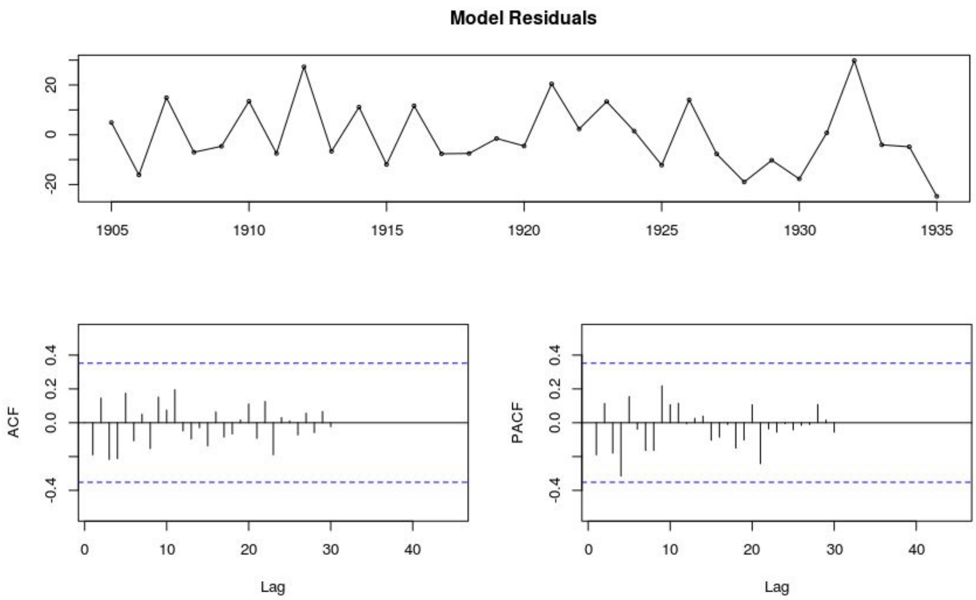


Figure 9: From the above tables for the square root data of Hare, `ARMA(3,0,0)` were the best fitting values. The plots at lag 5 suggests we can alter the ARMA model further, as see I created a `PACF` plot of `ARMA(3,0,5)` to create a plot much more reliable.

From the above tables all together, when we compare the results for both, we can see the AICc score for the square root is smaller.



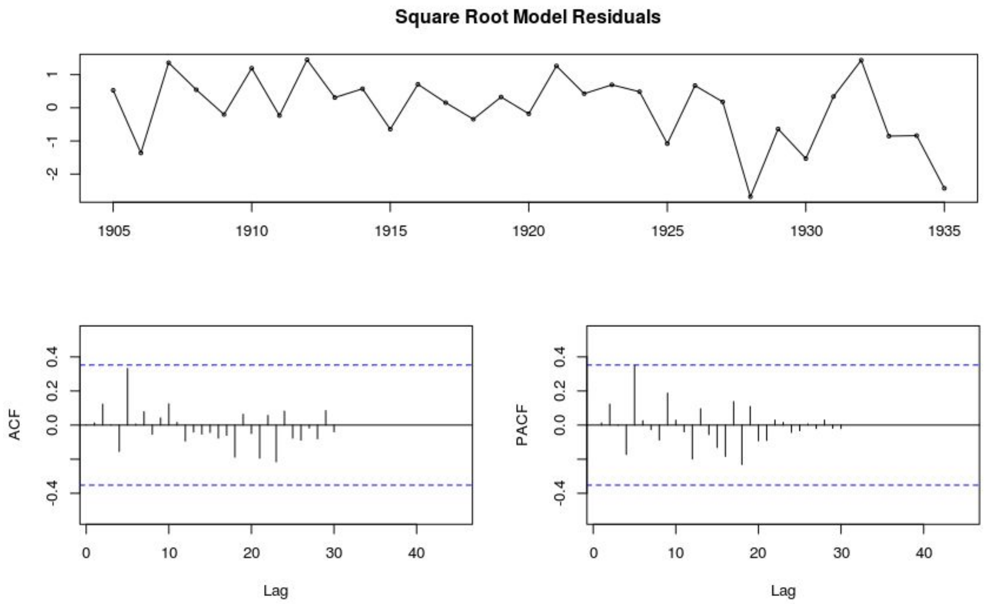


Figure 10: I have plotted the model residuals; for the untransformed data, we can see from the `ACF` plot that there is a clear pattern present in ACF/PACF and model residuals plots repeating at lag 4 for the untransformed data and lag 5 for the square root data - this suggests that our model may be better off with a different specification, such as $p = 4$ plus $p = 5$ for square root. These graphs, using `tsdisplay()` function, backs up our argument as said previously.

Exercise 4

Forecasting

a)

```
> auto.arima(sim.40)
Series: sim.40
ARIMA(0,1,0)

sigma^2 estimated as 1.106:  log likelihood=-57.31
AIC=116.61   AICc=116.72   BIC=118.28
> arima(sim.40, order=c(0,1,0), method="ML")

Call:
arima(x = sim.40, order = c(0, 1, 0), method = "ML")


sigma^2 estimated as 1.106:  log likelihood = -57.31,  aic = 114.61
>
```

Using the first 40 values of the simulated series, I used “ML” as a method for the ARIMA function. I used `auto.arima()` first to figure out the best order for ARIMA to find the values of the maximum likelihood estimates of ϕ and μ (as seen above).

b)

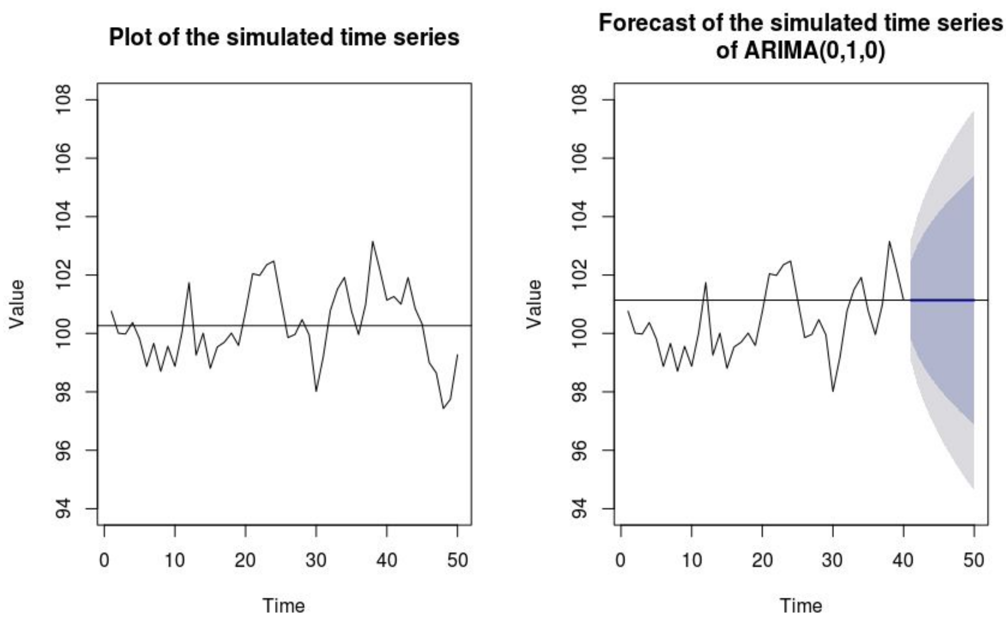


Figure 11: Above are plots of the simulated time series plus a forecast plot - using the estimated model from above, I produced a forecast minus the last 10 values - the forecast displays what it expects the next ten values should range inside. A horizontal line is placed at the estimate of the process mean of the forecasted plot. We can observe that the means for both differ with the forecast values increasing the average.

c)

```
> sim
Time Series:
Start = 1
End = 50
Frequency = 1
 [1] 100.76037 100.00773  99.97993 100.37381  99.81039  98.86910  99.65866  98.70437  99.56161  98.87459 100.03470 101.74040  99.25979
[14] 100.00852  98.80728  99.53672  99.70007 100.01152  99.58910 100.74587 102.04223 101.98658 102.35199 102.47710 101.14787  99.85923
[27]  99.96715 100.47338  99.95010  98.01374  99.20251 100.78645 101.51408 101.91785 100.75737  99.96081 100.99771 103.15186 102.18378
[40] 101.13908 101.26310 101.00197 101.90956 100.85791 100.31546  98.99960  98.64760  97.42812  97.75230  99.26885
> sim.10
Time Series:
Start = 41
End = 50
Frequency = 1
 [1] 101.26310 101.00197 101.90956 100.85791 100.31546  98.99960  98.64760  97.42812  97.75230  99.26885
> 
```

```
Forecasts:
   Point Forecast     Lo 80    Hi 80    Lo 95    Hi 95
41       101.1391 99.79102 102.4871 99.07740 103.2008
42       101.1391 99.23264 103.0455 98.22343 104.0547
43       101.1391 98.80417 103.4740 97.56815 104.7100
44       101.1391 98.44296 103.8352 97.01572 105.2624
45       101.1391 98.12473 104.1534 96.52903 105.7491
46       101.1391 97.83702 104.4411 96.08902 106.1891
47       101.1391 97.57245 104.7057 95.68439 106.5938
48       101.1391 97.32619 104.9520 95.30777 106.9704
49       101.1391 97.09490 105.1833 94.95404 107.3241
50       101.1391 96.87614 105.4020 94.61948 107.6587
> 
```

The above screenshots display the simulated time series with the last ten values of it and the forecast values: both lower/upper of 80% and 95%. We can see that the actual last 10 values range from 102 and 97, with a mean of 99.74445. The forecast value is 101.1391. Both are very similar with only a difference of 1.39465 between the mean of the actual values and the forecast value.

d)

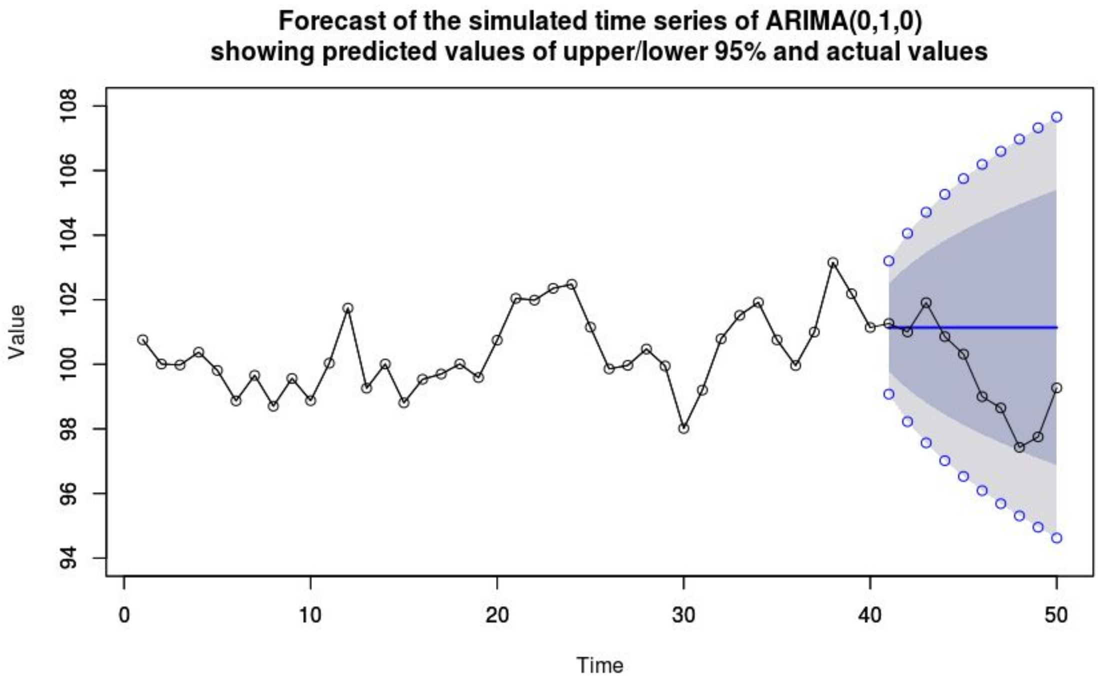


Figure 12: The plot above shows the actual values above the forecast plot and I included the points of the 95% forecast limits. We can see that the actual values fall within the forecast limits making the model/algorithm reliable. The actual values fall in the 80% confidence intervals.