Atmospheric Carbon Dioxide Forecasting

Aidan Jackson, Sandip Panesar

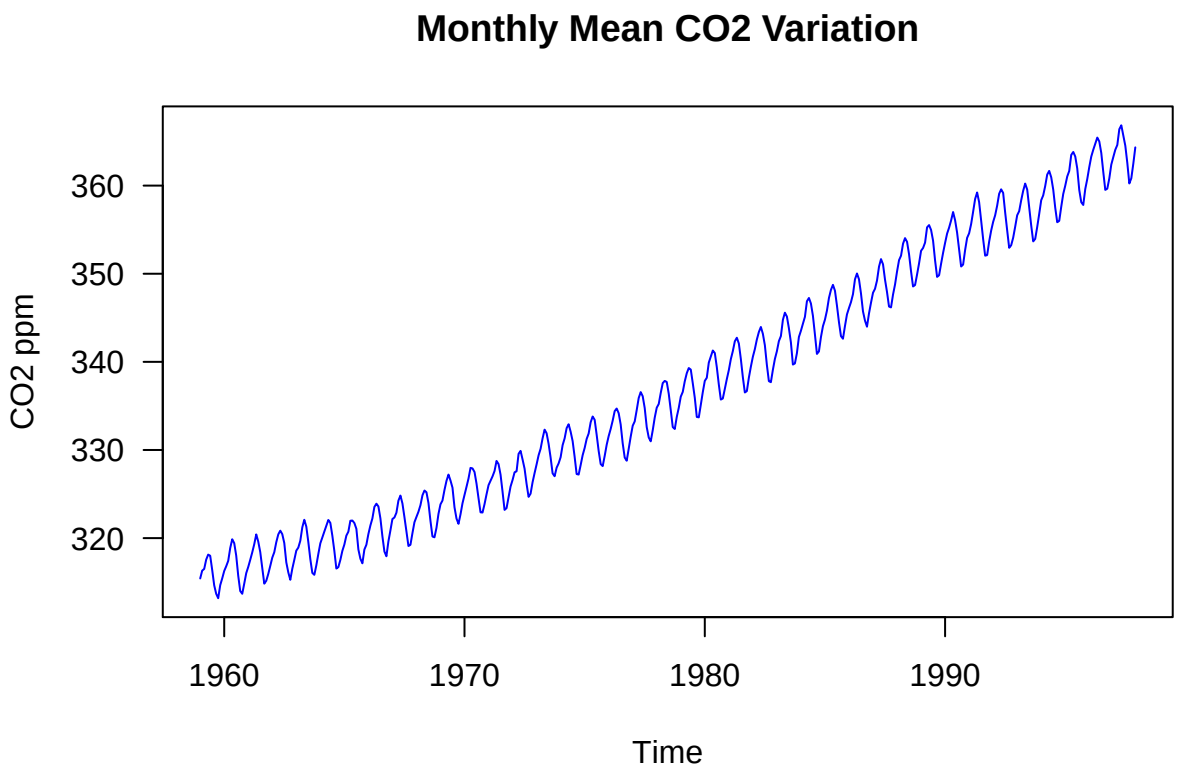
The Keeling Curve

In the 1950s, the geochemist Charles David Keeling observed a seasonal pattern in the amount of carbon dioxide present in air samples collected over the course of several years. He attributed this pattern to varying rates of photosynthesis throughout the year, caused by differences in land area and vegetation cover between the Earth's northern and southern hemispheres.

In 1958 Keeling began continuous monitoring of atmospheric carbon dioxide concentrations from the Mauna Loa Observatory in Hawaii. He soon observed a trend increase carbon dioxide levels in addition to the seasonal cycle, attributable to growth in global rates of fossil fuel combustion. Measurement of this trend at Mauna Loa has continued to the present.

The `co2` data set in R's `datasets` package (automatically loaded with base R) is a monthly time series of atmospheric carbon dioxide concentrations measured in ppm (parts per million) at the Mauna Loa Observatory from 1959 to 1997. The curve graphed by this data is known as the 'Keeling Curve'.

```
plot(co2, ylab = expression("CO2 ppm"), col = 'blue', las = 1)
title(main = "Monthly Mean CO2 Variation")
```



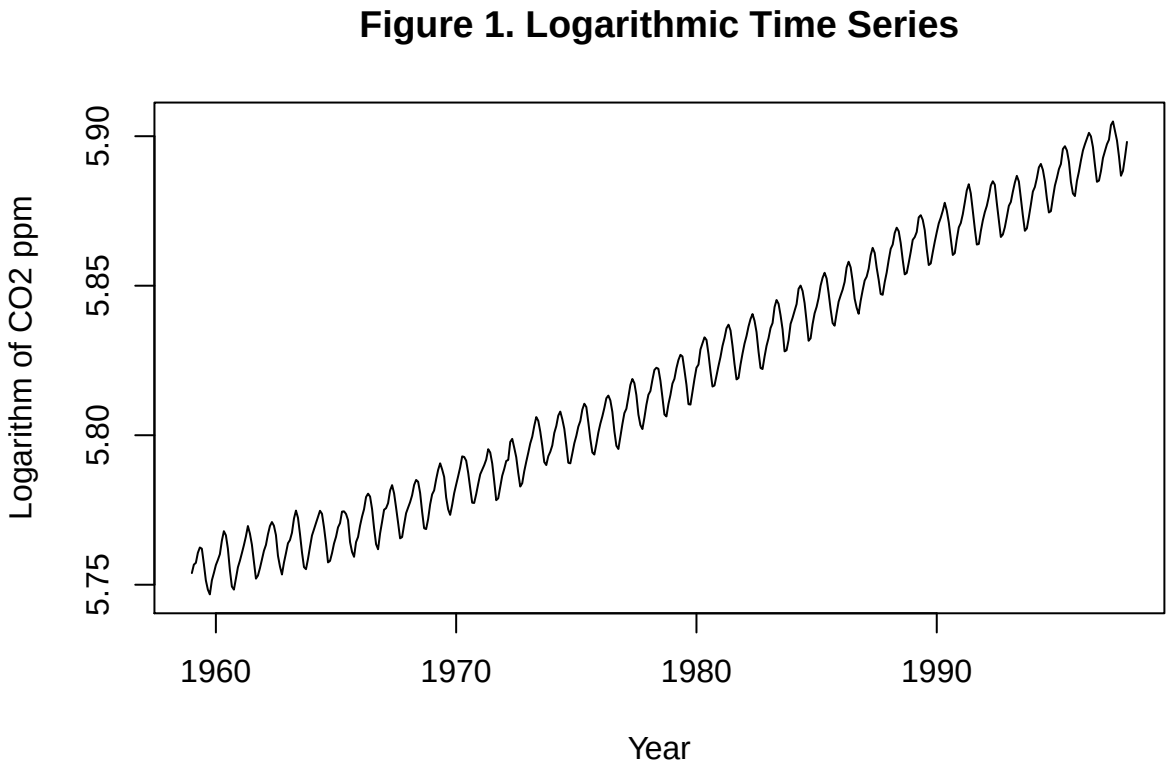
Part 1 (3 points)

Conduct a comprehensive Exploratory Data Analysis on the `co2` series. This should include (without being limited to) a thorough investigation of the trend, seasonal and irregular elements.

```
library(forecast)
library(ggplot2)
library(dplyr)
library(lubridate)
library(zoo)
```

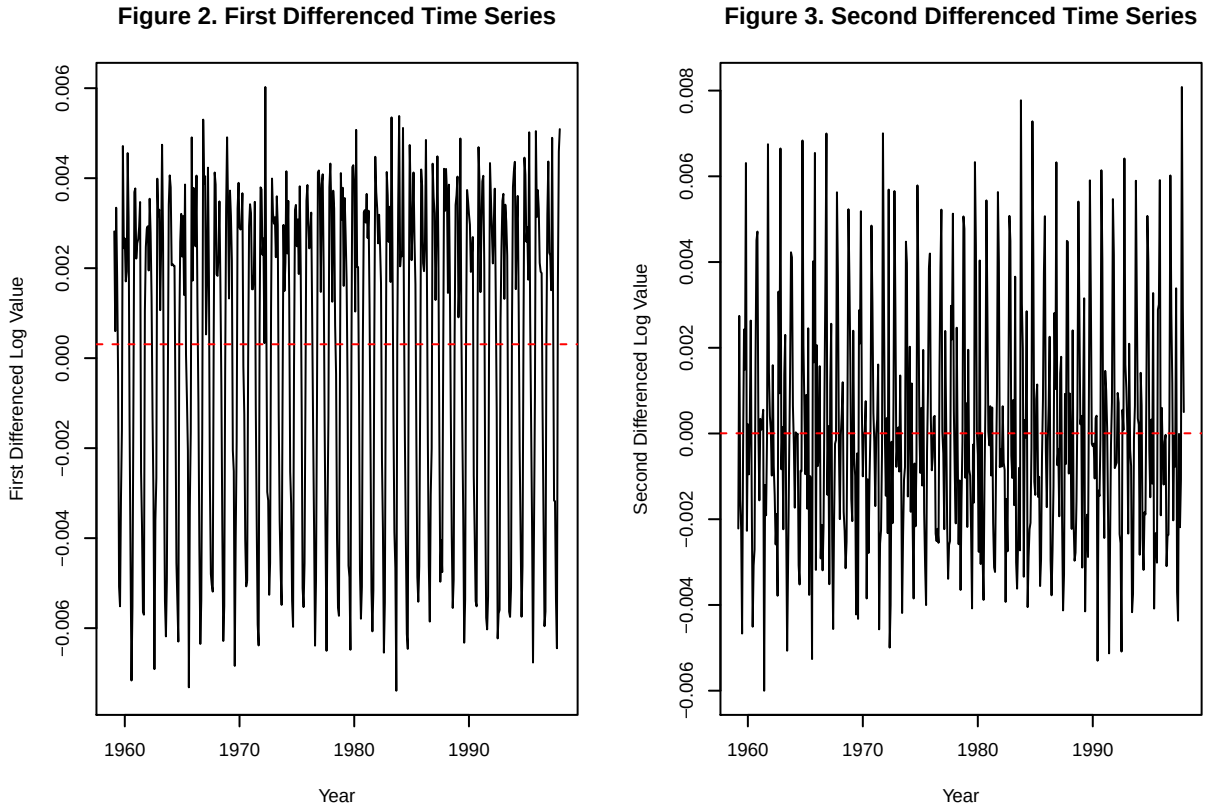
In the initial plot of the monthly CO2 data, there are 0 missing values and it is obvious that there is a positive trend that makes the series non-stationary in the mean. It also appears that the variance could be increasing over time, with the cycles being of greater amplitude later in the series than in earlier sections. This makes the series non-stationary in the variance as well. To address this increasing variance, the logarithm may be taken and examined.

```
d <- log(co2)

plot(d, xlab = "Year", ylab = "Logarithm of CO2 ppm",
     main = "Figure 1. Logarithmic Time Series", type = "l")
```



Shown in **Figure 1**, taking the logarithm appears to have better stabilized the variance over the course of the time series. Now the data can be considered stationary in the variance, but the positive time trend must still be examined. It appears as though the series could either have a first order or second order time trend, so both will be investigated.

```
d1 <- diff(d)
d2 <- diff(d1)

par(mfrow = c(1, 2), cex = 0.6)
plot(d1, xlab = "Year", ylab = "First Differenced Log Value",
     main = "Figure 2. First Differenced Time Series", type = "l")
abline(h = mean(d1), lty = 2, col = "red")
plot(d2, xlab = "Year", ylab = "Second Differenced Log Value",
     main = "Figure 3. Second Differenced Time Series", type = "l")
abline(h = mean(d2), lty = 2, col = "red")
```



Figures 2 and **3** show the first and second differenced times series after applying the previous logarithm transformation. It can be seen in **Figure 2** that the first difference does much to remove the positive time trend, but the majority of the data still appears to slightly fluctuate in the mean over time. For example, the second half of the data appears to have a slightly higher mean value than the first half. On the other hand, **Figure 3** shows that after taking the second difference, the data appears to completely stationary in the mean. Therefore, any model with this data should also take the second difference with $d = 2$ and/or $D = 2$ to achieve this.

Next, the seasonality and other elements of the data will be investigated.

```
par(mfrow = c(1, 2), cex = 0.6)
plot(acf(d2, plot = FALSE), main = "")
title("Figure 4. Autocorrelation Plot")
```

```
plot(pacf(d2, plot = FALSE), main = "")
title("Figure 5. Partial Autocorrelation Plot")
```

Figure 4. Autocorrelation Plot

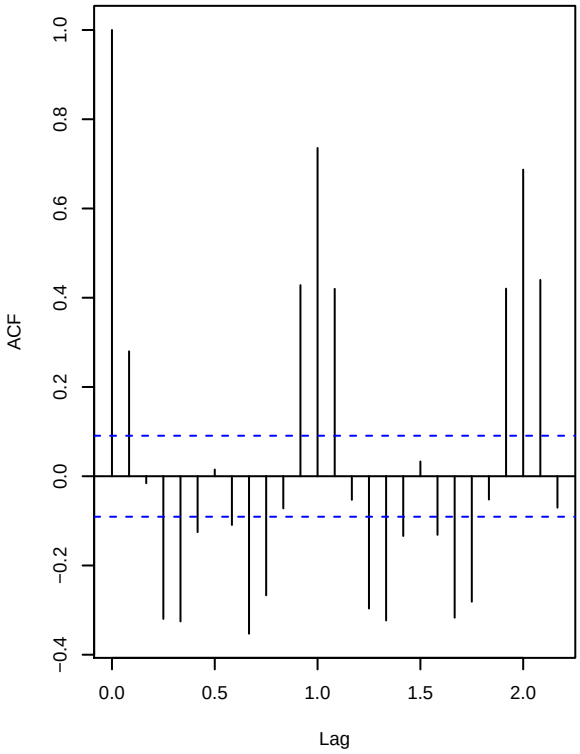
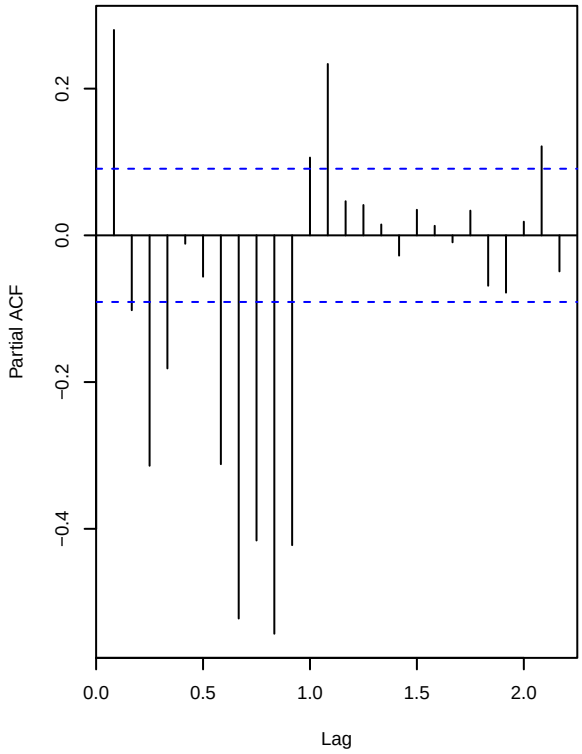


Figure 5. Partial Autocorrelation Plot



Figures 4 and **5** show the ACF and PACF of the data after the logarithm transformation and second differencing. It can be seen there is a significant lag at $k = 1$ and $k = 2$, measured in years. In between these points (and also for the subsequent interval) appears to be a repetitive oscillatory pattern indicating an element of seasonality to this data. This fits with Keeling's original hypothesis that stated annual cyclic changes in the environment and vegetation would affect CO2 concentrations in the atmosphere. Interestingly, the pattern in **Figure 4** assumes a shape where lag is significantly positive but intermediate lags are significantly negative, and with decreasing magnitudes over time-period shifts. Similar to **Figure 5**, these oscillations appears to be decreasing in magnitude over time. These repeating patterns may explain seasonality in the time series, rather than being indicative of the non-seasonal AR/MA components. Full investigation of AR/MA components of the data will be continued in Part 3.

Part 2 (3 points)

Fit a linear time trend model to the `co2` series, and examine the characteristics of the residuals. Compare this to a higher-order polynomial time trend model. Discuss whether a logarithmic transformation of the data would be appropriate. Fit a polynomial time trend model that incorporates seasonal dummy variables, and use this model to generate forecasts up to the present.

As discussed in Part 1, a logarithm transformation should be applied to the data in order to make it stationary in the variance. Having decided to use the logarithm transform, a linear time trend can then be fit to the data without any applied differencing.

```
Time <- time(d)
linear_model <- lm(d ~ Time)

plot(d, xlab = "Year", ylab = "Value", main = "Figure 8. Fitted Linear Model",
     type = "l")
lines(d + linear_model$resid, col = "blue")
legend('bottomright', legend=c("Original", "Linear Model"), col=c("black","blue"),
       lty=1)
```

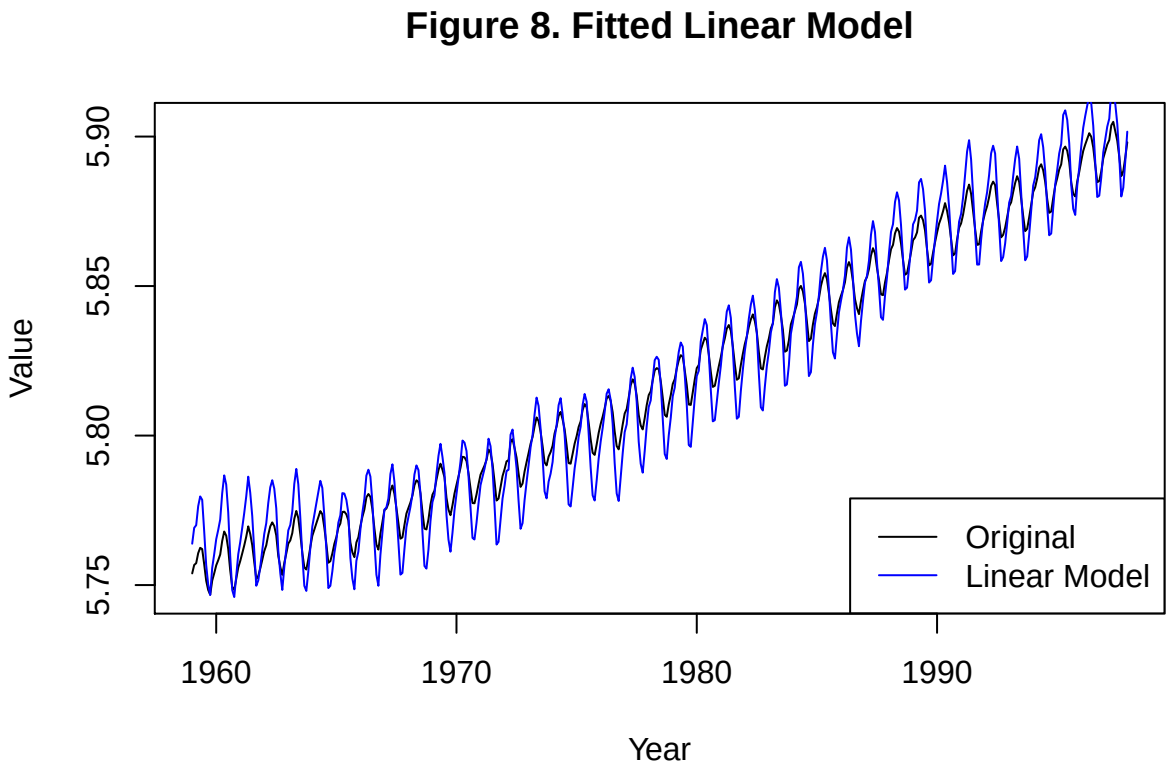


Figure 8 displays the fitted linear model overlaid on top of the original data. Both have been logarithm transformed for comparison. It can be seen that the linear model generally overestimates the amount of variance in the data, with cyclic amplitudes that are much greater than in the original time series. However, it is able to estimate the frequency of the cycles quite well, with an almost exact overlap between the crests and troughs of each series.

The residuals may be examined on their own for an evaluation of the fit.

```
par(mfrow = c(1, 2), cex = 0.6)
plot(acf(linear_model$residuals, plot = FALSE), main = "")
title("Figure 9. Autocorrelation Plot")
plot(pacf(linear_model$residuals, plot = FALSE), main = "")
title("Figure 10. Partial Autocorrelation Plot")
```

Figure 9. Autocorrelation Plot

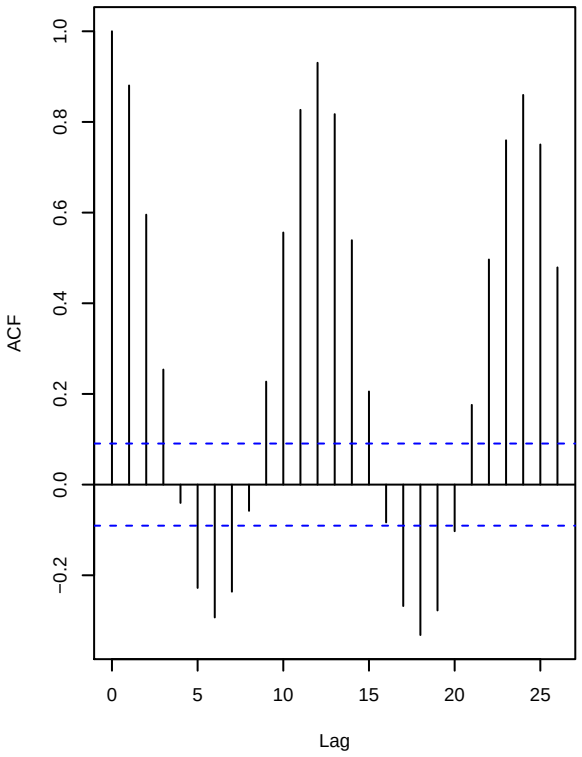
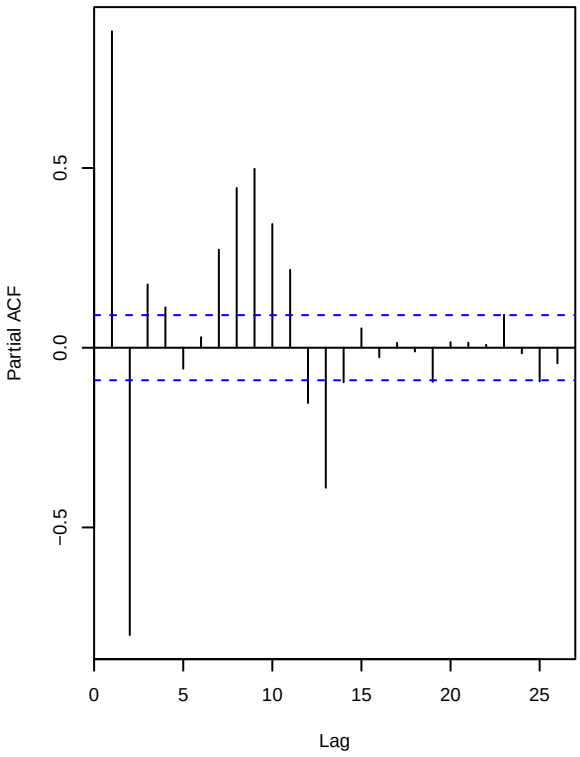


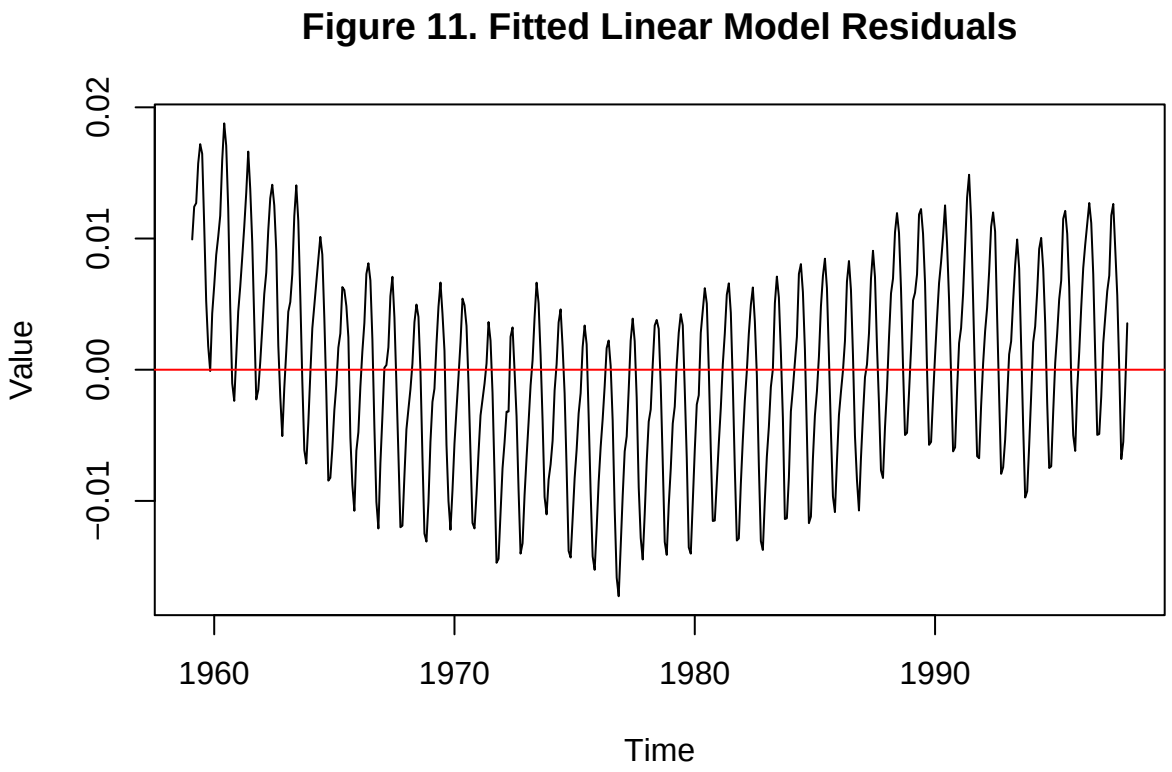
Figure 10. Partial Autocorrelation Plot



Figures 9 and **10** show the ACF and PACF of the linear model's residuals. A slightly damped oscillatory pattern can be observed in both, indicating that the model does not account well for the seasonal variation which is present in the original data. This appears to have a period of 12 months based upon **Figure 9**.

Finally, the residuals can be viewed on their own over time.

```
plot(linear_model$residuals,
     xlab = "Time", ylab = "Value", xaxt = "n",
     main = "Figure 11. Fitted Linear Model Residuals", type = "l")
axis(1, at=c(12,132,252,372), labels=c(1960, 1970, 1980, 1990))
abline(h = mean(linear_model$resid), col = "red")
```



Clearly, the residuals in **Figure 11** are non-stationary in the mean and do not resemble a white noise series. At earlier and later decades, the residuals are consistently higher than the mean. In the middle of the time series, however, they are consistently less than the mean. This demonstrates that there are still components in the original time series that have not been accounted for by the linear model.

For a better fit, a polynomial time trend can be estimated:

```
poly_model <- lm(d ~ Time + I(Time^2) + I(Time^3))

plot(d, xlab = "Year", ylab = "Value", main = "Figure 12. Fitted Polynomial Model",
     type = "l")
lines(d + linear_model$resid, col = "blue", cex = 0.6)
lines(d + poly_model$resid, col = "red", cex = 0.6)
legend('bottomright',
       legend = c("Original", "Linear Model", "Polynomial Model"),
       col=c("black","blue","red"), lty=1)
```

Figure 12. Fitted Polynomial Model

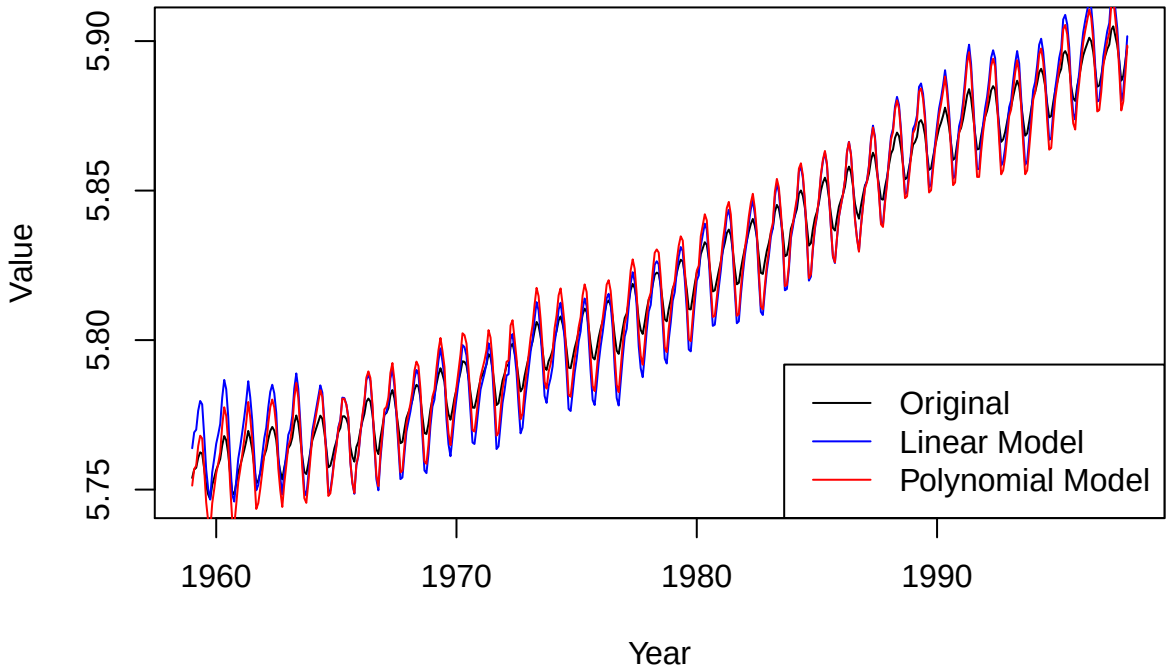
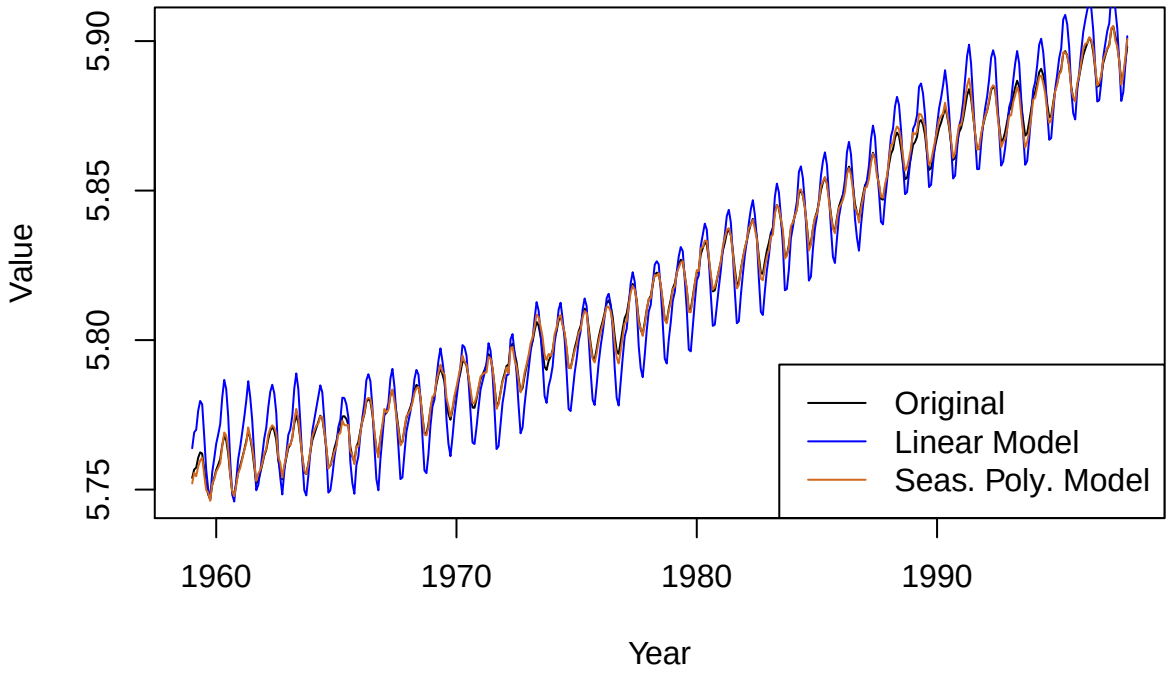


Figure 12 overlays the fitted time trend model of order 3 with the original linear model. It is apparent that there are very few locations during the time series where the original linear model deviated but the polynomial did not. This indicates that using higher orders of time trend alone will not improve the model while the seasonality is still not addressed.

This can be accounted for with the use of seasonal dummy variables.

```
poly_dummy_model <- tslm(formula = d ~ 0 + trend + I(trend^2) +
                           I(trend^3) + season)

plot(d, xlab = "Year", ylab = "Value",
     main = "Figure 13. Fitted Seasonal Polynomial Model", type = "l")
lines(d + linear_model$resid, col = "blue", cex = 0.6)
lines(d + poly_dummy_model$resid, col = "chocolate", cex = 0.6)
legend('bottomright',
       legend = c("Original", "Linear Model", "Seas. Poly. Model"),
       col=c("black","blue","chocolate"), lty=1)
```

Figure 13. Fitted Seasonal Polynomial Model



Demonstrated in **Figure 13**, the use of seasonal dummy variables for each month results in a polynomial time trend model with a dramatic improvement. This is both compared to the original linear model, as well as the previous polynomial model without any seasonal components.

The residuals of this model will be examined similar to what was done for the linear model:

```
par(mfrow = c(1, 2), cex = 0.6)
plot(acf(poly_dummy_model$residuals, plot = FALSE), main = "")
title("Figure 14. Autocorrelation Plot")
plot(pacf(poly_dummy_model$residuals, plot = FALSE), main = "")
title("Figure 15. Partial Autocorrelation Plot")
```

Figure 14. Autocorrelation Plot

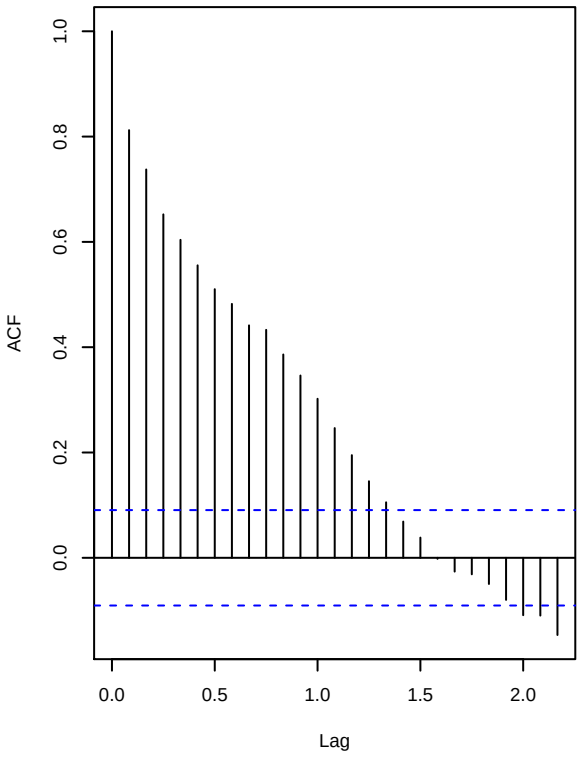
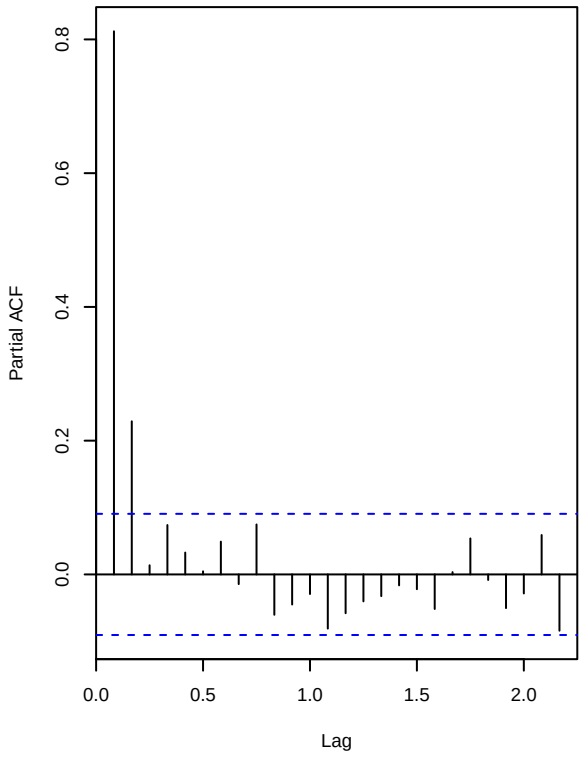


Figure 15. Partial Autocorrelation Plot



Figures 14 and **15** display the ACF and PACF of the residuals from the polynomial model with seasonal dummy variables. The cyclic components found with the linear model in **Figures 9** and **10** are no longer visible in these examples. **Figure 14** shows heavy autocorrelation until lag ~15, while **Figure 15** shows that the first lag still has a very significant partial autocorrelation associated with it.

The residuals appearance can also be directly examined.

```
plot(poly_dummy_model$residuals,
     xlab = "Time", ylab = "Value", xaxt = "n",
     main = "Figure 16. Seasonal Polynomial Model Residuals", type = "l")
axis(1, at=c(12,132,252,372), labels=c(1960, 1970, 1980, 1990))
abline(h = mean(poly_dummy_model$resid), col = "red")
```

Figure 16. Seasonal Polynomial Model Residuals

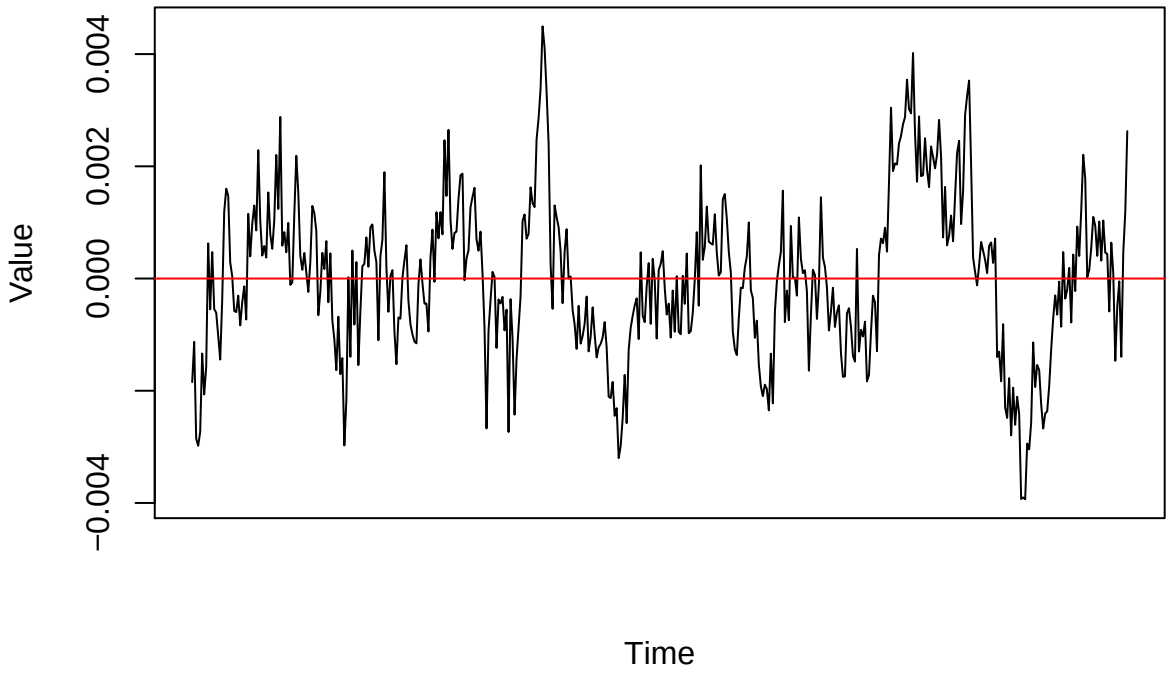


Figure 16 shows the residuals of the polynomial model with seasonal dummy variables. Though still fluctuating, the residuals now appear stationary in the mean compared to **Figure 11**. This is indicative of an improved fit. While there is still some cyclic pattern that can be observed in this series, it has an appearance much closer to that of white noise. These cycles still indicate that there may be components in the original time series still not accounted for by the model, however.

Finally, this best performing polynomial model can also be used to forecast values to the present.

```
# make predictions 20 years from end of time series
predictions1 <- forecast(poly_dummy_model, h = 240)
pred.ts <- ts(predictions1$mean, start=c(1998,1), frequency=12)
ci.df <- data.frame(avg=predictions1$mean, upr=predictions1$upper[,2],
                    lwr=predictions1$lower[,2])

ggplot() +
  geom_line(data = d, aes(x = time(d), y = d,
                          colour = "Original Values")) +
  geom_line(data=pred.ts, aes(x=time(pred.ts), y=pred.ts,
                              colour="Forecasted Values")) +
  geom_line(aes(x = 1959+(1:(39*12))/12,
                          y = d + poly_dummy_model$residuals,
                colour="Fitted Values")) +
  geom_ribbon(data=ci.df, aes(x=time(pred.ts),
                              ymin=lwr, ymax=upr), alpha=0.5) +
  labs(y = "Value", x = "Year", title = "Figure 17. Polynomial Model Forecasts")
```

Figure 17. Polynomial Model Forecasts

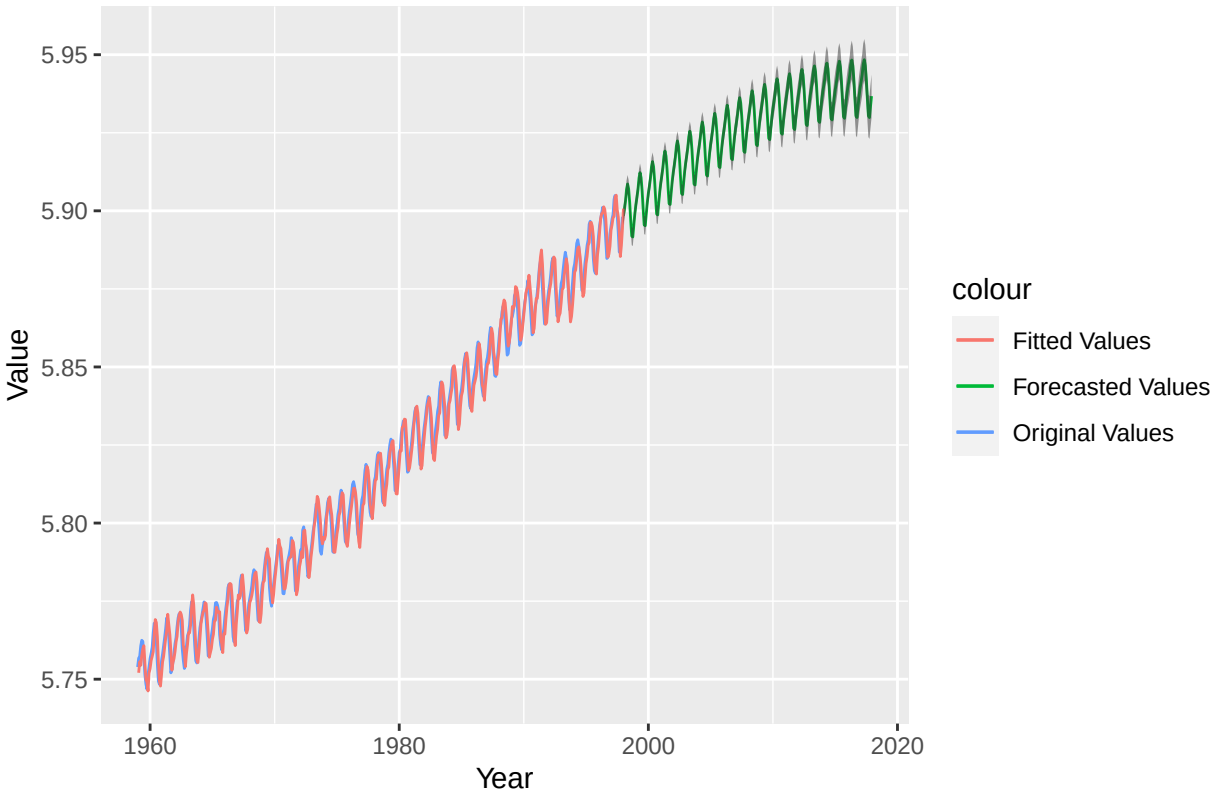


Figure 17 shows the original time series along with the polynomial trend + seasonal dummy variable fit. The forecast of the fitted model 20 years into the future, reaching our present, is also shown. Finally, the confidence interval of the forecast is also shown in the transparent coloring around the values. It can be seen that these confidence intervals are quite tight to the forecast and do not project a large amount of uncertainty in the prediction. In general, uncertainty increases further into the future but not by an excessive amount.

In addition, the forecast generally predicts CO2 concentrations to continue to increase, but at a decreasing rate. By the year 2020, it almost appears as if the forecast expects them to level off. However, knowing that CO2 concentrations have continued to increase at about the same rate, this is inaccurate. Because of this, the model predicts a CO2 concentration of about ~403 ppm by the year 2020 when in reality the measured value is almost 420 ppm.

Part 3 (4 points)

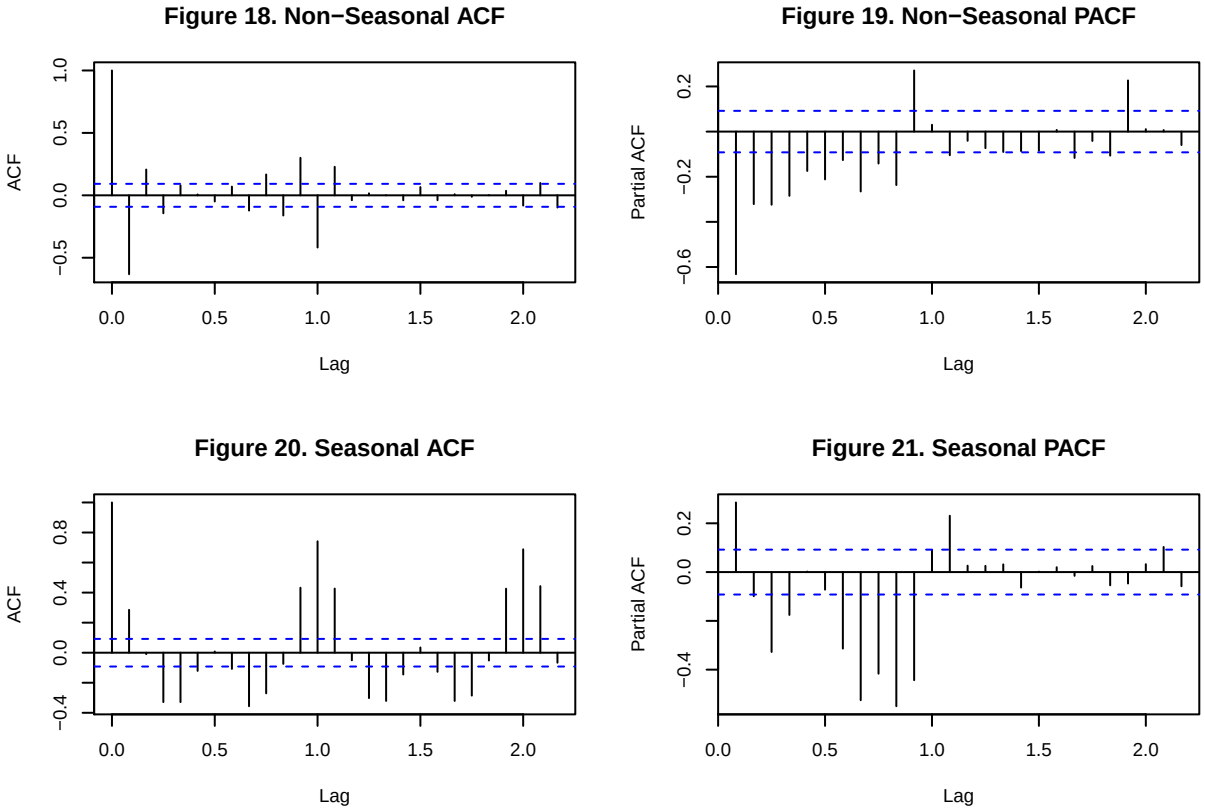
Following all appropriate steps, choose an ARIMA model to fit to this `co2` series. Discuss the characteristics of your model and how you selected between alternative ARIMA specifications. Use your model to generate forecasts to the present.

As shown in **Figures 2** and **3** in Part One, a second differenced time series is a better fit to the data than one with no differencing or only one difference. This would result in a value of $d = 2$ for the ARIMA model and potentially $D = 2$ for the seasonal component.

Likewise, the ACF and PACF of the **Figures 4** and **5** can also be used to estimate the seasonality of the model. In both figures it can clearly be seen that there is an annual seasonal pattern, which should be included to estimate a SARIMA model. With this determined, the time series can be split into seasonal and non-seasonal components to investigate the individual potential AR/MA

terms of each.

```
d_non_seas <- diff(d2, lag = 12)
d_seas <- d2 - d_non_seas

par(mfrow = c(2, 2), cex = 0.6)
plot(acf(d_non_seas, plot = FALSE), main = "")
title("Figure 18. Non-Seasonal ACF")
plot(pacf(d_non_seas, plot = FALSE), main = "")

title("Figure 19. Non-Seasonal PACF")
plot(acf(d_seas, plot = FALSE), main = "")
title("Figure 20. Seasonal ACF")
plot(pacf(d_seas, plot = FALSE), main = "")
title("Figure 21. Seasonal PACF")
```



Figures 18-19 show the ACF and PACF of the non-seasonal time series component while **Figures 20-21** show the same information for the seasonal component. It can be seen that the majority of the cyclic pattern is removed from the non-seasonal component once the lag difference was subtracted, but that some still remains with smaller magnitude. For the non-seasonal component, the single significant lag at *k=1* in both **Figures 18** and **19** suggests $p = q = 1$. There are more significant lags in **Figure 19**, but upon comparison with **Figure 21** it appears these could be related to seasonality which was imperfectly removed. For the seasonal component of the time series, a similar observation can be made of **Figure 20** and **21** that suggests $P = Q = 1$. Likewise,

the significant lag at $k = 3$ in **Figure 21** suggests that P could be higher order, but this may still be a product of the seasonality that is imperfectly isolated.

Although these values may be what are expected when manually examining the time series, the use of the AIC to differentiate this potential model with others like it can also be used. This will be investigated for SARIMA models which vary from the above specifications by one or two orders for each of the components. To simplify the search, and because the order of the terms appeared similar above, the seasonal and non-seasonal components will be assumed to be of the same order (i.e. $p = P$, $d = D$, and $q = Q$).

```
# placeholders
aic <- 0
arima_model <- NA

for (p in 1:3) {
  for (q in 0:1) {
    for (d_arima in 1:2) {
      model <- try(arima(d[], order = c(p,d_arima,q),
                         seasonal = list(order = c(p, d_arima, q), period = 12)),
                   silent = TRUE)
      if (length(model) > 1) {
        if (model$aic < aic) {
          d_model <- d_arima
          aic <- model$aic
          arima_model <- model
        }
      }
    }
  }
}

cat("Order of Differencing:", d_model, "\n")
```

```
## Order of Differencing: 1
```

```
print(arima_model$coef)
```

```
##         ar1         ar2         ma1        sar1        sar2        sma1
##  0.35919671  0.09848122 -0.71127156  0.02266567 -0.08062029 -0.90213416
```

Shown above, the SARIMA model with the best AIC had an order of $p = P = 2$, $d = D = 1$, and $q = Q = 1$. Notably, not performing a second differencing was found to work best with the time series. Having determined the order of each of the components, the SARIMA model can be created.

```
arima_model <- arima(d, order = c(2, 1, 1),
                     seasonal = list(order = c(2,1,1)))

plot(d, xlab = "Year", ylab = "Value",
     main = "Figure 22. Fitted SARIMA Model", type = "l")
lines(d + arima_model$resid, col = "red", cex = 0.6)
legend('bottomright',
       legend = c("Original", "SARIMA Model"),
       col=c("black","red"), lty=1)
```

Figure 22. Fitted SARIMA Model

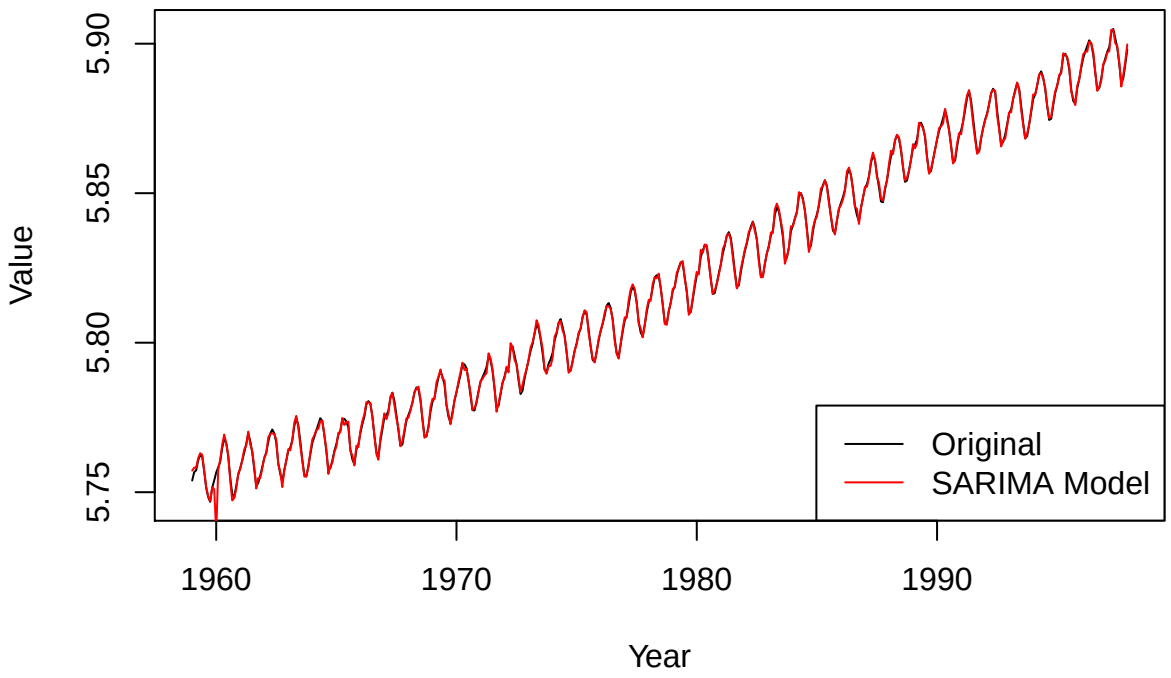


Figure 22 overlays the AIC-estimated SARIMA model with the original data. It can be seen that except for a small amount of jitter early in the time series, the model produces a very good fit.

The residuals may also be examined to investigate whether there is any remaining pattern not captured by the model.

```
plot(arima_model$residuals,
     xlab = "Time", ylab = "Value", xaxt = "n",
     main = "Figure 23. SARIMA Model Residuals", type = "l")
axis(1, at=c(12,132,252,372), labels=c(1960, 1970, 1980, 1990))
abline(h = mean(arima_model$resid), col = "red")
```

Figure 23. SARIMA Model Residuals

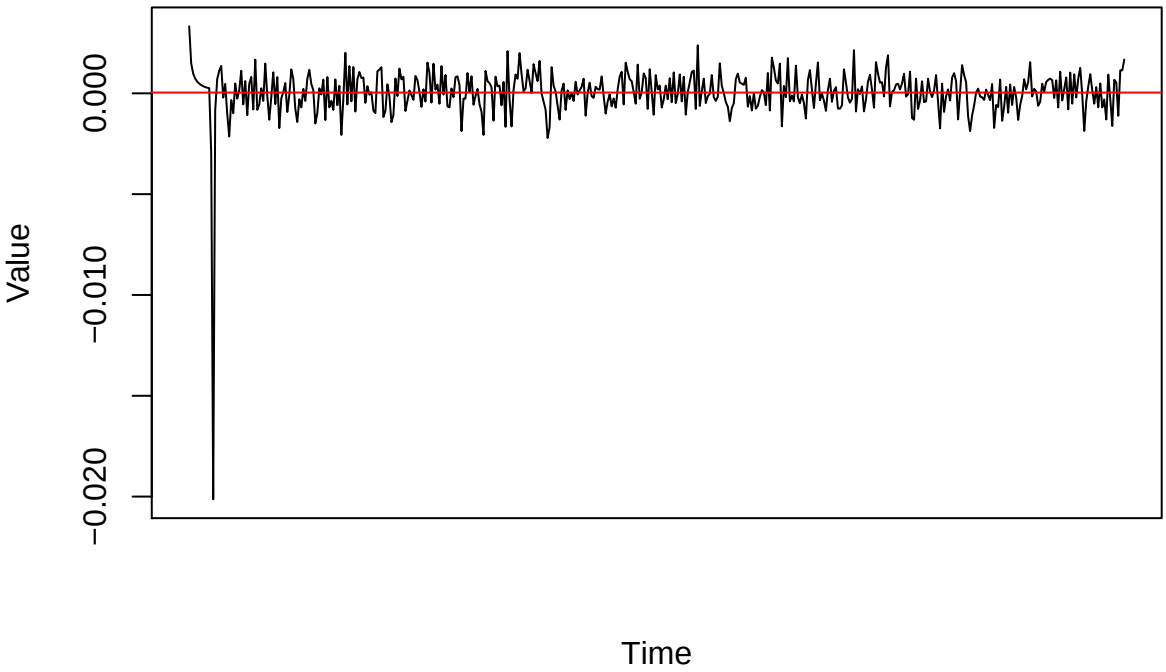


Figure 23 displays the residuals over time for the model, along with their mean value. There is a large fluctuation in the beginning of the model, corresponding to the error that is also seen in **Figure 22**. However, the rest of the series appears to resemble white noise indicating that the model specification is a good fit.

Finally, the model can be used to forecast values to the present.

```
# make predictions 20 years from end of time series
predictions2 <- forecast(arima_model, h = 240)
pred2.ts <- ts(predictions2$mean, start=c(1998,1), frequency=12)
ci.df2 <- data.frame(avg=predictions2$mean, upr=predictions2$upper[,2],
                     lwr=predictions2$lower[,2])

ggplot() +
  geom_line(data=pred2.ts, aes(x=time(pred2.ts), y=pred2.ts,
                               colour="Forecasted Values")) +
  geom_line(aes(x = 1959+(1:(39*12))/12,
                          y = d + arima_model$residuals,
                colour="Fitted Values")) +
  geom_ribbon(data=ci.df2, aes(x=time(pred2.ts),
                               ymin=lwr, ymax=upr), alpha=0.5) +
  labs(y = "Value", x = "Year", title = "Figure 24. SARIMA Model Forecast")
```

Figure 24. SARIMA Model Forecast

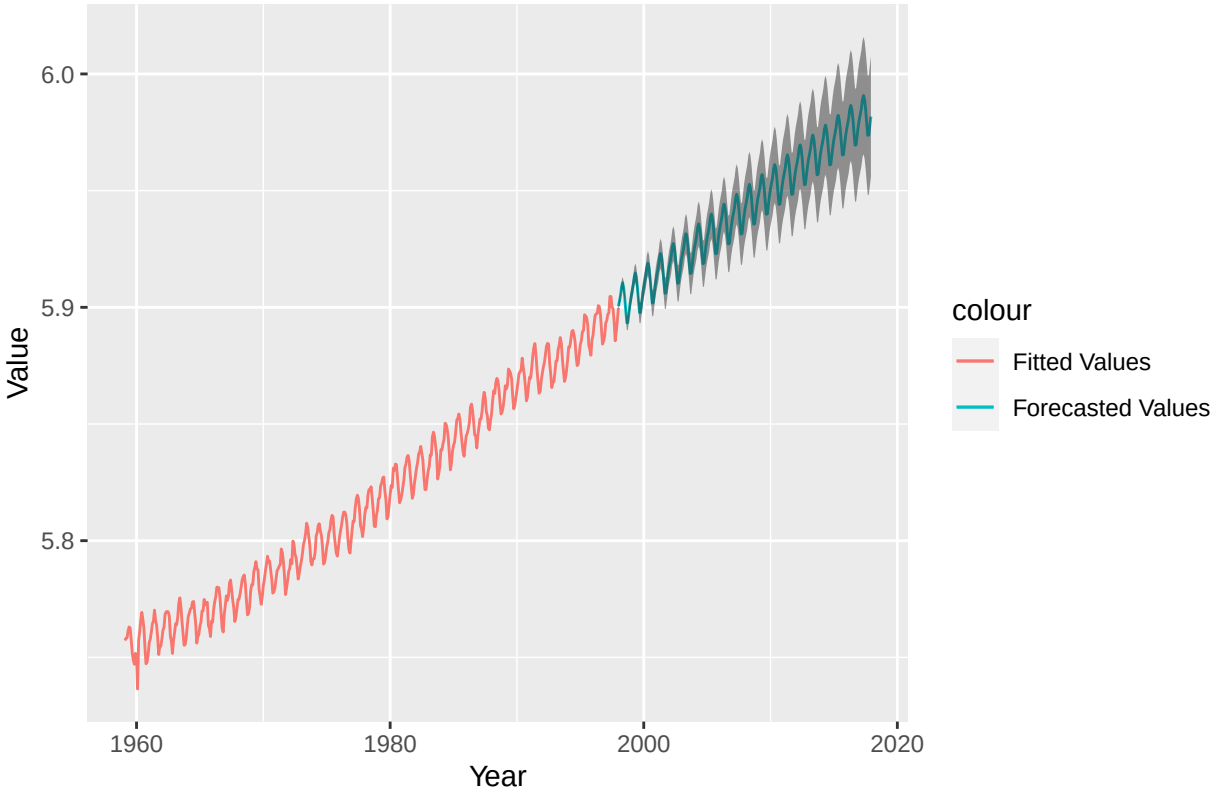


Figure 24 shows the forecast of the SARIMA model to the near present along with the 80% and 95% confidence intervals. As expected, the further into the future the model is used for forecasting, the wider the bounds become on the forecast. At the most recent point the model predicts a CO2 concentration of about ~396 ppm, which is greater than the polynomial model and closer to the true recorded value of ~420 ppm. This is likely because the polynomial model of order 3 increases at a decreasing rate over time, whereas the SARIMA model continues closer to the observed rate in the time series. With this, the SARIMA model appears to be a better fit to the time series than the linear or polynomial trend models even though there is greater uncertainty in its predictions.

Part 4 (5 points)

The file `co2_weekly_mlo.txt` contains weekly observations of atmospheric carbon dioxide concentrations measured at the Mauna Loa Observatory from 1974 to 2020, published by the National Oceanic and Atmospheric Administration (NOAA). Convert these data into a suitable time series object, conduct a thorough EDA on the data, addressing the problem of missing observations and comparing the Keeling Curve's development to your predictions from Parts 2 and 3. Use the weekly data to generate a month-average series from 1997 to the present and use this to generate accuracy metrics for the forecasts generated by your models from Parts 2 and 3.

```
# Read table skipping all the irrelevant text
w <- read.table("co2_weekly_mlo.txt", skip=47)

#rename columns
colnames(w) <- c("yr", "mon", "day", "decimal", "ppm",
                 "no.days", "1yr", "10yr", "since.1800")
```

```r
# Merge
w <- w %>%
  mutate(date = make_date(yr, mon, day))
```

Based upon the raw dataset, the questions being being asked and our previous analysis, we are concerned with the actual measured values of carbon dioxide (in ppm) in the dataset. Nevertheless, the dataset seems to contain a lot of other data that we may or may not need. For the moment, we can focus on the `ppm` column and values contained within. First, we need to check if there are any missing data in this column. It appears there are 18 missing weekly measurements. These will have to be dealt with. We could potentially take the mean values of the immediately surrounding data to impute the missing values, however if there are serial missing values then this will be difficult. Lets isolate the rows with missing weekly measurements:

```r
# First drop irrelevant columns
w2 <- w[c('date','ppm')]

subset(w2, w2$ppm==-999.99)
```

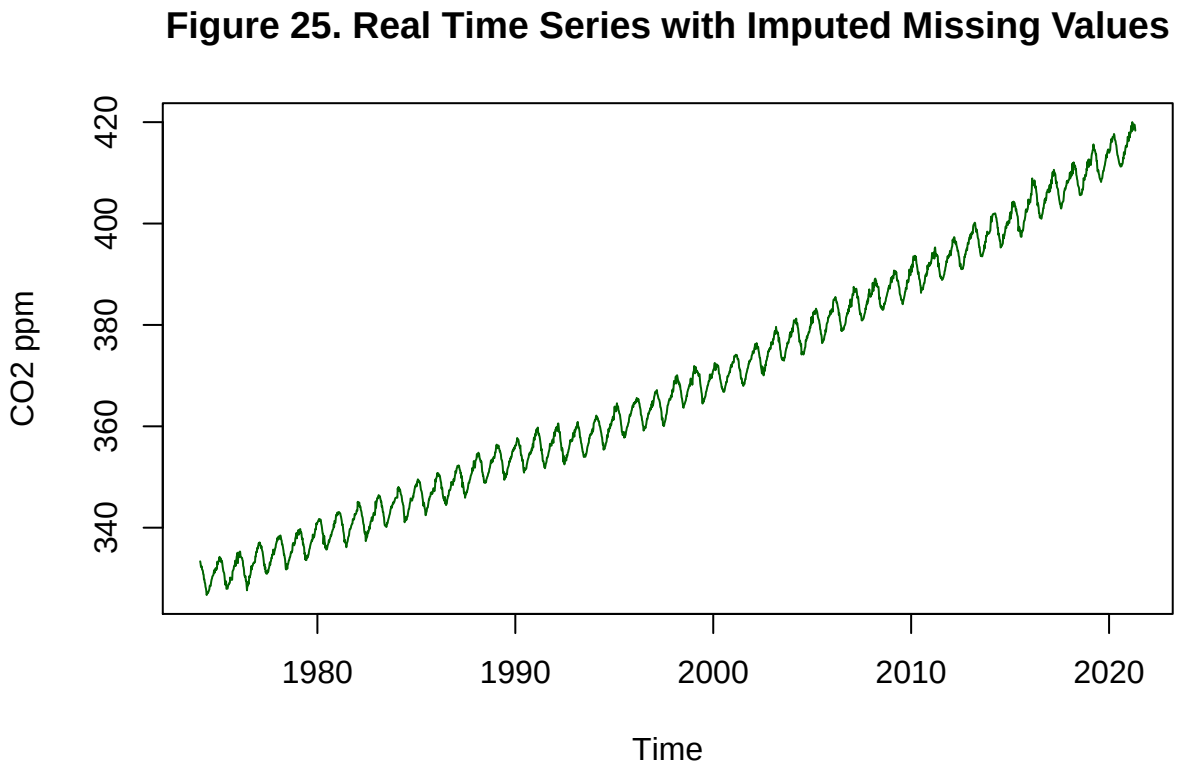
```
##            date     ppm
## 73   1975-10-05 -999.99
## 82   1975-12-07 -999.99
## 83   1975-12-14 -999.99
## 84   1975-12-21 -999.99
## 85   1975-12-28 -999.99
## 111  1976-06-27 -999.99
## 410  1982-03-21 -999.99
## 413  1982-04-11 -999.99
## 414  1982-04-18 -999.99
## 482  1983-08-07 -999.99
## 516  1984-04-01 -999.99
## 517  1984-04-08 -999.99
## 518  1984-04-15 -999.99
## 519  1984-04-22 -999.99
## 1640 2005-10-16 -999.99
## 1781 2008-06-29 -999.99
## 1782 2008-07-06 -999.99
## 1783 2008-07-13 -999.99
```

It appears that there are a mixture of time periods containing missing values. For example, in 1975 there are no measurements for the entire month of December, while in 1982 there are multiple missing values for April. Similarly, in 1984 there are no data for April. As the data have a substantial amount of granularity, we can replace the missing values with the last available value. This will, however mean that for some months e.g. December 1975, the value for all 4 weeks will be identical to the last measurement from November 1975. This might be a better way to handle missing data than simply filling with the mean of the entire series due to the obvious known increase in mean carbon dioxide concentration.

```
# recursively replace consecutive series of missing values
for (i in 1:(length(w2[,2])-1)) {
   if (w2[,2][i+1]==-999.99) {w2[,2][i+1]<- w2[,2][i]}
 }
```

We can then convert the data into a time series object:

```
ts1 <- ts(w2$ppm, start=c(1974, 5), frequency=52)

plot(ts1, main="Figure 25. Real Time Series with Imputed Missing Values",
     ylab=expression("CO2 ppm"), col="darkgreen")
```



From **Figure 25** we can see that this plot of actual measured carbon dioxide levels highly resembles that of the Keeling curve, and the imputation has not produced any obvious or drastic deviations from the overall seasonal and increasing trend. Unlike the original `co2` series, it does not appear that the variance of the trend is increasing. We can double check this by decomposing the time series into constituent parts:

```
decomp.ts1 <- decompose(ts1, type="additive")
plot(decomp.ts1$seasonal,
     main="Figure 26. Seasonal Component of Real Time Series",
     ylab="Deviation")
```

Figure 26. Seasonal Component of Real Time Series

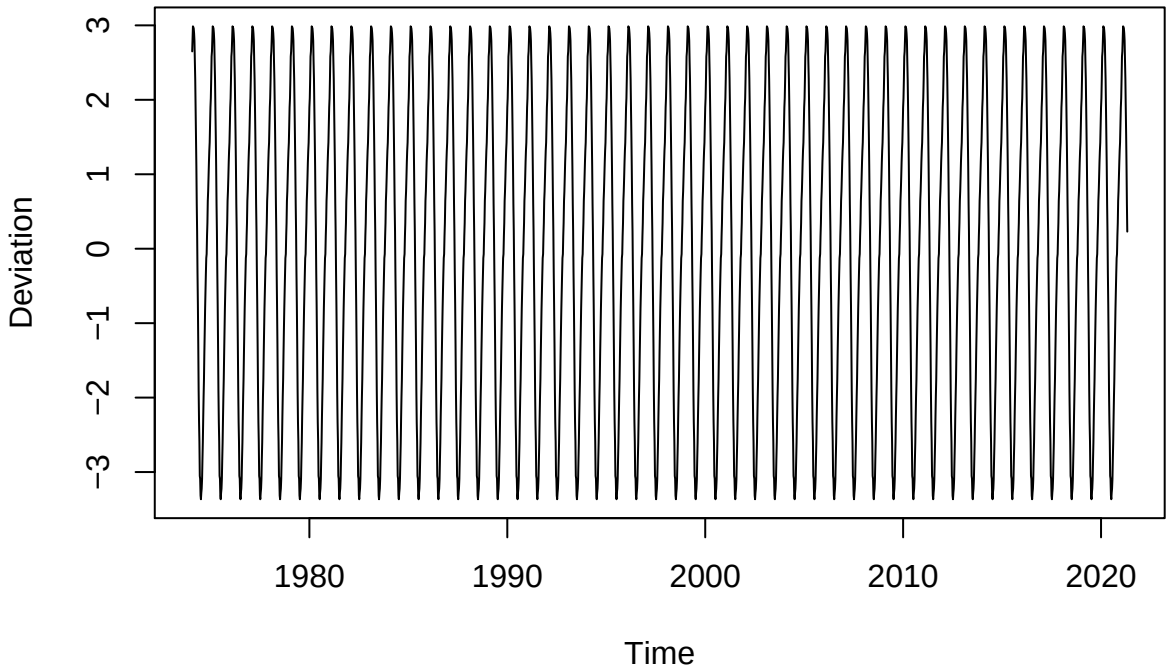


Figure 26 demonstrates that there is no variation in the seasonal component of the actual data, unlike that which we observed in the first data set. This would make a logarithm transformation of this data unnecessary as the data is already stationary in the variance. Next, the increasing mean must still be addressed, which can be done by differencing the values:

```
ts1.d1 <- diff(ts1)
ts1.d2 <- diff(ts1.d1)

par(mfrow = c(1, 2), cex = 0.6)
plot(ts1.d1, xlab = "Year", ylab = "First Differenced Value",
     main = "Figure 27. First Differenced Real Time Series", type = "l")
abline(h = mean(d1), lty = 2, col = "red")
plot(ts1.d2, xlab = "Year", ylab = "Second Differenced Value",
     main = "Figure 28. Second Differenced Real Time Series", type = "l")
abline(h = mean(d2), lty = 2, col = "red")
```

Figure 27. First Differenced Real Time Series

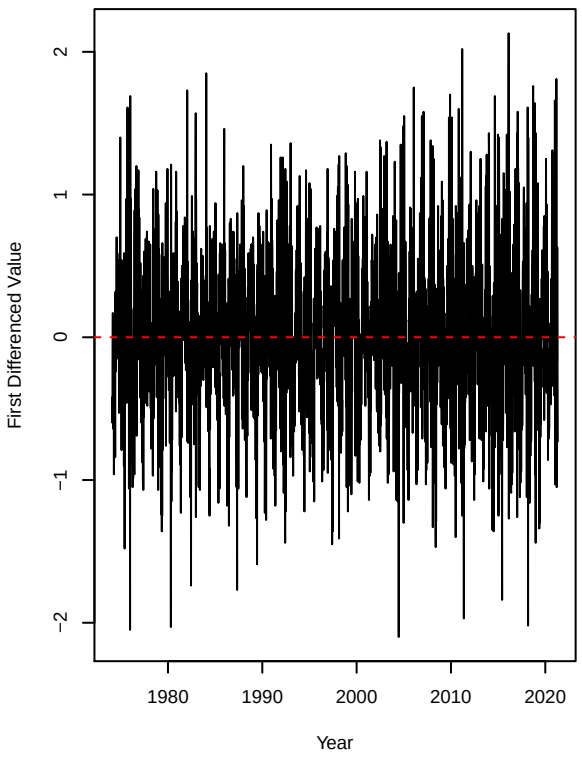
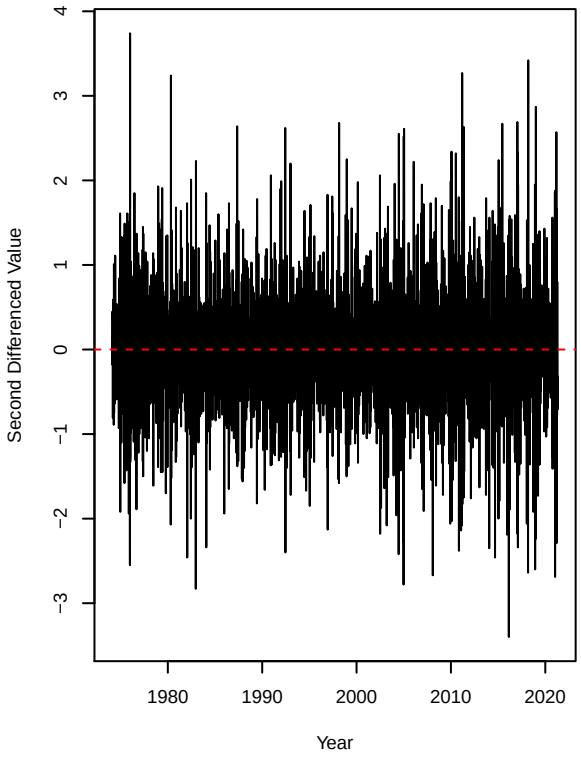


Figure 28. Second Differenced Real Time Series



Figures 27 and **28** display the time series with first and second order differencing respectively. Similar to the original `co2` series, we can see that though the differencing has removed the main increasing trend, there is still a large degree of fluctuation about the mean. Compared to the original series, the increased granularity of data at the weekly level accentuates this fluctuation. Nevertheless, just like with the original `co2` series, a second-order differencing operation not only removes the increasing trend but the data also becomes completely stationary around the mean. Next, we can look deeper into the seasonality of the data using autocorrelation and partial autocorrelation functions:

```
par(mfrow = c(1, 2), cex = 0.6)
plot(acf(ts1.d2, plot = FALSE), main = "")
title("Figure 29. Autocorrelation Plot")
plot(pacf(ts1.d2, plot = FALSE), main = "")
title("Figure 30. Partial Autocorrelation Plot")
```

Figure 29. Autocorrelation Plot

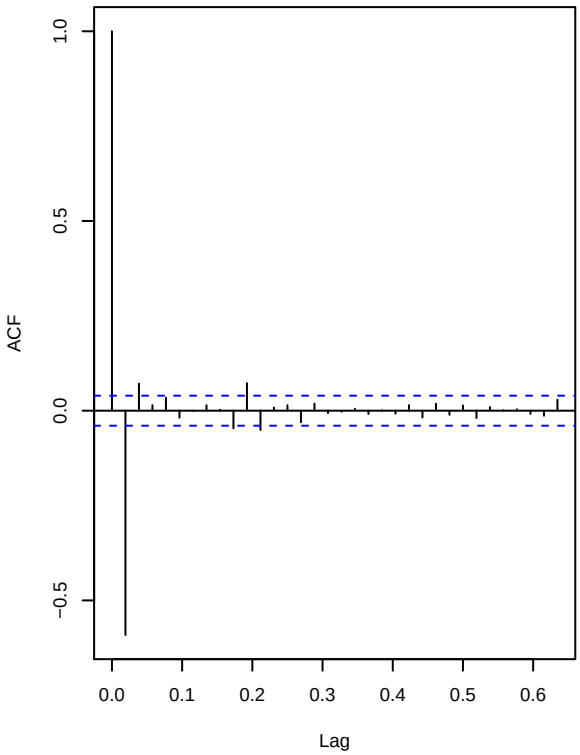
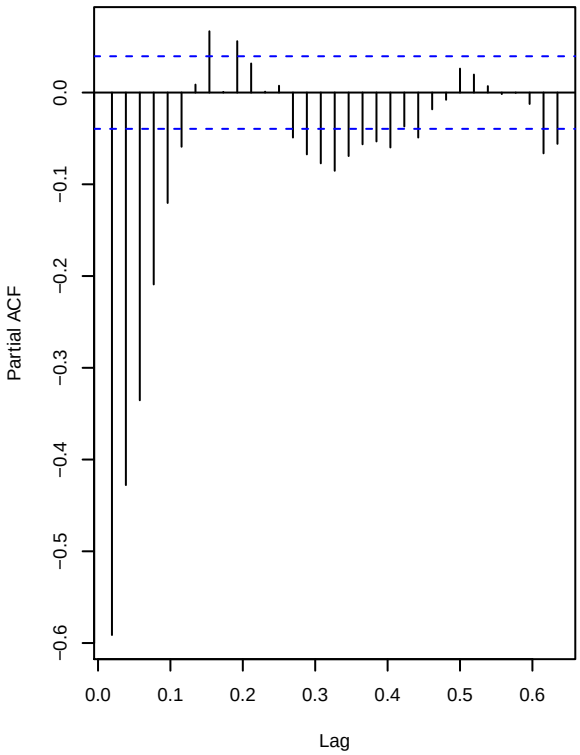


Figure 30. Partial Autocorrelation Plot

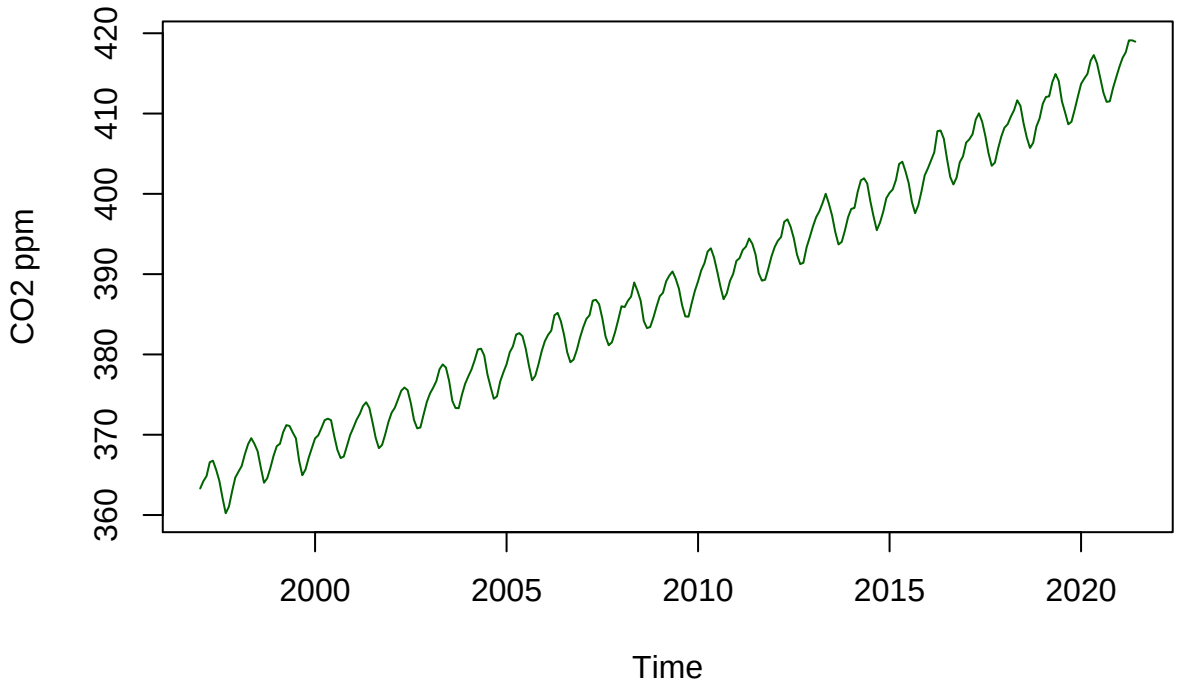


Figures 29 and **30** show the ACF and PACF of the observed data after second differencing. Unlike the ACF for the `co2` data, the ACF for this dataset shows a substantial, and highly significant negative lag at $k = 1$. Thereafter the lags drastically decrease, and it appears only the 3rd and 8th lags are weakly significant. Like with the `co2` series, the PACF has an oscillatory pattern that appears to be decreasing in magnitude over time. These all indicate a seasonal element may be present with the data.

For a more direct comparison with the `co2` dataset and the previous predicted values, aggregation can be performed to take the weekly (or approximately weekly) format into a monthly one:

```
fmt <- "%Y-%m-%d"
w3 <- aggregate(w2["ppm"], list(Date = as.yearmon(w2$date, fmt)), mean)
ts2 <- ts(subset(w3, w3$Date >= "Jan 1997")$ppm, start=c(1997,1), frequency=12)

plot(ts2, main="Figure 31. Aggregated (Monthly) Time Series From 1997",
     ylab=expression("CO2 ppm"), col="darkgreen")
```

Figure 31. Aggregated (Monthly) Time Series From 1997

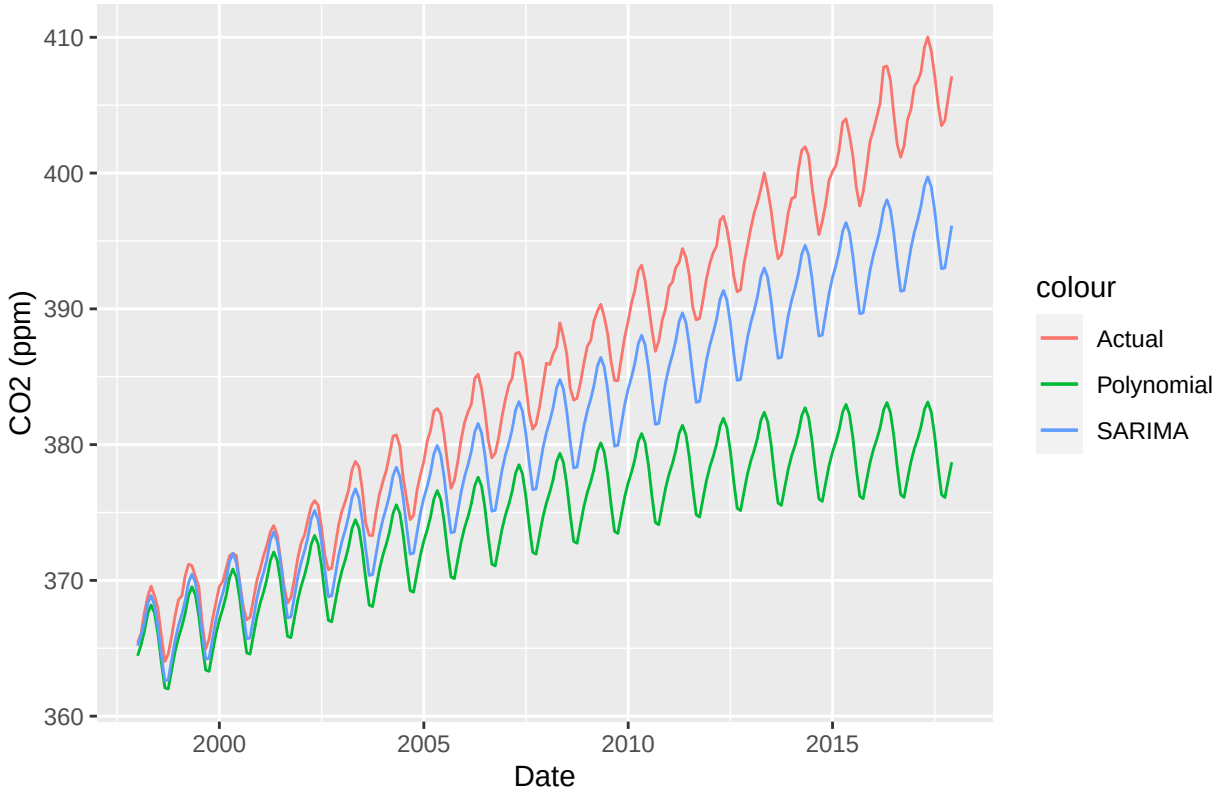


We can see from **Figure 31** that the monthly aggregated time series generally resembles the original weekly series shown in **Figure 25**.

In order to compare the previous forecasts with the actual data they may first be visualized together:

```
ts3 <- ts(subset(w3, w3$Date >= "Jan 1998" & w3$Date <= "Dec 2017")$ppm,
          start=c(1998,1), frequency=12)
p.ts <- ts(exp(predictions1$mean), start=c(1998,1), frequency=12)
p.ts2 <- ts(exp(predictions2$mean), start=c(1998,1), frequency=12)
months <- as.Date(subset(w3$Date, w3$Date >= "Jan 1998" &
                           w3$Date <= "Dec 2017"))

ggplot() +
  geom_line(data=ts3, aes(x=months, y=ts3, color="Actual")) +
  geom_line(data=p.ts, aes(x=months, y=p.ts, color="Polynomial")) +
  geom_line(data=p.ts2, aes(x=months, y=p.ts2, color="SARIMA")) +
  ggtitle("Figure 32. Predicted (Polynomial & SARIMA) vs. Actual CO2 Values") +
  labs(x="Date", y="CO2 (ppm)")
```

Figure 32. Predicted (Polynomial & SARIMA) vs. Actual CO2 Values



As we can see from **Figure 32**, our SARIMA model generally predicts carbon dioxide levels very accurately including seasonal variation, until ~2004. Thereafter, the model begins to underestimate the true CO2 levels, though it appears the cyclical variation in carbon dioxide levels remains well represented by the model. In comparison, though the polynomial can also provide a generally reasonable short-term forecast until ~2004, thereafter it begins to severely underestimate CO2 levels relative to the SARIMA model. In order to quantify the performance of each, we can use metrics such as mean absolute error (MAE) and root mean squared error (RMSE), among others:

```
poly.acc <- accuracy(p.ts, ts3)
sarima.acc <- accuracy(p.ts2, ts3)
poly.acc
```

```
##                 ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
## Test set 10.85246 13.32521 10.85246 2.757882 2.757882 0.9841837  9.904399
```

```
sarima.acc
```

```
##                 ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
## Test set 4.477818 5.330831 4.477818 1.140596 1.140596 0.9743537  3.973992
```

Here, we can see that the SARIMA model quantitatively outperforms the polynomial model, with a MAE of 4.4778 (vs. 10.8525 for the polynomial model) and an RMSE of 5.3308 compared to an RMSE of 13.3252 for the polynomial model. Between the two models, the SARIMA approximates

the real data more closely, as evidenced both visually and the accuracy metrics. Nevertheless, if we continue to extrapolate further the SARIMA and actual values will further bifurcate, meaning the gap between them will increase as will the error metrics.

Part 5 (5 points)

Split the NOAA series into training and test sets, using the final two years of observations as the test set. Fit an ARIMA model to the series following all appropriate steps, including comparison of how candidate models perform both in-sample and (psuedo-) out-of-sample. Generate predictions for when atmospheric CO2 is expected to reach 450 parts per million, considering the prediction intervals as well as the point estimate. Generate a prediction for atmospheric CO2 levels in the year 2100. How confident are you that these will be accurate predictions?

In order to allow for maximum iteration in our SARIMA models, we will use the aggregated monthly dataset rather than weekly CO2 readings.

```
train <- subset(w3, w3$Date <= "Jun 2019")
test <- subset(w3, w3$Date > "Jun 2019")

train.ts <- ts(train$ppm, start=c(1974, 5), frequency=12)
test.ts <- ts(test$ppm, start=c(2019, 6), frequency=12)
```

```
d_non_seas <- diff(train.ts, lag = 12)
d_seas <- train.ts - d_non_seas

par(mfrow = c(2, 2), cex = 0.6)
plot(acf(d_non_seas, plot = FALSE), main = "")
title("Figure 33. Non-Seasonal ACF")
plot(pacf(d_non_seas, plot = FALSE), main = "")
title("Figure 34. Non-Seasonal PACF")
plot(acf(d_seas, plot = FALSE), main = "")
title("Figure 35. Seasonal ACF")
plot(pacf(d_seas, plot = FALSE), main = "")
title("Figure 36. Seasonal PACF")
```

Figure 33. Non−Seasonal ACF

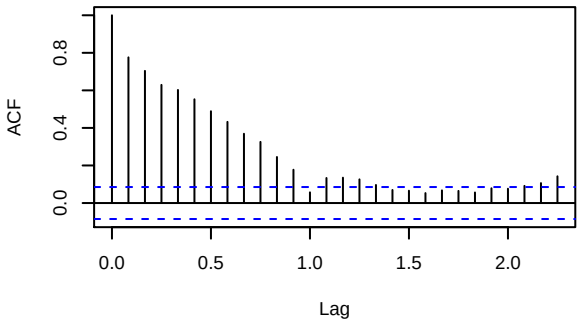


Figure 34. Non−Seasonal PACF

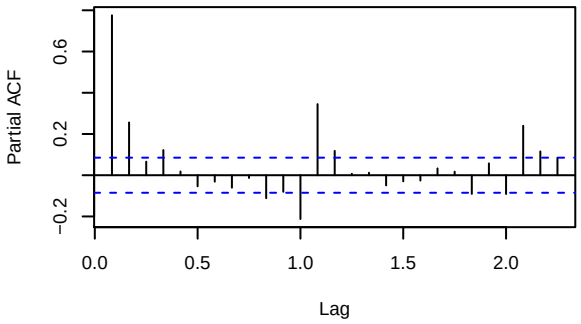


Figure 35. Seasonal ACF

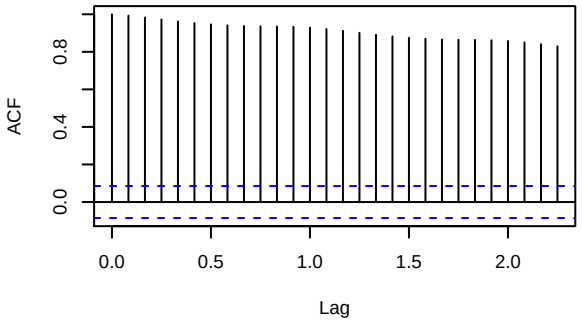
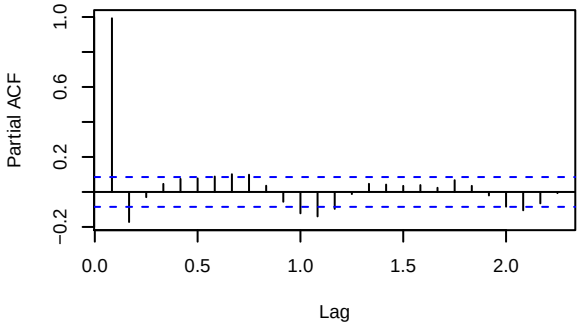


Figure 36. Seasonal PACF



Based upon the above non-seasonal (**Figures 33** and **34**) and seasonal (**Figures 35** and **36**) ACF and PACF plots, we can see that there is an element of seasonality in the data, with the non-seasonal ACF (**Figure 33**) showing a seasonal pattern. Similarly, the non-seasonal PACF (**Figure 34**) shows a significant spike at the 12th lag, with a slightly smaller (yet still significant) lag at the 24th. We can then observe that after differencing and accounting for seasonality, the seasonal PACF (**Figure 36**) has a significant spike at the 1st, 2nd, 11th and 12th lags, This indicates an AR(1) or AR(2) model should be chosen. Moreover, by looking at **Figure 35**, though every lag appears to be highly significant, the autocorrelations greater than lag 2 might be caused by propagation of lag-1 autocorrelation.

Based upon our EDA, it appears that we can attempt to build an autoregressive model of at least order 1, with a differencing term of 1. Nevertheless, due to the high number of significant lags in **Figure 35** perhaps a moving average parameter should be incorporated into the model. In order to find the best possible model, we can create a nested loop that iterates through various potential parameters for **p**, **d**, **q**, and their seasonal equivalents. We can then look at the Akaike information criterion (AIC) values for each model, and select the model with the minimum AIC for further use.

```
params <- list()
aics <- list()

for (p in 0:2) {
  for (q in 0:2) {
    for (d in 1:2) {
      for (P in 0:2) {
        for (Q in 0:2) {
```

```
          for (D in 1:2) {
            model <- try(arima(train.ts, order=c(p,q,d),
                               seasonal=list(order=c(P,D,Q),
                                             period=12), method="ML"),
                         silent=T)
            params <- append(params, paste(p,d,q,P,D,Q))
            default <- NULL
            aic <- try(default <- model$aic, silent=T)
            aics <- append(aics, aic)
              }
            }
          }
        }
      }
    }
```

Next we can select the model with the lowest AIC:

```
aics2 <- as.numeric(unlist(aics))
min <- ifelse(!all(is.na(aics2)), min(aics2, na.rm=T), NA)
index <- which(aics2==min)
```

Going by AIC, model 147 produced the best fitting model, with an AIC of 342.2747. The model had ARIMA parameters **p=1**, **d=1**, **q=1** and seasonal parameters of **P=0**, **D=1** and **Q=1** and a seasonality of 12. We can build this model out specifically to create predictions:

```
f.md <- arima(train.ts, order=c(1,1,1), seasonal=list(order=c(0,1,1), period=12))

plot(train.ts, xlab = "Year", ylab = "Value",
     main = "Figure 37. Fitted Best Performing SARIMA Model", type = "l")
lines(train.ts + f.md$resid, col = "orange", cex = 0.5)
legend('bottomright',
       legend = c("Original", "SARIMA Model"),
       col=c("black","orange"), lty=1)
```

Figure 37. Fitted Best Performing SARIMA Model

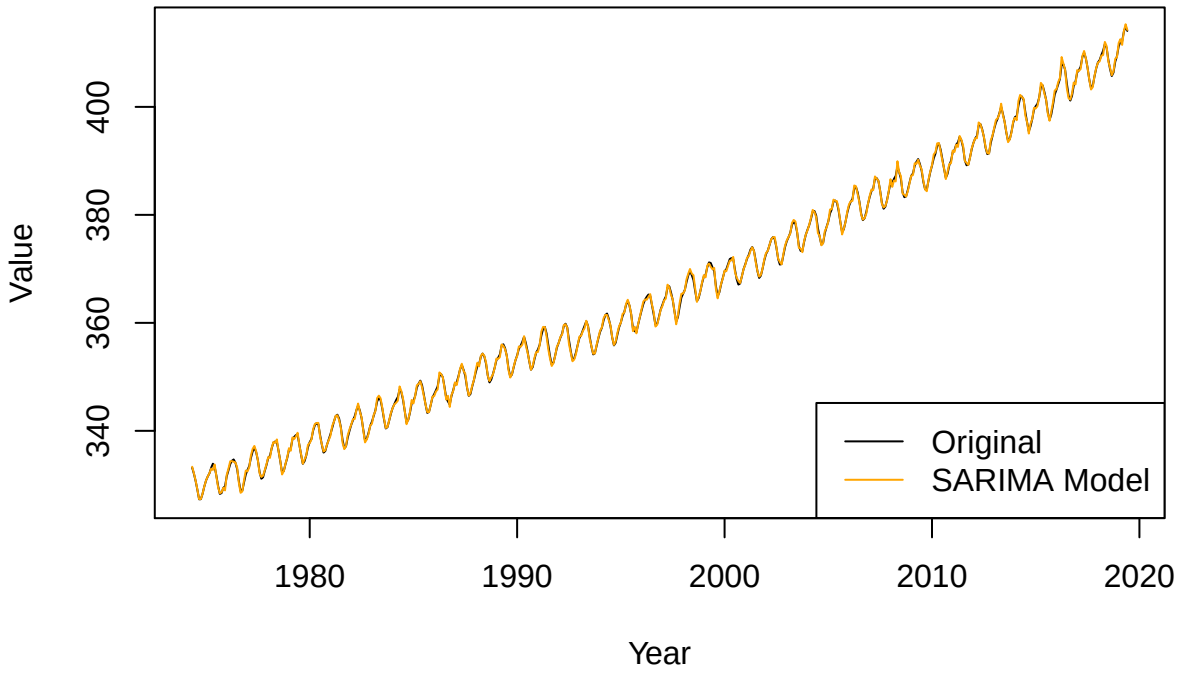


Figure 37 shows the best fit SARIMA model on the dataset, which matches the original data almost identically.

We can also look at the residuals for this model:

```
par(mfrow = c(1, 2), cex = 0.6)
plot(f.md$residuals,
     xlab = "Time", ylab = "Value", xaxt = "n",
     main = "Figure 38. Best Performing SARIMA Model Residuals", type = "l")
axis(1, at=c(12,132,252,372), labels=c(1960, 1970, 1980, 1990))
abline(h = mean(f.md$resid), col = "red")
plot(hist(f.md$residuals, plot = FALSE), xlab = "Residual Value",
     main="Figure 39. Histogram of Residuals")
```

Figure 38. Best Performing SARIMA Model Residual

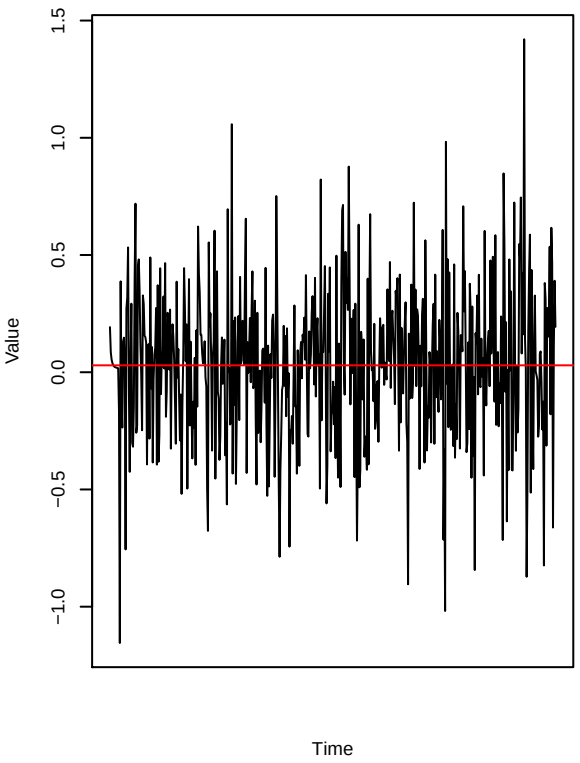
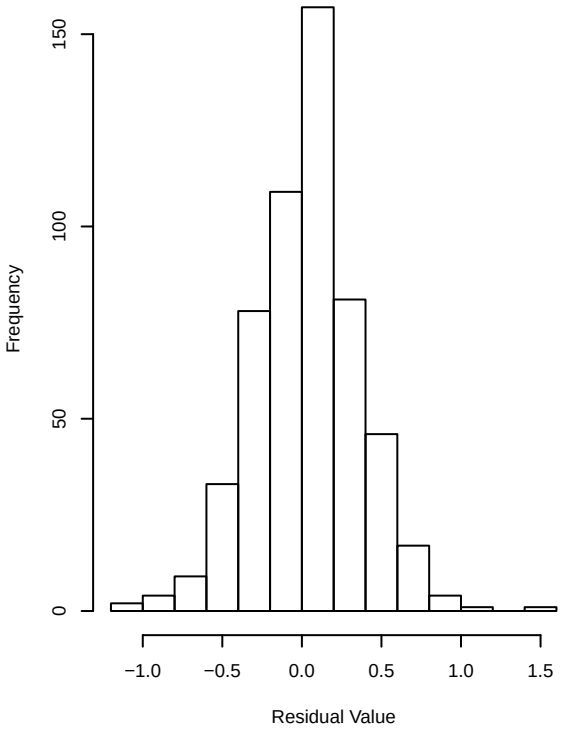


Figure 39. Histogram of Residuals



Figures 38 and **39** show the residuals of the model over time along with their distribution. It can be seen that there is no obvious pattern remaining in **Figure 38**, suggesting the model accounts for the majority of the components in the underlying data. Likewise, the distribution appears normal, also suggesting there is no remaining trend in the data not accounted for by the applied transformations and modeling.

Finally, we can look at the in-sample model performance metrics:

```
fitteds <- train.ts + f.md$residuals
isa <- accuracy(fitteds, train.ts)
isa
```

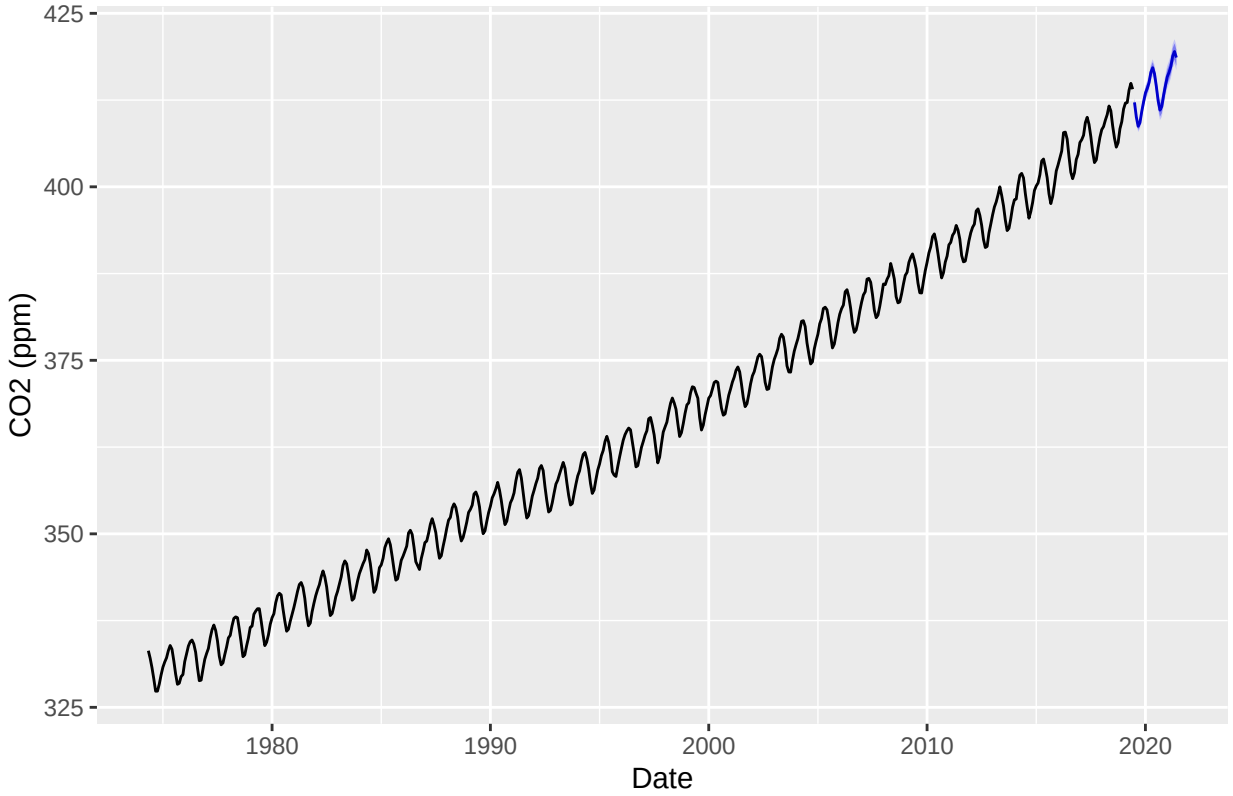
```
##                      ME    RMSE       MAE          MPE       MAPE        ACF1
## Test set -0.02942438 0.32666 0.2508338 -0.007799298 0.06834831 -0.01355779
##          Theil's U
## Test set 0.2543037
```

Based upon **Figure 37** our best performing model has a fit that is almost identical to the training set values. This is evidenced by the fact that we cannot easily visually differentiate the fitted values from the real values, and there are only glimpses where there is not an exact overlap. This may be of some concern if attempting to use it for prediction as overfitting might have occurred. In the performance metrics shown above, we can see that the in-sample performance achieves a MAE of 0.2508 and a RMSE of 0.3267 which reinforce our visual findings.

The next step is to use this model to forecast for the next 2 years, which is the period for which our testing dataset allows for comparison. We can visualize the predictions alone and together with the testing dataset:

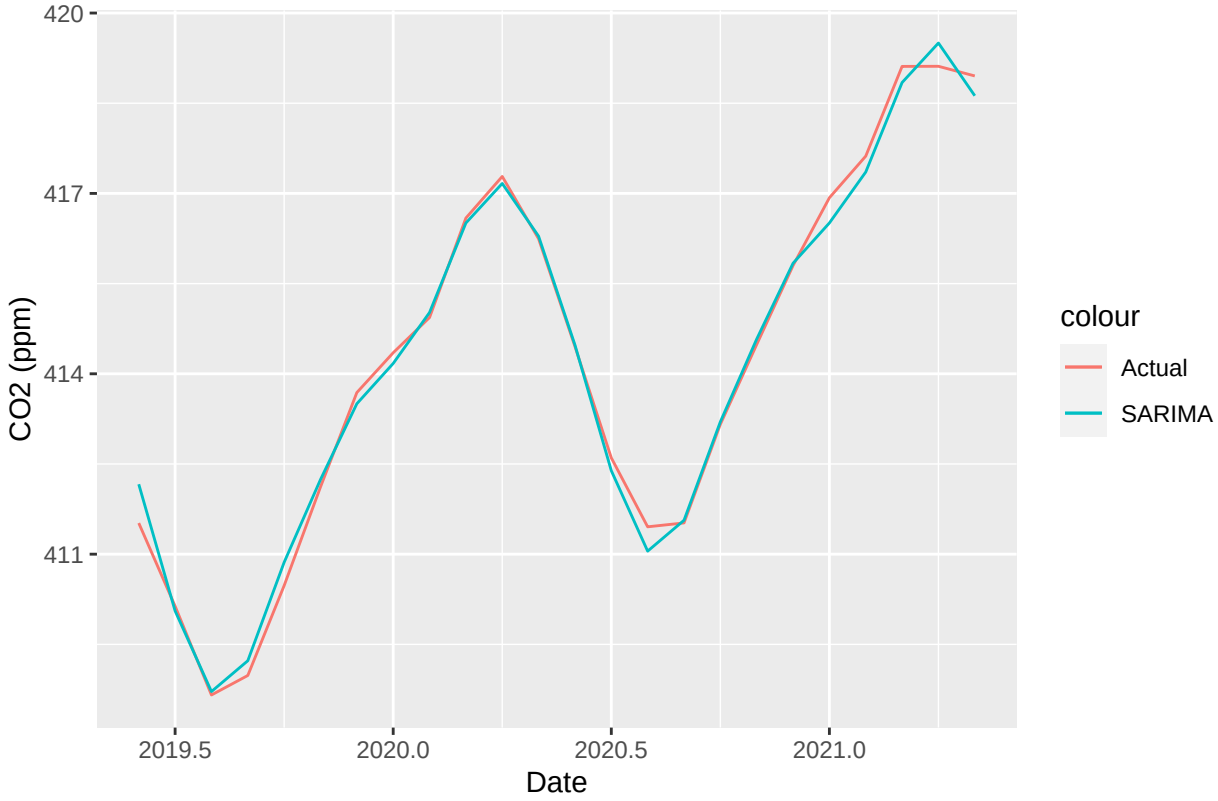
```
f1 <- forecast(f.md, h=24)
f.ts <- ts(f1$mean, start=c(2019,7), frequency=12)

autoplot(f1) +
  ggtitle("Figure 40. Forecasts from Best Fitting ARIMA Model") +
  labs(y="CO2 (ppm)", x="Date")
```

Figure 40. Forecasts from Best Fitting ARIMA Model




```
ggplot() +
  geom_line(data=test.ts, aes(x=time(test.ts), y=test.ts, color="Actual")) +
  geom_line(data=f.ts, aes(x=time(test.ts), y=f.ts, color="SARIMA")) +
  labs(x="Date", y="CO2 (ppm)") +
  ggtitle("Figure 41. Comparison of Predicted Vs. Actual Values")
```

Figure 41. Comparison of Predicted Vs. Actual Values



Based upon **Figure 40** we can see that the 2 year prediction of the SARIMA (1,1,1)(0,1,1)[12] model looks almost like a continuation of the original line. Moreover we can see that the confidence intervals are extremely narrow to the point of being almost indistinguishable from the mean values. Again, this might be due to the overfitting of the model or the fact that it is a relatively short-term forecast. Looking at **Figure 41**, we can see that the predictions for CO2 almost completely overlap the values of the original data.

We can also calculate error metrics for our predictions compared to the test values:

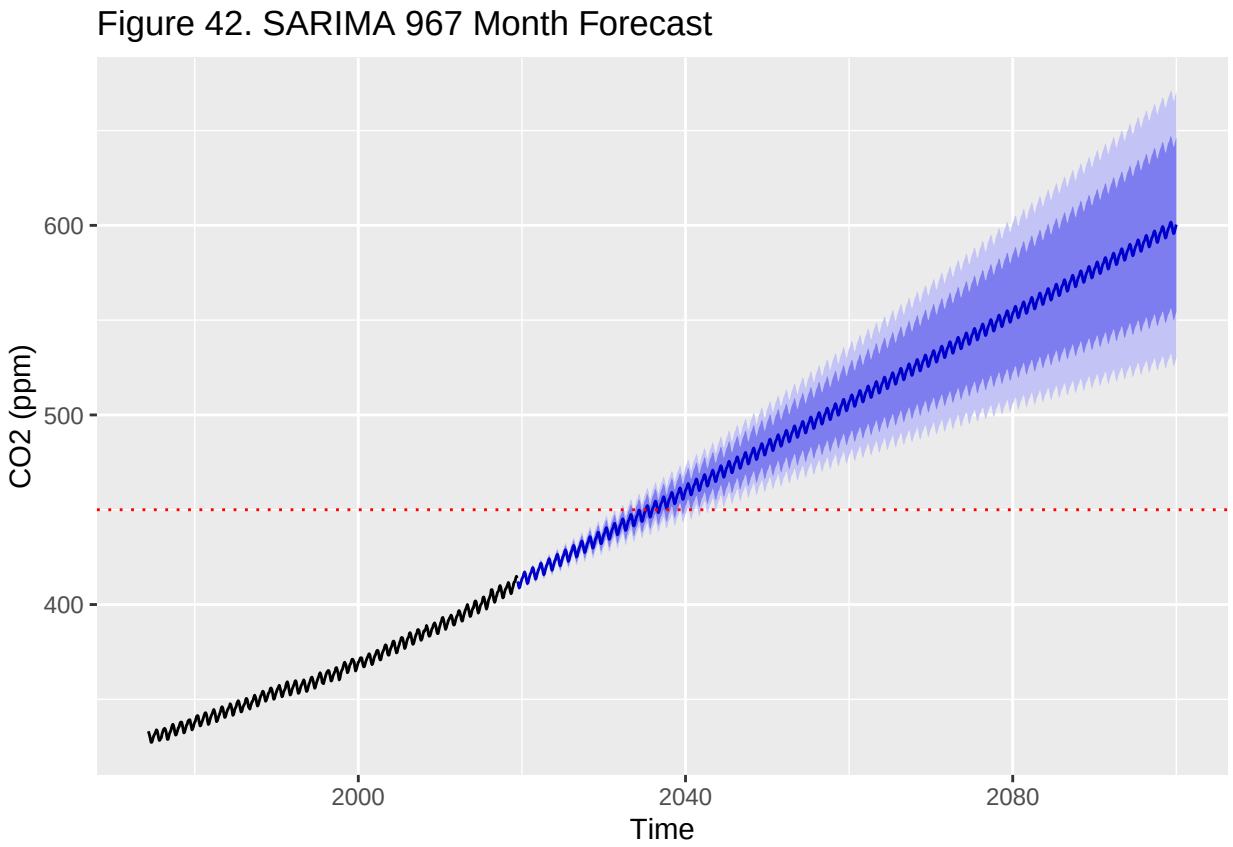
```
accf1 <- accuracy(f.ts, test.ts)
accf1
```

```
##                    ME     RMSE      MAE         MPE      MAPE      ACF1 Theil's U
## Test set 0.3233109 1.253458 1.152814 0.07729329 0.2784139 0.6536864 0.9856333
```

In regards to out-of-sample performance we can see that the MAE is 1.1528 while the RMSE is 1.2535. These results are very low and together with the visual inspection conducted previously, we can see that our predictions are an almost exact match to the testing data. Though we might be inclined to be optimistic - we should remember that this is a relatively short-term forecast (24 months) and it is likely that our model has overfit the training data. This would make longer term forecasts be taken with more caution.

In order to find the time when the CO2 value will be 450 ppm (and to see what the CO2 concentration might be in 2100), we can extend our prediction further into the future by approximately 1000 months, as our previous forecast until mid-2021 stops prior to reaching 420 ppm:

```
f2 <- forecast(f.md, h=967)
f2df <- data.frame(f2)

autoplot(f2) +
  geom_hline(yintercept = 450, linetype="dotted", color='red') +
  ggtitle("Figure 42. SARIMA 967 Month Forecast") +
  labs(x="Time", y="CO2 (ppm)")
```



According to this model and depicted by the dotted red line in **Figure 42**, the CO2 concentration is expected to reach or exceed 450 ppm in March 2035. The 95% confidence interval for this reading is between 440.8005 and 459.3223 for March 2035.

Figure 42 shows a 200 month ahead forecast from June 2019. As we can see here, the trend the model predicts is approximately linear and the CO2 levels are expected to breach 450 ppm at the crest of a cycle in approximately ~March 2035 by the point estimate. Nevertheless, we can also see that the uncertainty of predictions increases the further we look into the future, as evidenced by the widening 95% confidence intervals. When considering this, 450 ppm is first reached by the upper 95% confidence interval in approximately ~March 2032. As such we cannot be entirely certain about our forecasts which may be much higher or much lower than what we have predicted. Thus, while we may predict that in January 2100 the point forecast of CO2 levels will be 600.3921, the 95% confidence interval is extremely wide and between 530.098 and 670.6861. Consequently this indicates that our model should not be used to make predictions so far ahead into the future, and any predictions that are made in this capacity should be considered unreliable.