# Statistical Methods for Discrete Response, Time Series, and Panel Data (W271): Lab 2

Due Sunday 21st March 2021 11:59pm

Shishir Agarwal, Jenny Pyon, Chris Weyandt

**Formatting**

```
library(knitr)
opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=TRUE)
```

**Load Packages**

```
library(readr)
library(lubridate)
library(forecast)
library(fable)
library(fabletools)
library(feasts)
library(tsibble)
library(fpp2)
library(fpp3)
library(astsa)
library(dplyr)
library(urca)
library(Hmisc)
library(seasonal)
library(car)
library(ggpubr)
# load the workspace so knitr can skip re-running models
load("co2.RData")
```

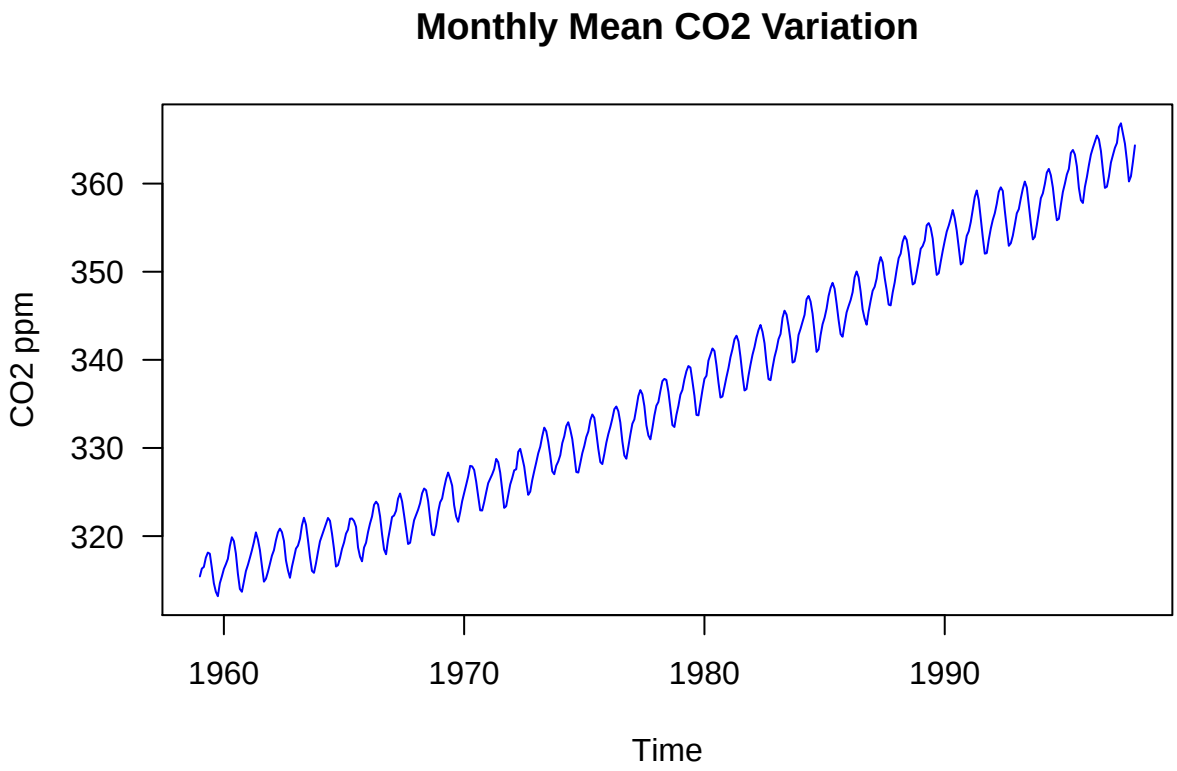
## The Keeling Curve

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.

'co2' is a times series data set with 468 observations of carbon dioxide concentration from 1959 to 1998.

```
str(co2)
```

```
##  Time-Series [1:468] from 1959 to 1998: 315 316 316 318 318 ...
```

Monthly data is provided in the data set, with complete observations in the first and last years.

```
head(co2, 12)
```

```
##         Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep    Oct
## 1959 315.42 316.31 316.50 317.56 318.13 318.00 316.39 314.65 313.68 313.18
##         Nov    Dec
## 1959 314.66 315.43
```

```
tail(co2, 12)
```

```
##         Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep    Oct
## 1997 363.23 364.06 364.61 366.40 366.84 365.68 364.52 362.57 360.24 360.83
##         Nov    Dec
## 1997 362.49 364.34
```

Ensure that the frequency of the dataset is defined correctly as monthly. The data appears to show a clear seasonal cycling component, so we will later use the months as indicator variables in our analysis.

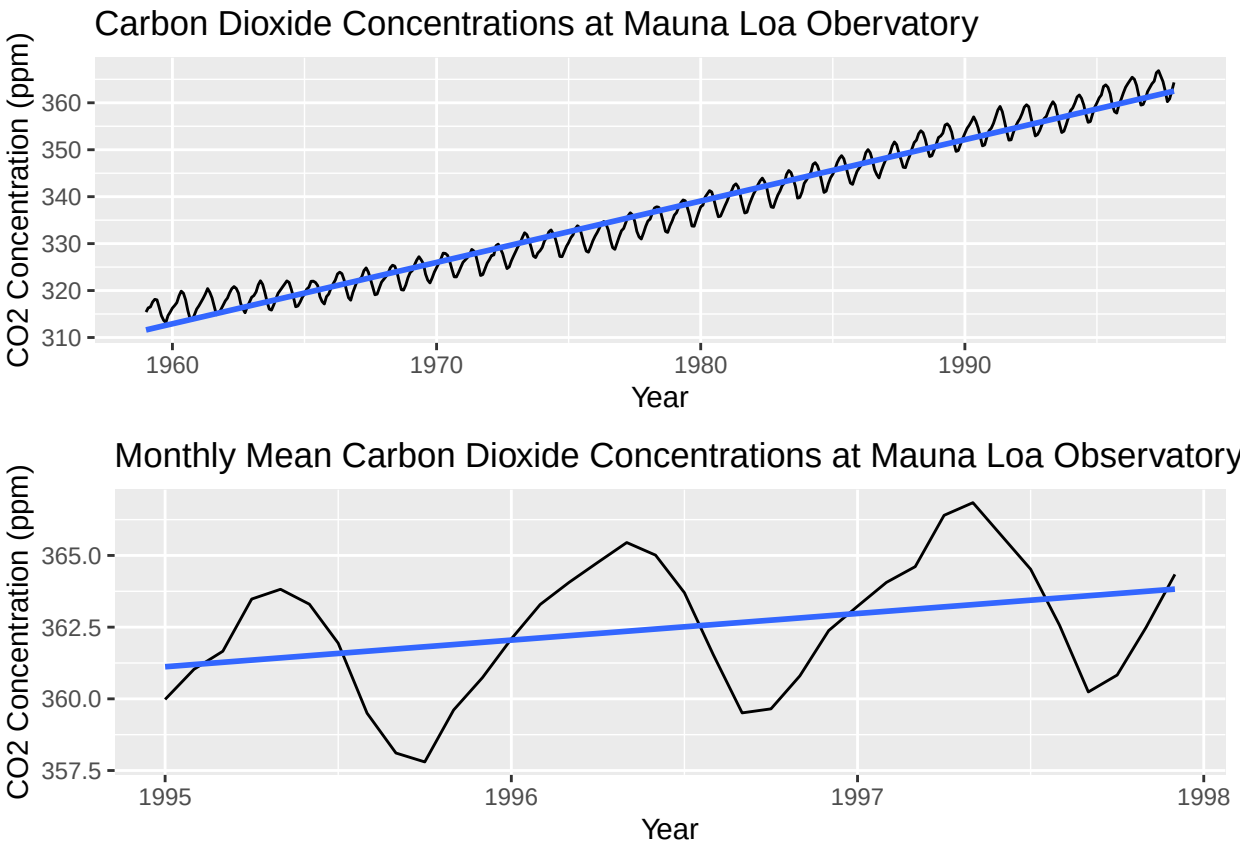
```
head(cycle(co2),36)
```

```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 1959   1   2   3   4   5   6   7   8   9  10  11  12
## 1960   1   2   3   4   5   6   7   8   9  10  11  12
## 1961   1   2   3   4   5   6   7   8   9  10  11  12
```

A plot of the timeseries data shows increasing trend and seasonal trend in the carbon dioxide concentration measured at Mauna Loa Observatory. The seasonal component appears to be very consistent, and the trend component is nearly linear. A simple linear regression is overlaid on the data to emphasize the linear trend component (top). A subset of the data is plotted to emphasize the seasonal component (bottom). For each 12-month period, there appears to be an upward trend for six months and a downward trend for approximately six months.
The concentration rises at a slower rate than it falls, which is a common pattern indicating a

shift in the rate of co2 emissions versus the rate of co2 sequestration. One possible explanation is the annual growth patterns of plants that pull co2 from the atmosphere (sequestration), which occurs much more during warmer weather corresponding with summer in the Northern hemisphere. Likewise, far more fossil fuels are burned (emissions) in the colder months when buildings need to be heated and vegetation has begun to die and decay. If this data were modeled with explanatory variables such as global fossil fuel consumption, global temperature and global rainfall it might provide useful insight into this question.

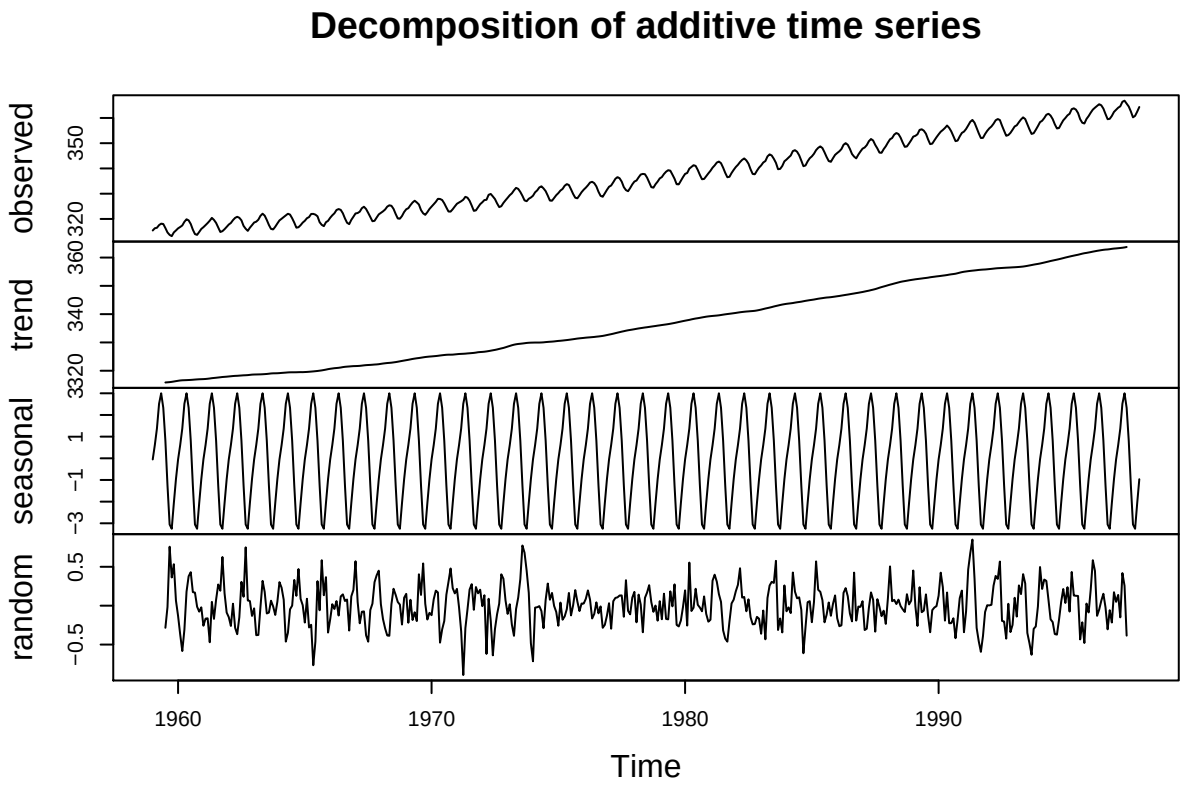
```
plot1 <- autoplot(co2) +
  ggtitle('Carbon Dioxide Concentrations at Mauna Loa Obervatory') +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)') +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE)

plot2 <- autoplot(window(co2, start = 1995)) +
  ggtitle('Monthly Mean Carbon Dioxide Concentrations at Mauna Loa Observatory') +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)') +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE)

ggarrange(plot1,plot2,nrow=2,ncol=1)
```




Performing a decomposition of the timeseries demonstrates that the trend component is nearly

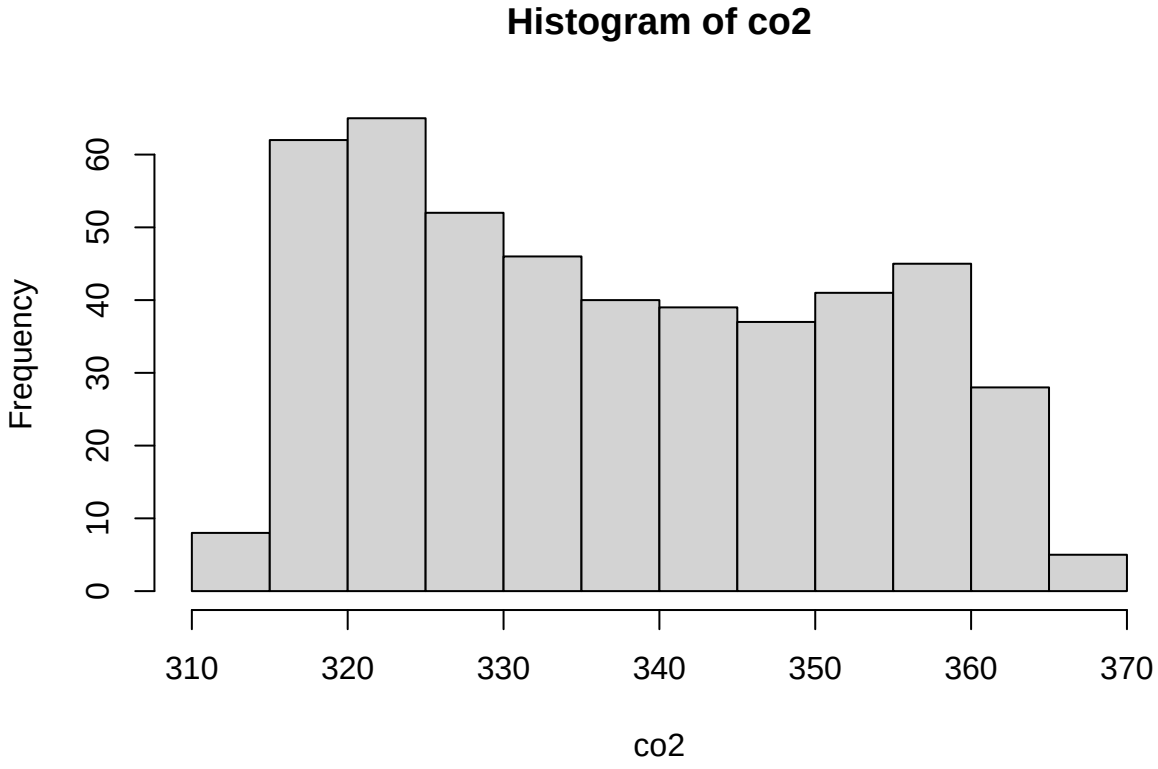
linear and the amplitude of the seasonal component is very consistent. The random component appears to be a stationary series of white noise.

```
plot(decompose(co2))
```



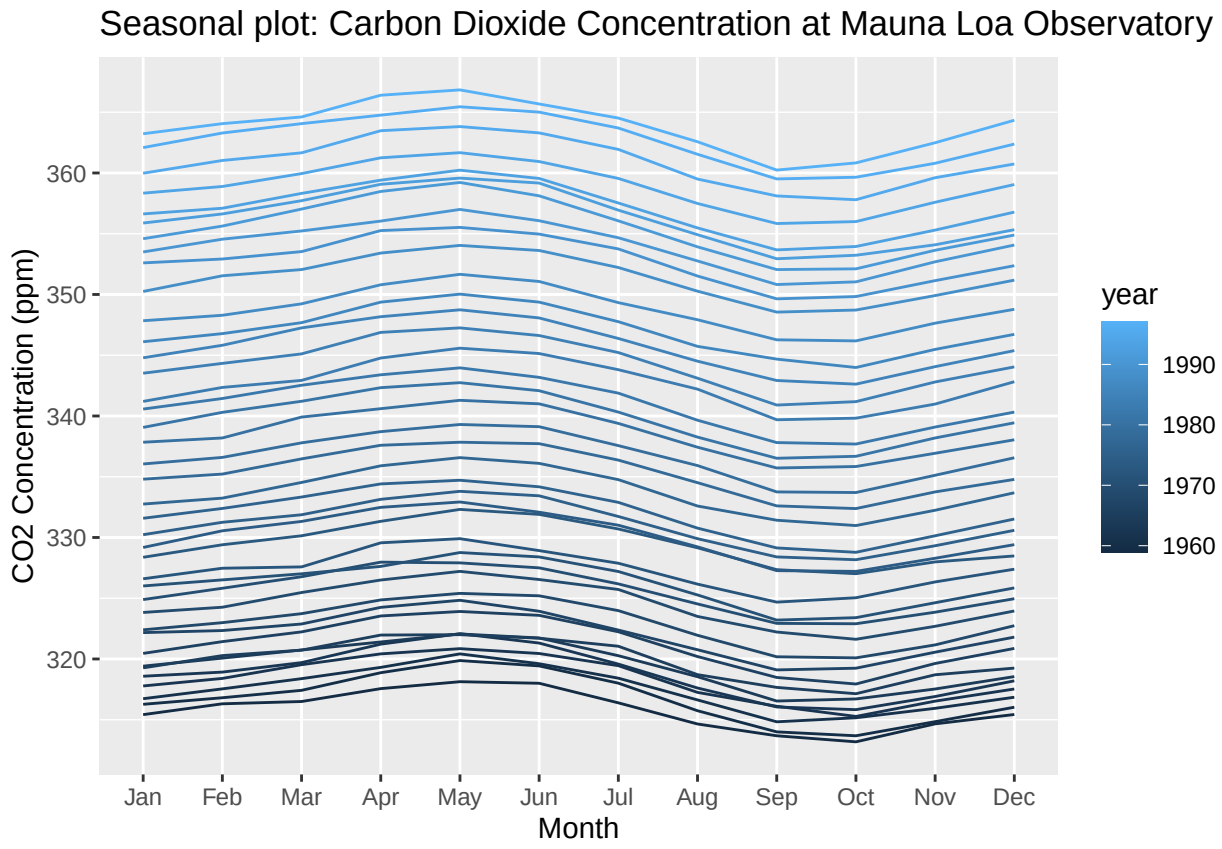
A histogram of the CO2 concentrations is shown below. If the trend component were perfectly linear, the expectation is that the distribution would be uniform. Actual results demonstrate that the trend component is not perfectly linear as there is higher number of samples at lower concentrations, suggesting that a polynomial regression fit will be required to accurately model this series.

```
hist(co2)
```



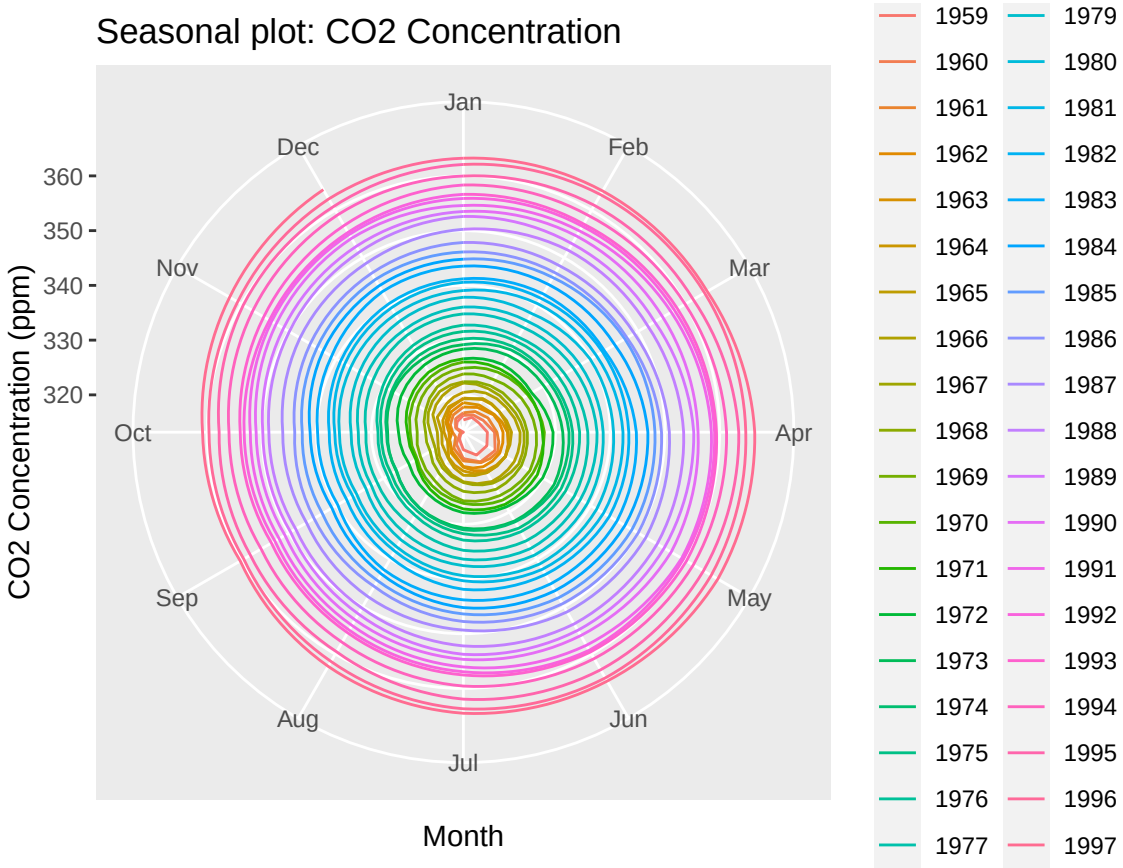
As shown in the seasonal plot below, the carbon dioxide concentration measured at Mauna Loa Observatory has an increasing trend and there is a seasonal component. The linear rate trend is demonstrated by the increasing intercept for each consecutive year, and the uniformity can be seen in the nearly constant difference between each year. There are a few points where the rate of increase seems to change, demonstrated by wider gaps between years (i.e. ~1974 and ~1988). The variance in this difference represents the white noise component of the timeseries.

```
ggseasonplot(co2, year.labels=FALSE, continuous=TRUE) +
  ylab('CO2 Concentration (ppm)') +
  ggtitle('Seasonal plot: Carbon Dioxide Concentration at Mauna Loa Observatory')
```



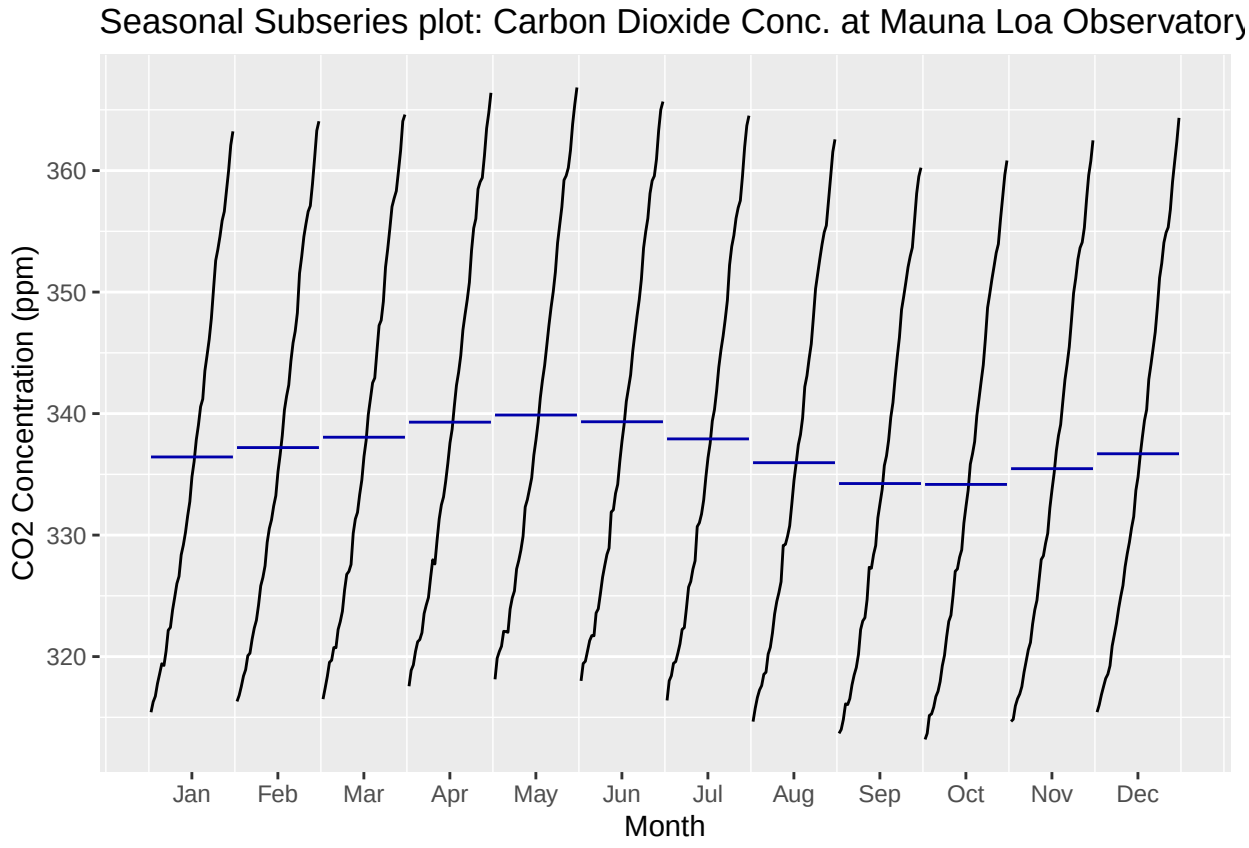
A polar seasonal plot is another great way to confirm the near-linearity of the timeseries trend component and the uniformity of the seasonal component. Note that the line never crosses over itself, which demonstrates that the annual increase is greater than the amplitude of the seasonal variation.

```
ggseasonplot(co2, polar=TRUE) +
  ylab('CO2 Concentration (ppm)') +
  ggtitle('Seasonal plot: CO2 Concentration')
```



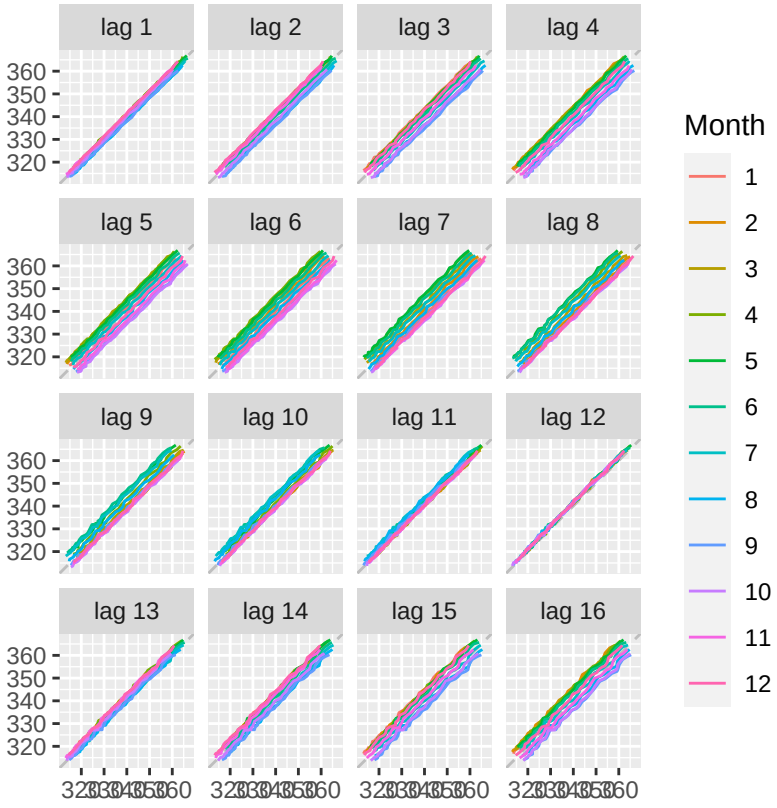
The seasonal subseries plot shows the mean carbon dioxide concentration is highest in May and lowest in October. It is more informative than a boxplot for timeseries data because it includes the linear trend compoenent of the data that is masked by the boxplot aggregation.

```
ggsubseriesplot(co2) +
  ylab('CO2 Concentration (ppm)') +
  ggtitle('Seasonal Subseries plot: Carbon Dioxide Conc. at Mauna Loa Observatory')
```



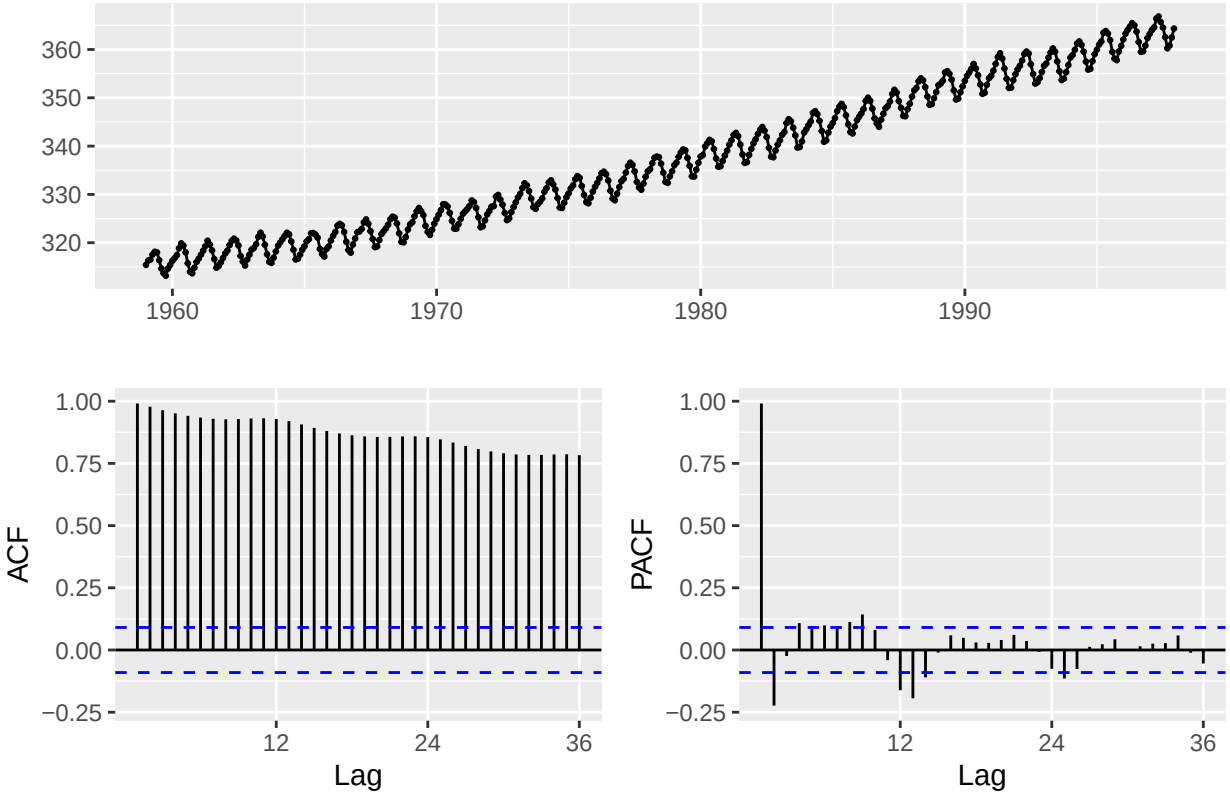
There is strong monthly seasonality in the data as shown in lag 12. Lags between lag 3 and lag 9 show negative relationship because highs are plotted against lows. It is also clear that the lag plots become repetitive after the 12th lag, with lag1 being nearly identical to lag13, lag2 being nearly identical to lag14, and so on.

```
gglagplot(co2)
```



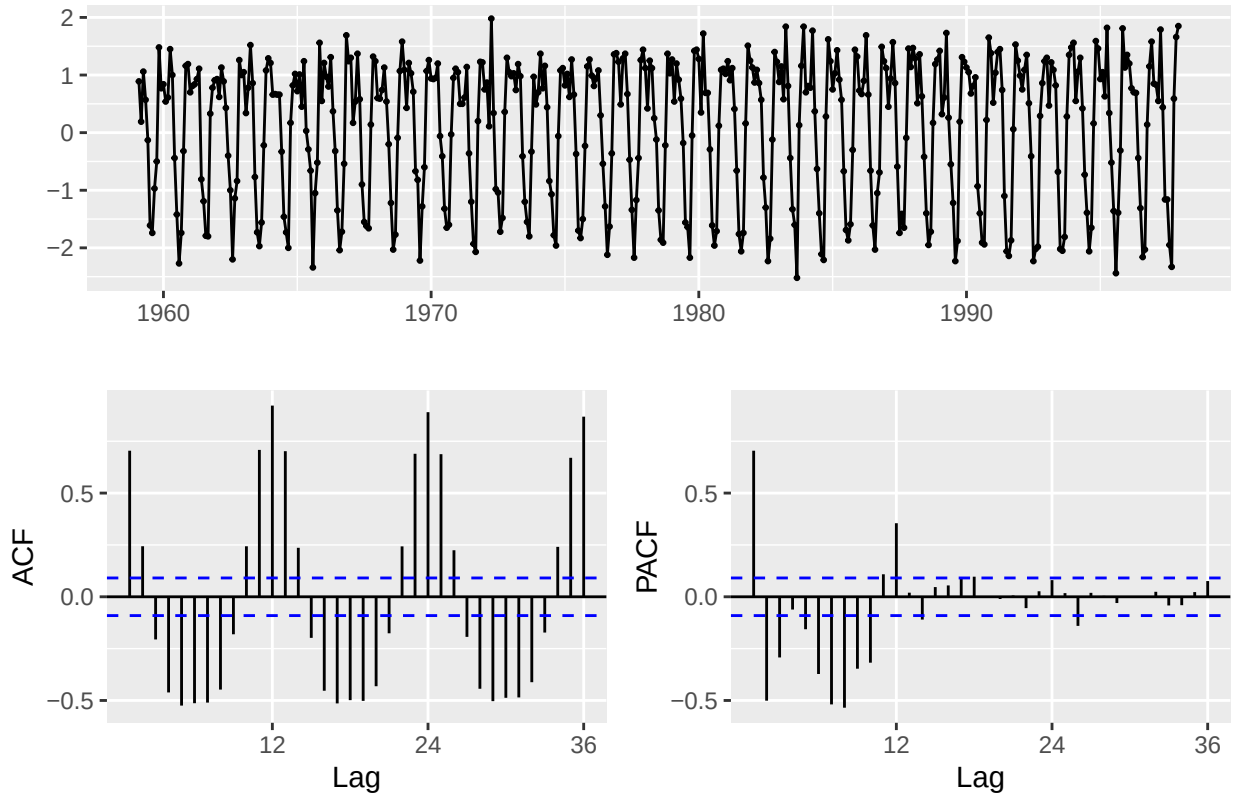
The autocorrelation plot is extremely persistent, with very gradual decay. Even after 24 lags the ACF is well above the significance level, clearly indicating that the series is non-stationary.

```
co2 %>% ggtsdisplay()
```



The trend can be removed by differencing the series at one lag. The resulting plot shows the seasonal and white noise components that remain in the series. The autocorrelation plot clearly shows the seasonal component is significant and therefor this differenced series is still non-stationary.

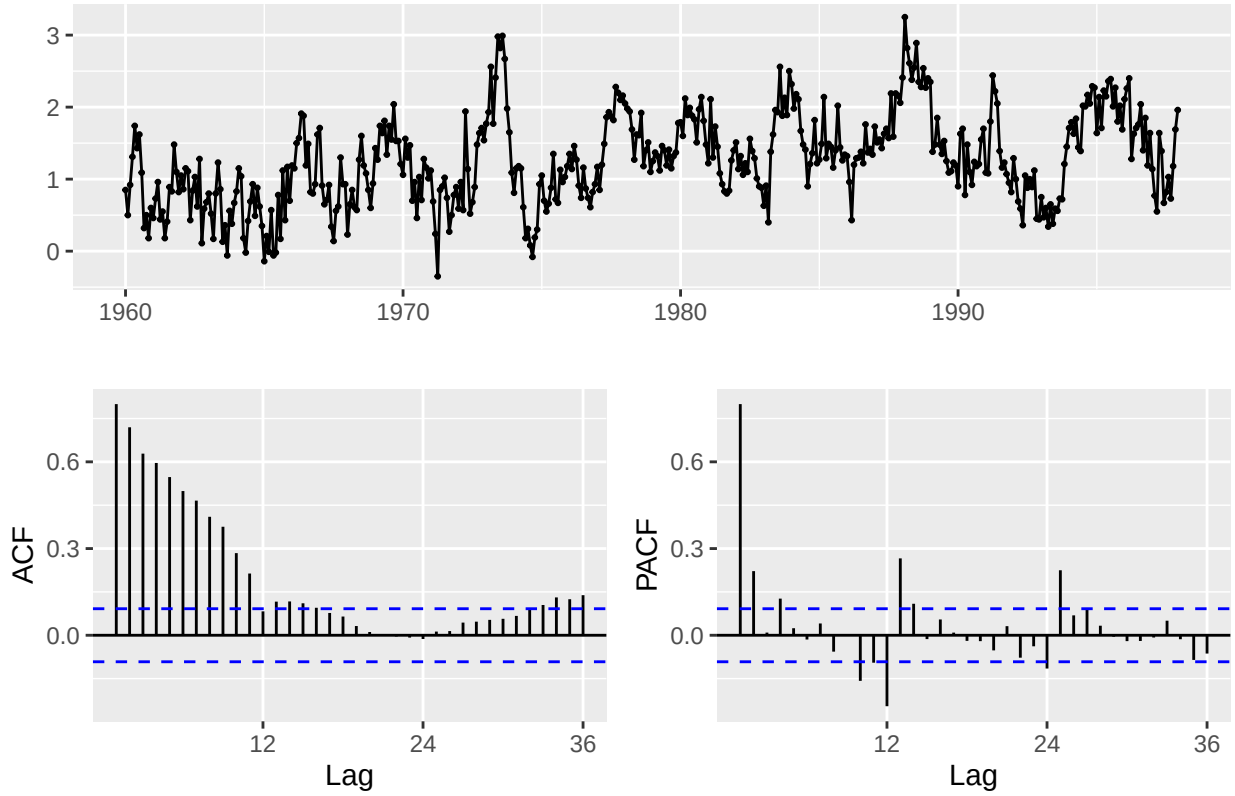
```
co2 %>% diff() %>% ggtsdisplay()
```




The above ACF shows that the seasonal component is quite consistent even out to 36 lags.

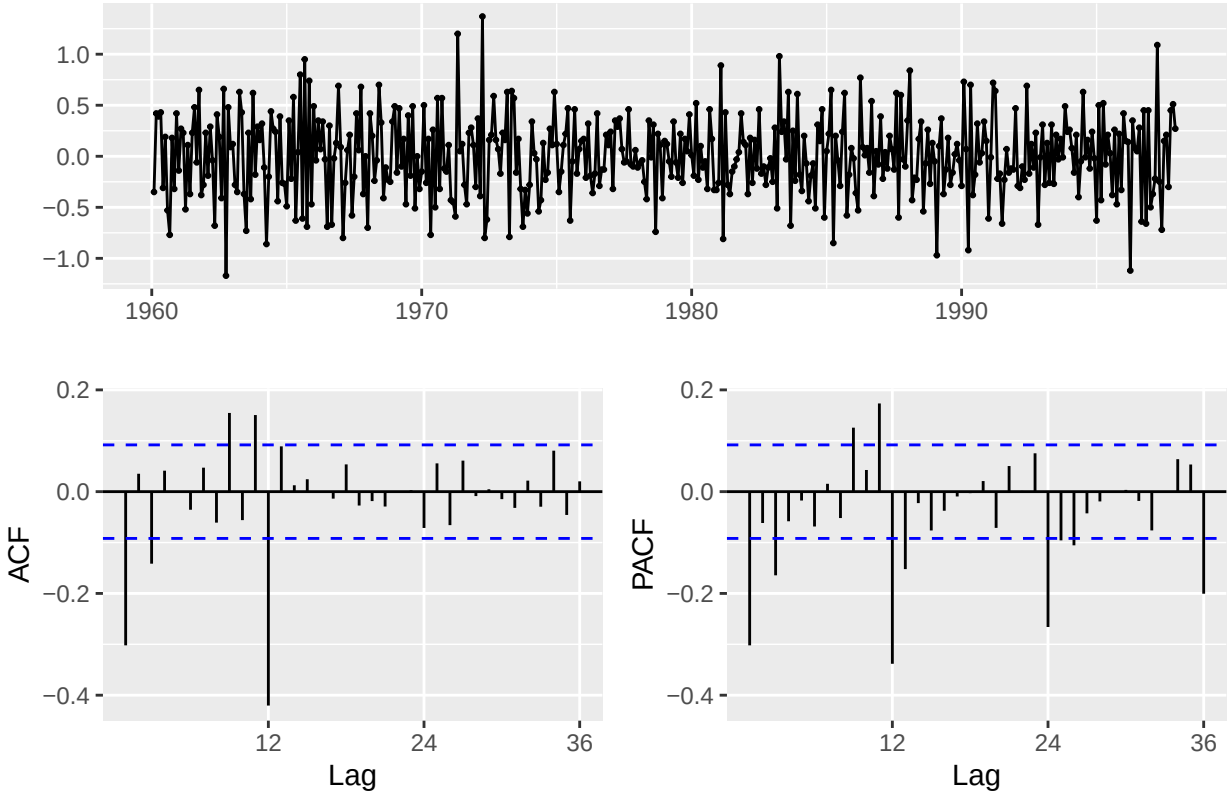
The seasonality can be removed by differencing the series over 12 lags. The resulting plots below show the remaining white noise and a slight positive trend. It is interesting to note that this plot also clearly shows the peaks where the trend component is not perfectly linear. With two significant peaks (~1974 and ~1988) it is assumed that a 3rd order polynomial regression will provide the best fit for the linear trend. The increase in the ACF plot after 24 lags and the significant lags in the PACF suggest that this simple 12-lag differencing has not fully explained the cyclical nature of this series.

```
co2 %>% diff(12) %>% ggtsdisplay()
```



Finally, the differentiation can be combined to remove both the linear and seasonal components from the series, yielding what appears to be stationary white noise.

```
co2 %>% diff(12) %>% diff() %>% ggtsdisplay()
```



## Part 2 (3 points)

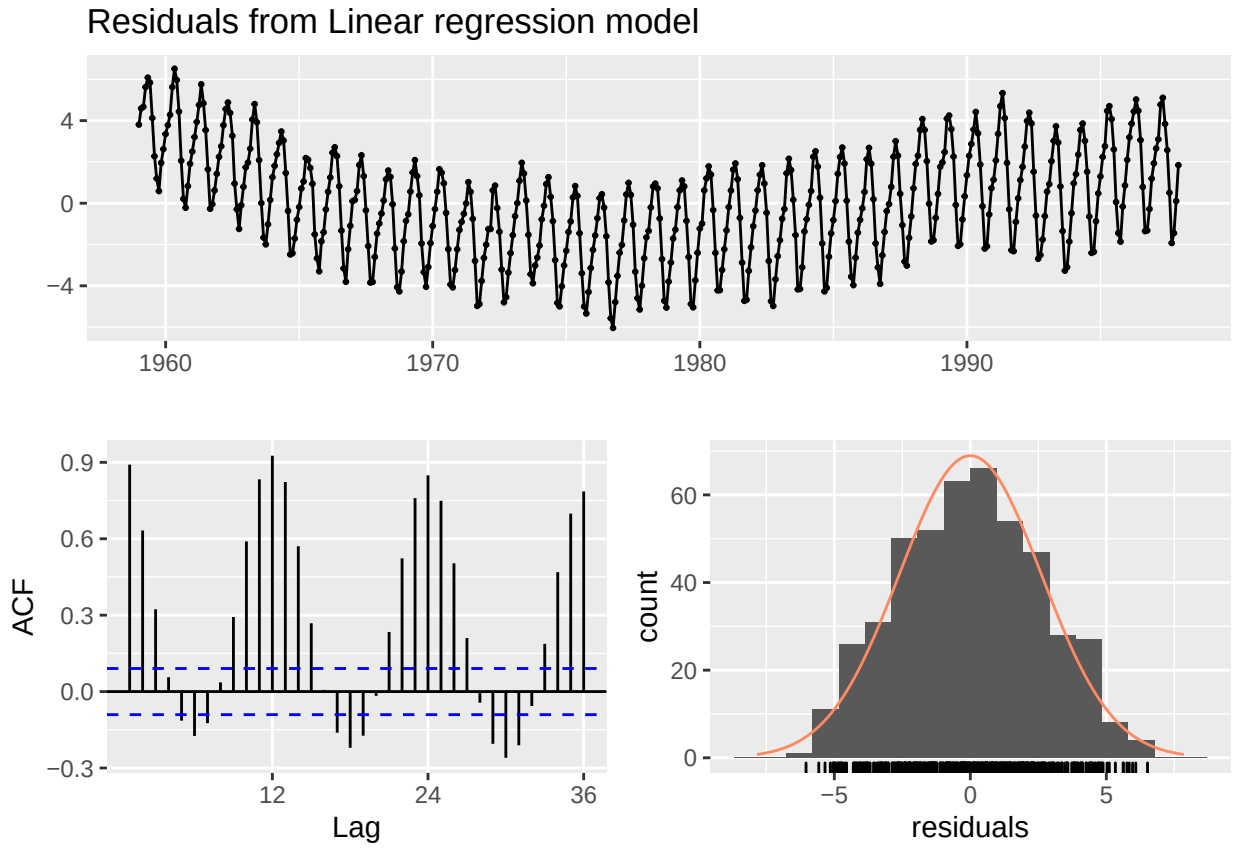
Fit a linear time trend model to the `co2` series, and examine the characteristics of the residuals. Compare this to a higher-order polynomial time trend model. Discuss whether a logarithmic transformation of the data would be appropriate. Fit a polynomial time trend model that incorporates seasonal dummy variables, and use this model to generate forecasts to the year 2020.

We'll start by performing a simple linear regression of the series and examine the residuals. As shown above, it was obvious that there were several points in the timeseries where the slope of the trend clearly changed, so we do not expect the residuals to have a constant mean. We also know that the linear model will not account for seasonality and therefore the residuals will not be stationary. The positive component of the ACF plot plot decays slowly over 36 lags, however the negative components are increasing. These negative peaks occur at the inverse point in the cycle which would be a six month offset. Considering the difference in rate of change in the season component's rise versus fall, this would imply that the rate of increase during prior years helps to explain the rate of decrease in the current year.

```
co2.lm <- tslm(co2 ~ trend)
summary(co2.lm)
```

```
##
## Call:
## tslm(formula = co2 ~ trend)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.0399 -1.9476 -0.0017  1.9113  6.5149
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.115e+02  2.424e-01  1284.9   <2e-16 ***
## trend       1.090e-01  8.958e-04   121.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.618 on 466 degrees of freedom
## Multiple R-squared:  0.9695, Adjusted R-squared:  0.9694
## F-statistic: 1.479e+04 on 1 and 466 DF,  p-value: < 2.2e-16
```

```
checkresiduals(co2.lm)
```



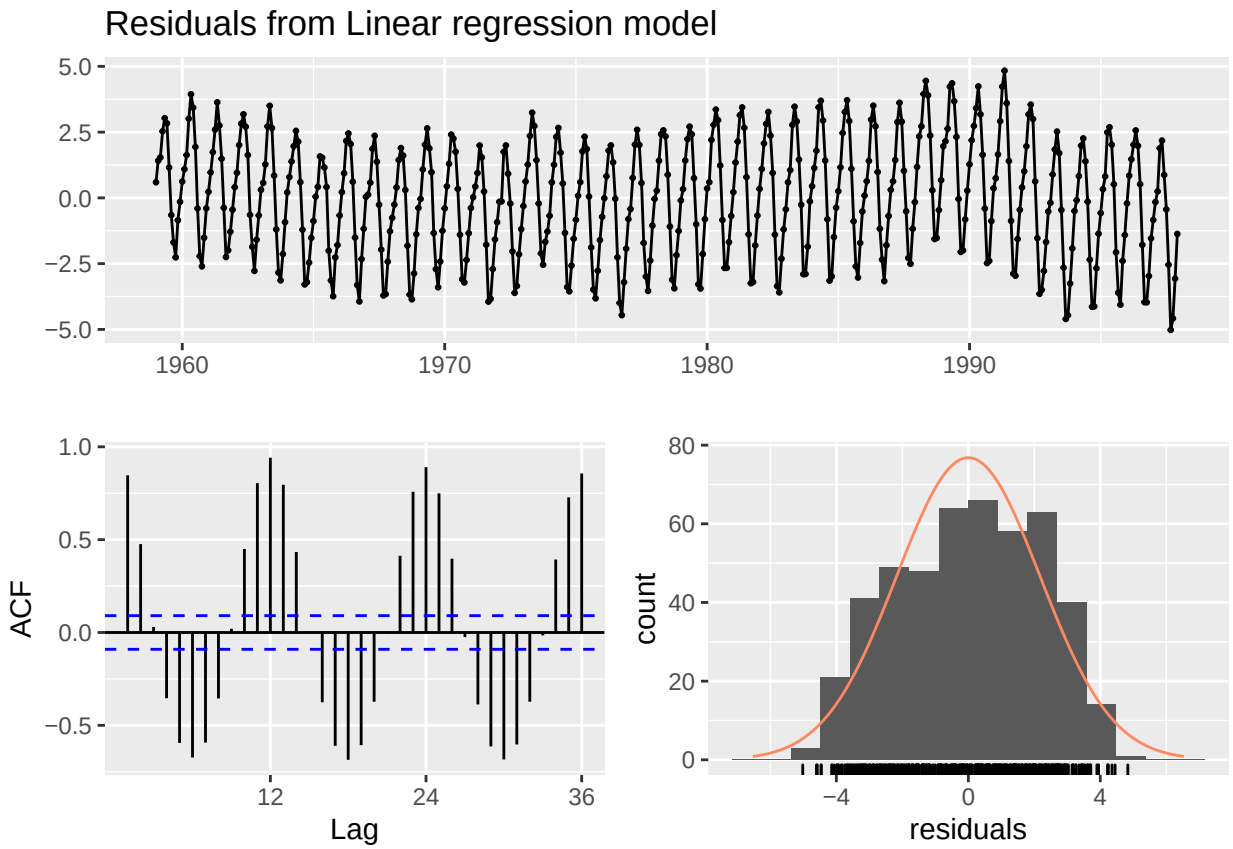

```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 457.32, df = 24, p-value < 2.2e-16
```

Next, second and third order polynomial fit is examined. The model summary indicates that all coeffients are significant up to order=3. Since we are only trying to explain the linear trend in this data the ideal residual plot would have constant mean and a uniform distribution. The second order polynomial plot still demonstrates a fair amount of drift over time and while there may still be a cyclical component to the residuals of the third order polynomial, they are much closer to having a constant mean. The ACF of the third order polynomial appears far more consistently seasonal than either of the lower order regressions, and the histogram of values is closer to the expected uniform distribution.

```
co2.lm2 <- tslm(co2 ~ trend + I(trend^2))
co2.lm3 <- tslm(co2 ~ trend + I(trend^2) + I(trend^3))
summary(co2.lm2)
```

```
##
## Call:
## tslm(formula = co2 ~ trend + I(trend^2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.0195 -1.7120  0.2144  1.7957  4.8345
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.148e+02  3.039e-01 1035.65   <2e-16 ***
## trend       6.739e-02  2.993e-03   22.52   <2e-16 ***
## I(trend^2)  8.862e-05  6.179e-06   14.34   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.182 on 465 degrees of freedom
## Multiple R-squared:  0.9788, Adjusted R-squared:  0.9787
## F-statistic: 1.075e+04 on 2 and 465 DF,  p-value: < 2.2e-16
```

```
checkresiduals(co2.lm2)
```



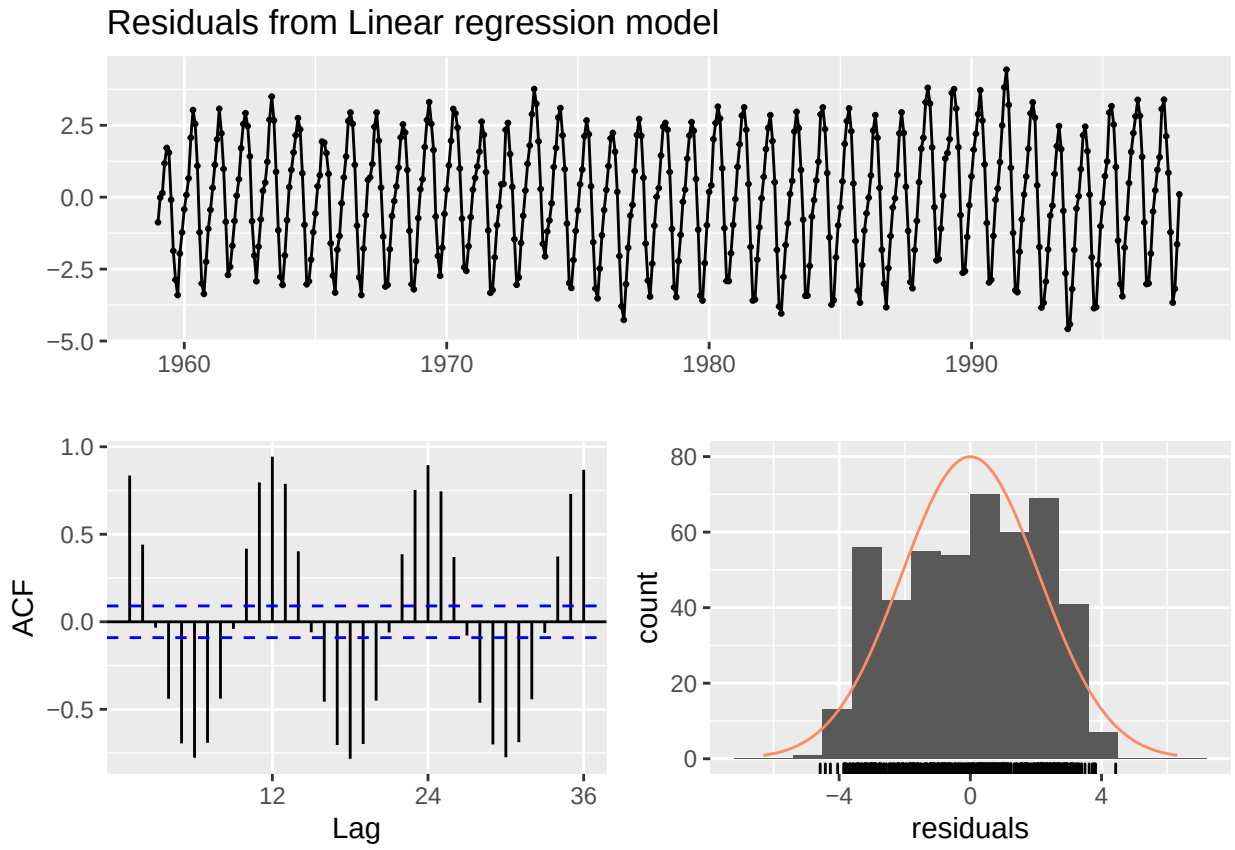

```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 456.54, df = 24, p-value < 2.2e-16
```

```
summary(co2.lm3)
```

```
##
## Call:
## tslm(formula = co2 ~ trend + I(trend^2) + I(trend^3))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5786 -1.7299  0.2279  1.8073  4.4318
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.163e+02  3.934e-01 804.008  < 2e-16 ***
## trend        2.905e-02  7.256e-03   4.004 7.25e-05 ***
## I(trend^2)   2.928e-04  3.593e-05   8.149 3.44e-15 ***
## I(trend^3)  -2.902e-07  5.036e-08  -5.763 1.51e-08 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.11 on 464 degrees of freedom
## Multiple R-squared:  0.9802, Adjusted R-squared:  0.9801
## F-statistic:  7674 on 3 and 464 DF,  p-value: < 2.2e-16
```

```
checkresiduals(co2.lm3)
```

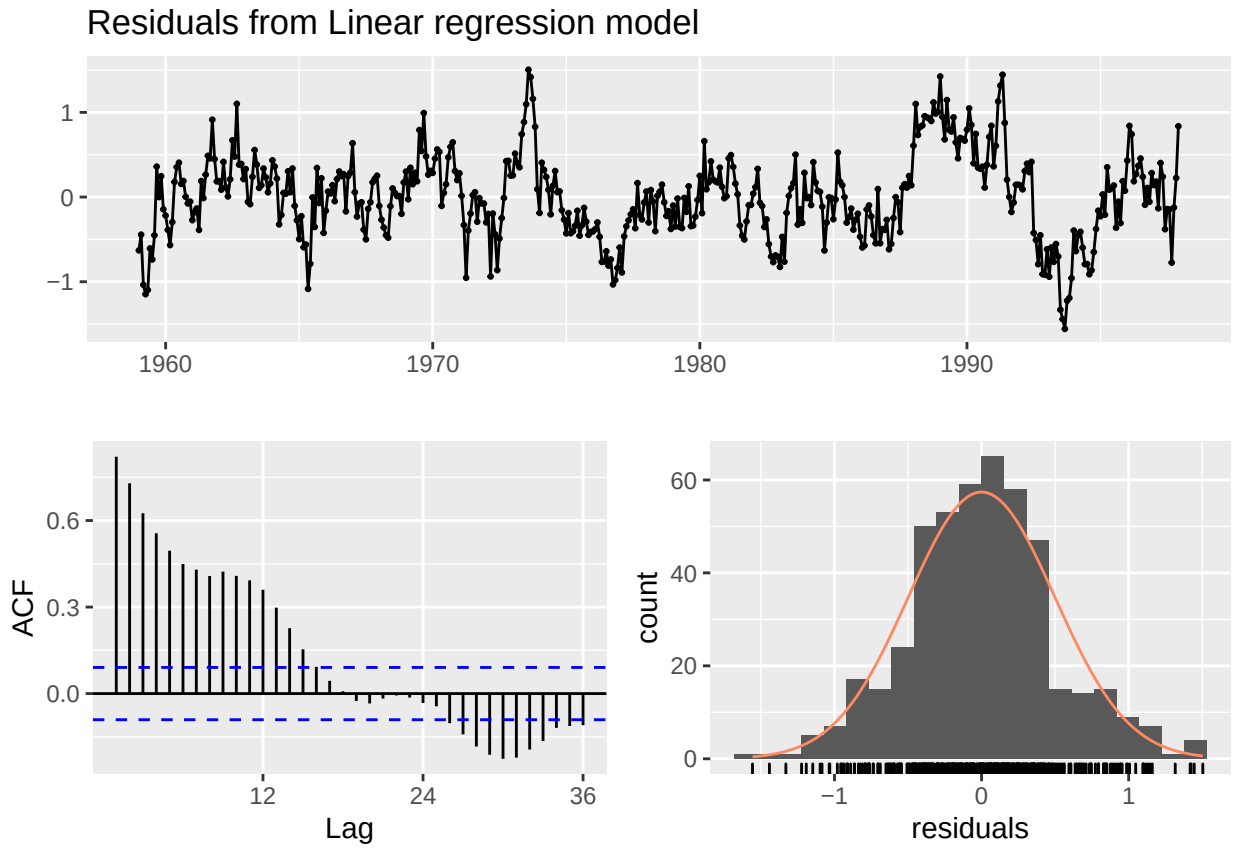



```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 455.88, df = 24, p-value < 2.2e-16
```

Next we considered whether a log-linear transformation would add any valuable information to the model. Since the data shows a fairly constant variance and a strong linearity between the co2 concentration trend and our time index, we did not feel that a logarithmic transformation would be useful. The data does not represent any evidence that the relationship is exponential, which is best demonstrated by the season plots above. The year-on-year increase in co2 concentration appears to be nearly constant at this time interval, so transformations will not be explored.

Applying dummy variables for the seasonality component should yield stationary residuals. The dummy variables were defined as a factor variable indicate each month, and are added to the third order polynomial regression. Now that the seasonal component has been accounted for the residuals are much closer to demonstrating a constant mean and the ACF shows a more stable decay. The residuals are becoming more normally distributed which is desirable to acheiving stationarity.

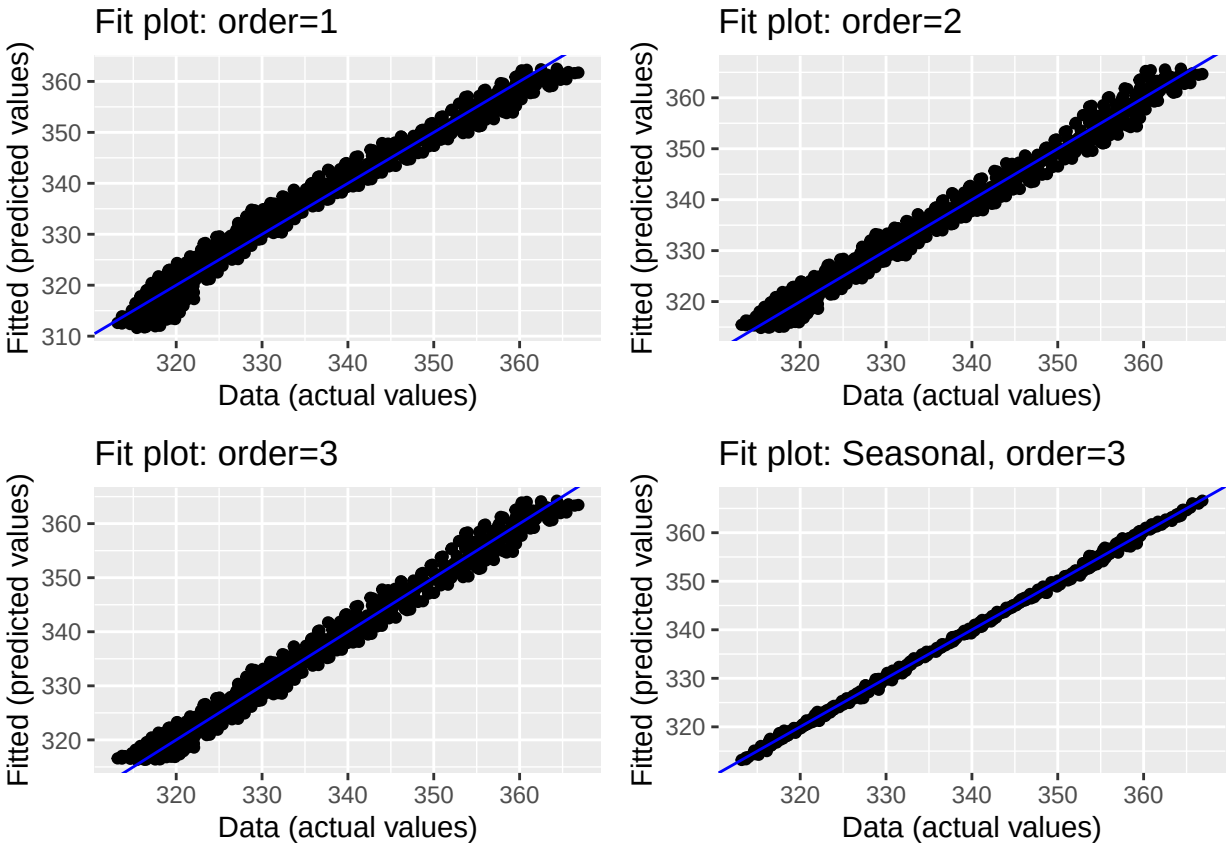
```
co2.lm3.Seas <- tslm(co2 ~ trend + I(trend^2)+ I(trend^3) + season)
checkresiduals(co2.lm3.Seas)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 332.67, df = 24, p-value < 2.2e-16
```

Fitment plots comparing the four models are shown below. The residuals in the second and third order polynomial regressions appear nearly identical, and much more linear than the first order regression. Once the seasonal component has been accounted for the residuals are dramatically reduced yielding a much tighter model fit.

```
fit.plot.lm <- cbind(Data = co2, Fitted = fitted(co2.lm)) %>%
  as.data.frame() %>%
  ggplot(aes(x=Data, y=Fitted)) +
    geom_point() +
    ylab("Fitted (predicted values)") + xlab("Data (actual values)") +
    ggtitle("Fit plot: order=1") +
    geom_abline(intercept=0, slope=1, col='blue')

fit.plot.lm2 <- cbind(Data = co2, Fitted = fitted(co2.lm2)) %>%
  as.data.frame() %>%
  ggplot(aes(x=Data, y=Fitted)) +
    geom_point() +
    ylab("Fitted (predicted values)") + xlab("Data (actual values)") +
    ggtitle("Fit plot: order=2") +
    geom_abline(intercept=0, slope=1, col='blue')

fit.plot.lm3 <- cbind(Data = co2, Fitted = fitted(co2.lm3)) %>%
  as.data.frame() %>%
  ggplot(aes(x=Data, y=Fitted)) +
    geom_point() +
    ylab("Fitted (predicted values)") + xlab("Data (actual values)") +
    ggtitle("Fit plot: order=3") +
    geom_abline(intercept=0, slope=1, col='blue')

fit.plot.lm3.Seas <- cbind(Data = co2, Fitted = fitted(co2.lm3.Seas)) %>%
  as.data.frame() %>%
  ggplot(aes(x=Data, y=Fitted)) +
    geom_point() +
    ylab("Fitted (predicted values)") + xlab("Data (actual values)") +
    ggtitle("Fit plot: Seasonal, order=3") +
    geom_abline(intercept=0, slope=1, col='blue')

ggarrange(fit.plot.lm, fit.plot.lm2, fit.plot.lm3, fit.plot.lm3.Seas,
                    ncol = 2, nrow = 2)
```



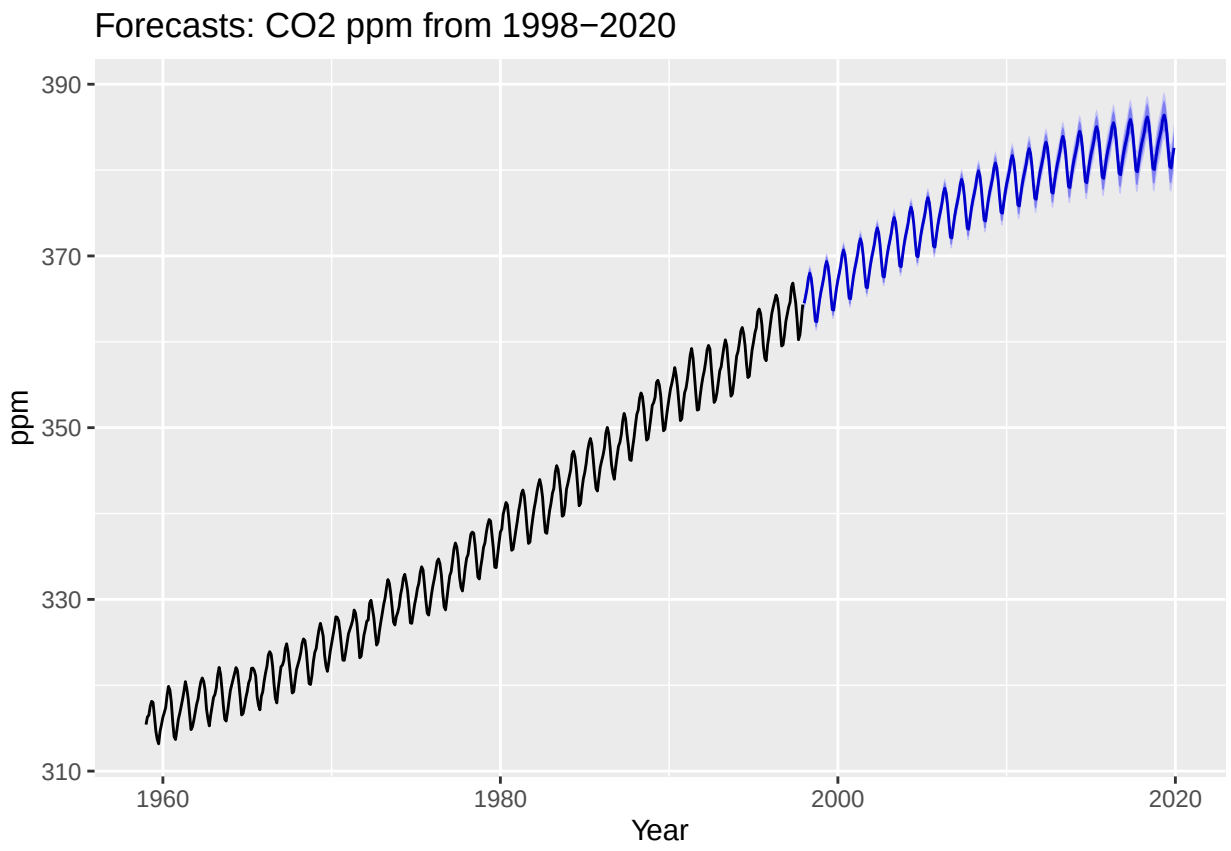

The following table shows various model diagnostics. AIC and BIC values are not useful when comparing models with different levels of differencing, so the RMSE is used to evaluate the model fit. The non-seasonal models clearly improve by all metrics as the polynomial order increases, giving confidence to our decision to use the third order trend in the final model with seasonal components. The AdjR2, deviance, and RMSE values clearly indicate that the seasonal dummy variables provided a valuable contribution to the model.

```
data.frame( model=c('lm','lm2','lm3','lm3.Seas'),
      round(rbind(
          CV(co2.lm), CV(co2.lm2), CV(co2.lm3), CV(co2.lm3.Seas)),6),
      deviance = round(rbind(
        deviance(co2.lm), deviance(co2.lm2),
        deviance(co2.lm3), deviance(co2.lm3.Seas)),4),
      RMSE = round(rbind(
        RMSE(RMSE(co2.lm$residuals)), RMSE(RMSE(co2.lm2$residuals)),
        RMSE(RMSE(co2.lm3$residuals)), RMSE(RMSE(co2.lm3.Seas$residuals))),4))
```

```
##      model       CV       AIC      AICc       BIC    AdjR2 deviance   RMSE
## 1       lm 6.888531  904.8343  904.8861  917.2798 0.969399 3194.080 2.6125
## 2      lm2 4.794167  735.4090  735.4954  752.0029 0.978739 2214.454 2.1753
## 3      lm3 4.490846  705.0603  705.1902  725.8026 0.980116 2066.558 2.1014
## 4 lm3.Seas 0.264132 -621.6144 -620.4082 -555.2389 0.998859  115.799 0.4974
```

Finally, the seasonal model is used to forecast future co2 levels out to 2020. The forecast function is used to apply the model to the future time period defined as 12 months * 22 years. This is plotted along with the original dataset showing what appears to be a reasonable (yet unfortunately optimistic) trend for future co2 concentrations. Confidence intervals are included on the plot demonstrating the tight model fit on past values.

```
fcast.co2.2020 <- forecast(co2.lm3.Seas, h=12*22)
autoplot(fcast.co2.2020) +
  ggtitle("Forecasts: CO2 ppm from 1998-2020") +
  xlab("Year") + ylab("ppm")
```

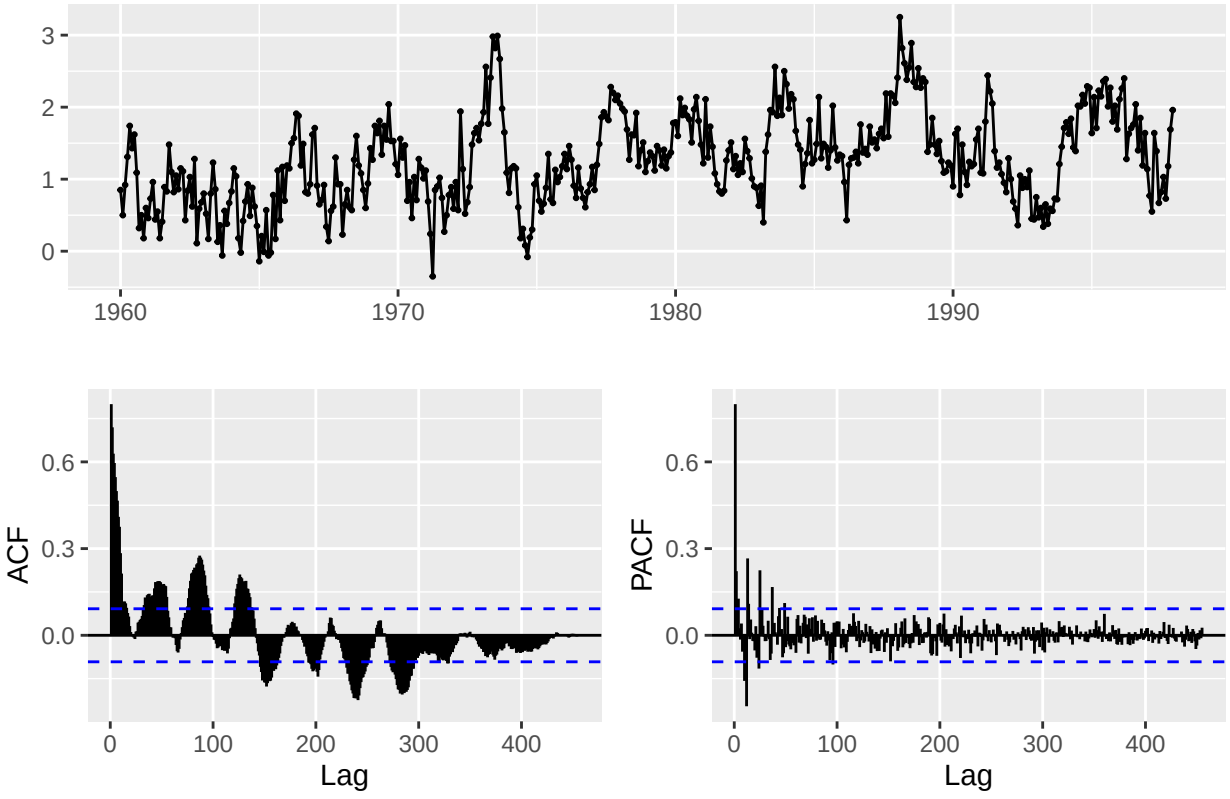


## Part 3 (4 points)

**Following all appropriate steps, choose an ARIMA model to fit to the series. Discuss the characteristics of your model and how you selected between alternative ARIMA specifications. Use your model to generate forecasts to the year 2020.**

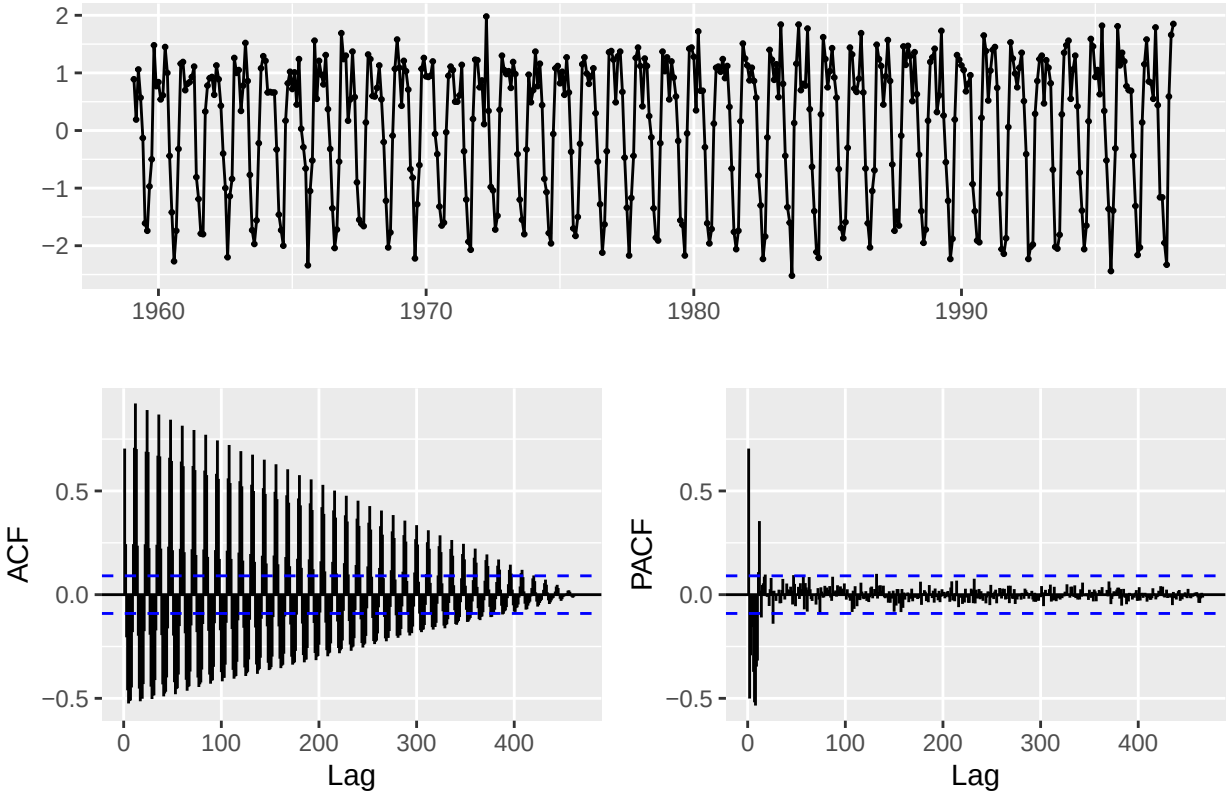
We know the CO2 series is not stationary because there is a strong trend as well as seasonality. So first we want to check whether we can turn the series into a stationary series via differencing. Below we perform seasonal differencing and note a persisitent pattern in the ACF plot. Thus, seasonal differencing does not turn the series into stationary series.

```
co2 %>% diff(lag=12) %>% ggtsdisplay(lag.max = 468)
```



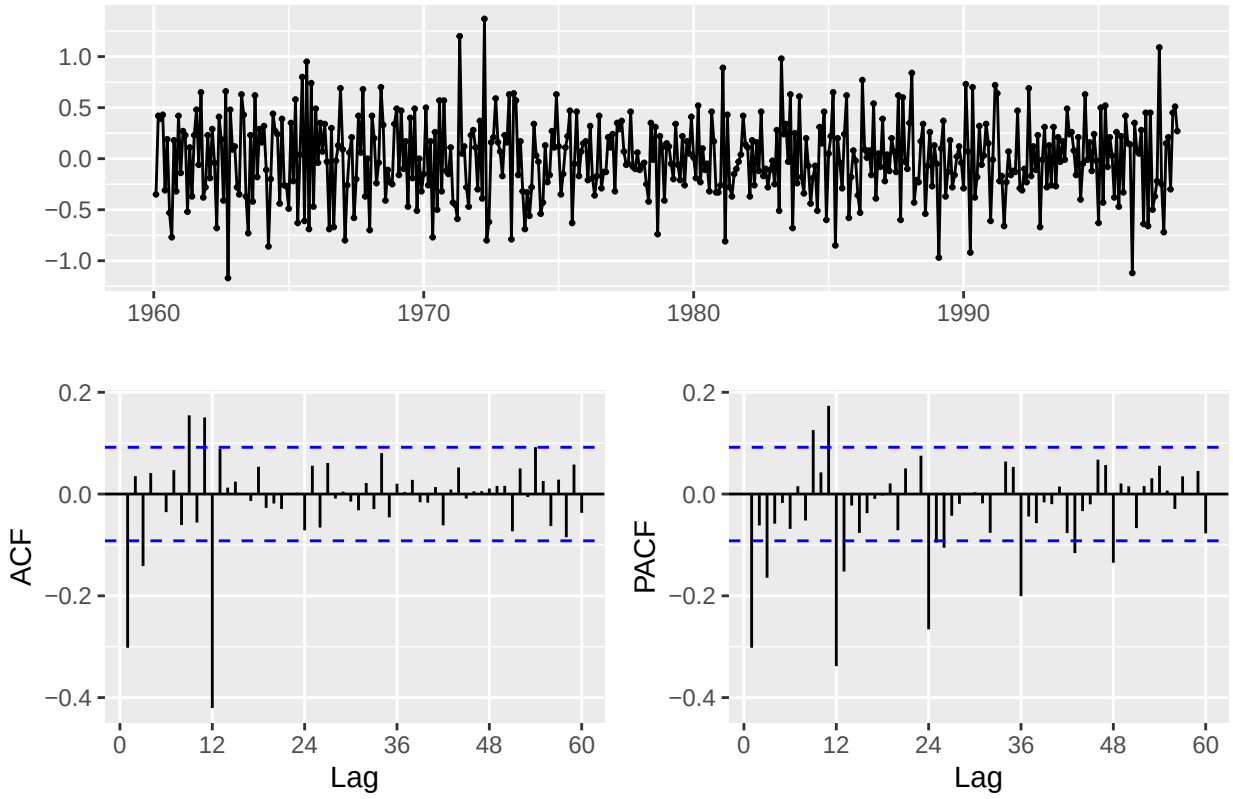
Below we perform non-seasonal differencing and note a persisitent pattern in the ACF plot. Thus, non-seasonal differencing does not turn the series into stationary series.

```
co2 %>% diff(lag=1) %>% ggtsdisplay(lag.max = 468)
```



Below we perform non-seasonal differencing on top of seasonal differencing. We note the ACF plot has a cut off after lag 12 which is like white noise. We also note significant autocorrelation at lag 1 and lag 12. Thus we will start with $d = 1$, $D = 1$, $q = 1$, $Q = 1$. Looking at the PACF plot we know there is significant partial correlation at lag 1, lag 3, lag 9, lag 11, lag 12. Thus we start $p = 1$ and $P = 1$.

```
co2 %>% diff(lag=12) %>% diff(lag=1) %>% ggtsdisplay(lag.max = 60)
```




We then loop through various SARIMA models using AICc as selection criteria.

```
results <- data.frame(p=integer(),
                      q=integer(),
                      P=integer(),
                      Q=integer(),
                      AICc=double())
for (p in 1:2){
  for (q in 1:3){
    for(P in 1:2){
      for (Q in 1:3){
        tryCatch(
          {
            mod <- co2 %>% as_tsibble() %>%
              model(ARIMA(value ~ 0 + pdq(p,1,q) + PDQ(P,1,Q)))
```

```
            if(has_name(glance(mod),'AICc')){
          }
        results <- results %>% add_row(p=p, q = q, P=P, Q=Q, AICc = as.numeric(glance(mod)$AIC
        print(paste(p, q, P, Q, as.numeric(glance(mod)$AICc)))
          },
        error=function(e) {
          print(paste('error encountered for', p, q, P, Q, e))
          }
        )
      }
    }
  }
}
```

View best model diagnostics:

```
results[which.min(results$AICc), ]
```

```
##    p q P Q    AICc
## 77 2 3 2 2 171.593
```

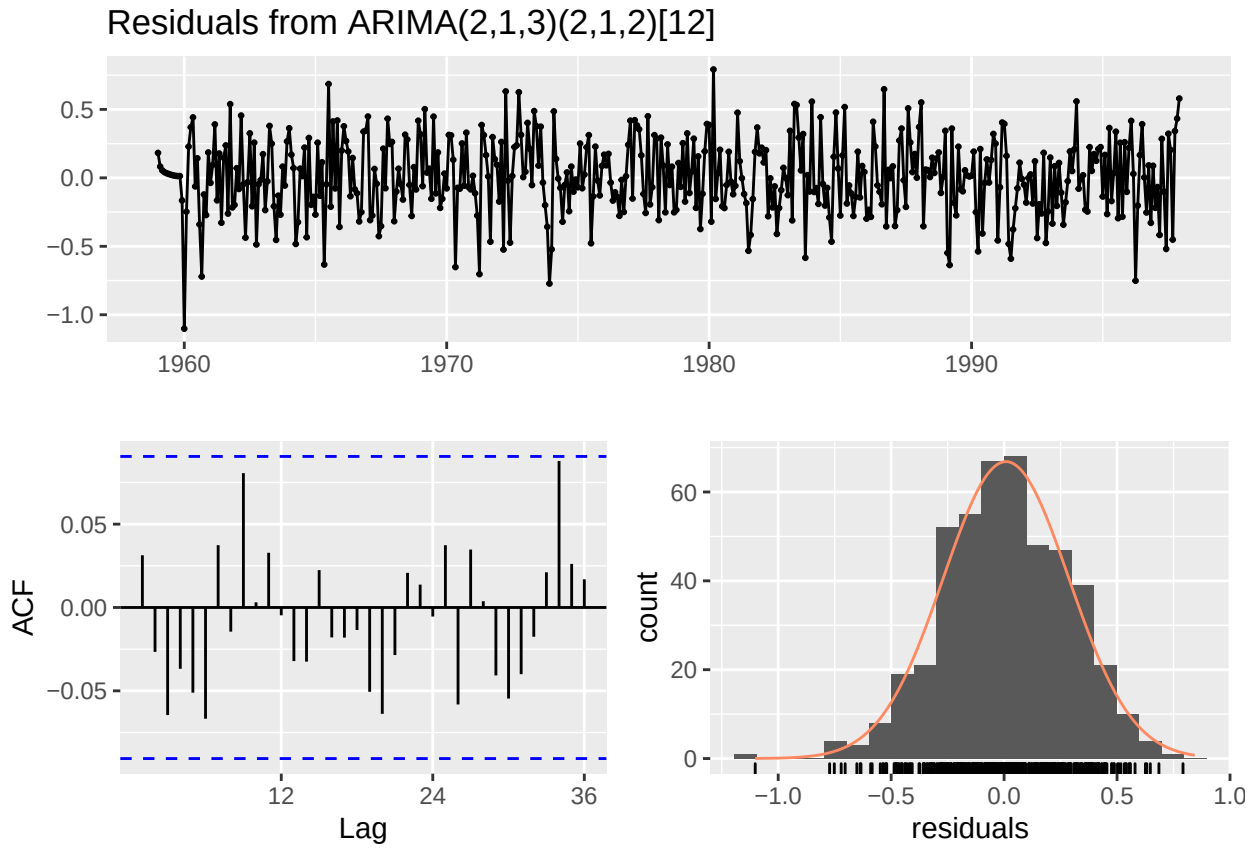
Of the models produced, smallest AICc is with SARIMA$(2,1,3)(2,1,2)_1 2$

```
fit.213.212 <-Arima(co2, order=c(2,1,3), seasonal = c(2,1,2))
summary(fit.213.212)
```

```
## Series: co2
## ARIMA(2,1,3)(2,1,2)[12]
##
## Coefficients:
##           ar1      ar2      ma1     ma2      ma3    sar1     sar2     sma1
##       -0.0178  -0.9465  -0.3367  0.9792  -0.3900  0.9738  -0.1410  -1.8176
## s.e.   0.0227   0.0192   0.0533  0.0190   0.0522  0.0784   0.0612   0.0696
##         sma2
##       0.8578
## s.e.  0.0618
##
## sigma^2 estimated as 0.08125:  log likelihood=-75.55
## AIC=171.1   AICc=171.59   BIC=212.3
##
## Training set error measures:
##                       ME      RMSE       MAE          MPE      MAPE      MASE
## Training set 0.00853114 0.2782601 0.2220805 0.002554682 0.0659964 0.1753935
##                     ACF1
## Training set 0.03132742
```

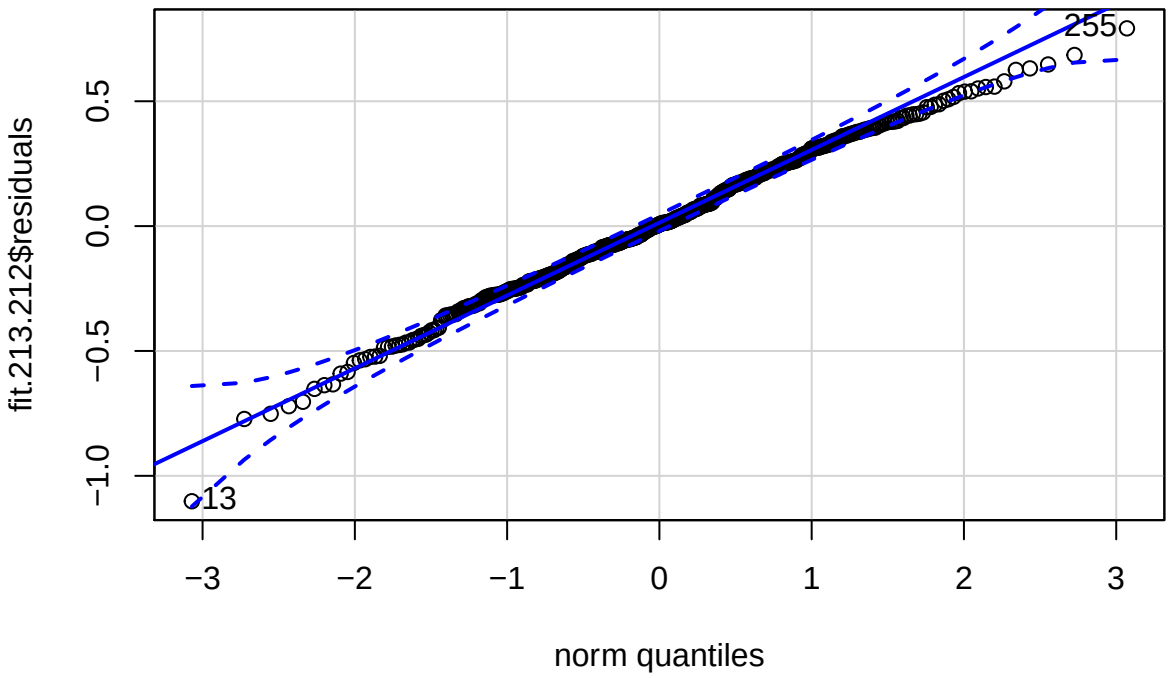
The residuals provide means to see the difference between the fitted and observed values. The plots show that the model has adequately captured the information in the data. The residuals have no significant ACF and have mean close to zero. In addition, the residual have reasonably constant variance and fairly normaly distributed.

```
checkresiduals(fit.213.212)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,3)(2,1,2)[12]
## Q* = 16.811, df = 15, p-value = 0.3303
##
## Model df: 9.   Total lags used: 24
```

```
qqPlot(fit.213.212$residuals)
```




```
## [1]  13 255
```

Both portmanteau tests have p-values that are relatively large suggesting that the residuals are white noise.

```
Box.test(residuals(fit.213.212))
```

```
##
##  Box-Pierce test
##
## data:  residuals(fit.213.212)
## X-squared = 0.4593, df = 1, p-value = 0.498
```

```
Box.test(residuals(fit.213.212), type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  residuals(fit.213.212)
## X-squared = 0.46225, df = 1, p-value = 0.4966
```

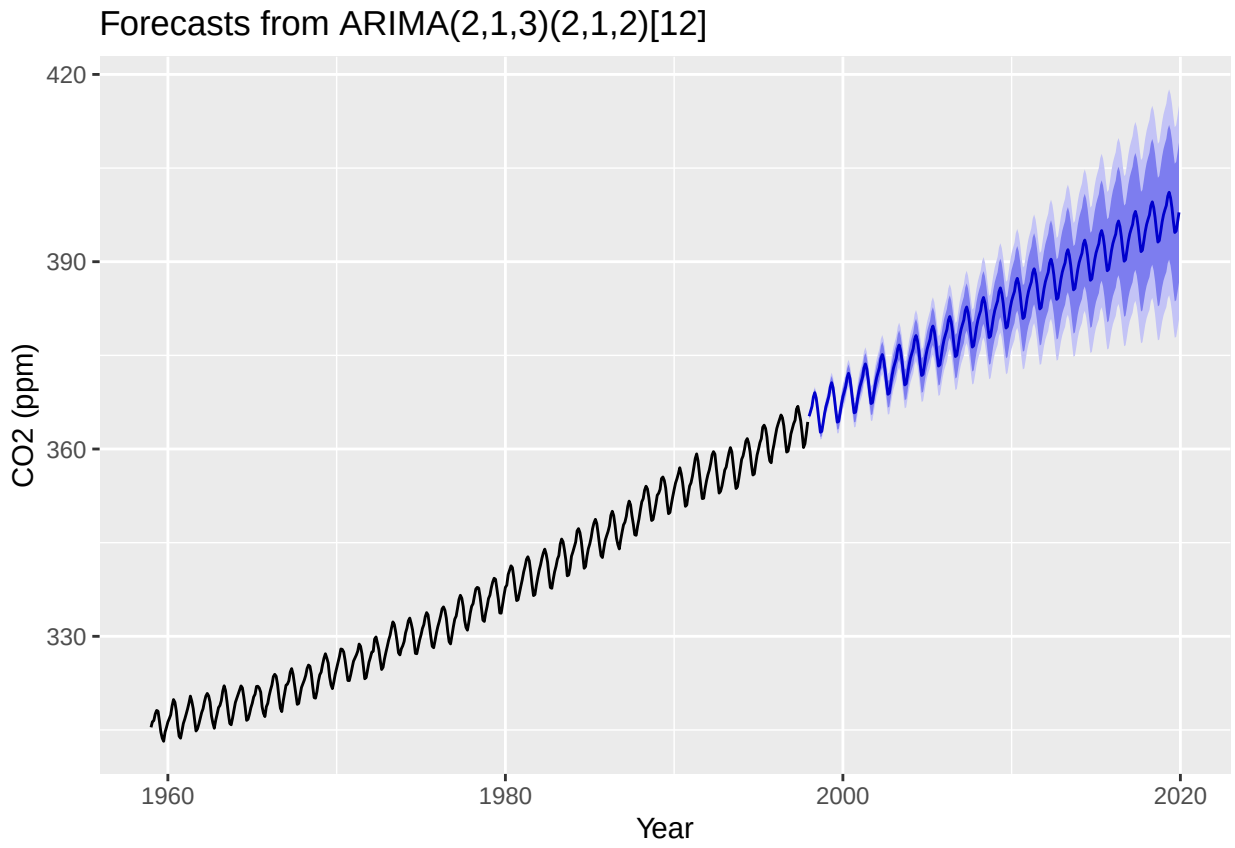
Try auto arima to compare and our SARIMA model does better.

```
fit.auto <-auto.arima(co2)
summary(fit.auto)
```

```
## Series: co2
## ARIMA(1,1,1)(1,1,2)[12]
##
## Coefficients:
##          ar1      ma1     sar1     sma1     sma2
##       0.2569  -0.5847  -0.5489  -0.2620  -0.5123
## s.e.  0.1406   0.1203   0.5881   0.5703   0.4820
##
## sigma^2 estimated as 0.08576:  log likelihood=-84.39
## AIC=180.78   AICc=180.97   BIC=205.5
##
## Training set error measures:
##                       ME     RMSE       MAE         MPE       MAPE      MASE
## Training set 0.01742092 0.287159 0.2303994 0.005073769 0.06845665 0.1819636
##                       ACF1
## Training set -0.002858162
```

Forecasts to 2020 shown with 80 and 95% prediciton intervals.

```
forecast.co2.2020.arima <- forecast(fit.213.212, h=264)
autoplot(forecast.co2.2020.arima) +
  ylab('CO2 (ppm)') + xlab('Year') +
  guides(fill = guide_legend(title = 'level'))
```



## # 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, and address the problem of missing observations. Describe how the Keeling Curve evolved from 1997 to the present and compare this to the predicitons from your forecasts in 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 the data file and apply column names.

```
df <- read.table('co2_weekly_mlo.txt', col.names = c('yr', 'mo', 'day',
                                                     'decimal', 'ppm', '#days',
                                                     'inc.1.yr.ago', 'inc.10.yr.ago', 'inc.sin
head(df)
```

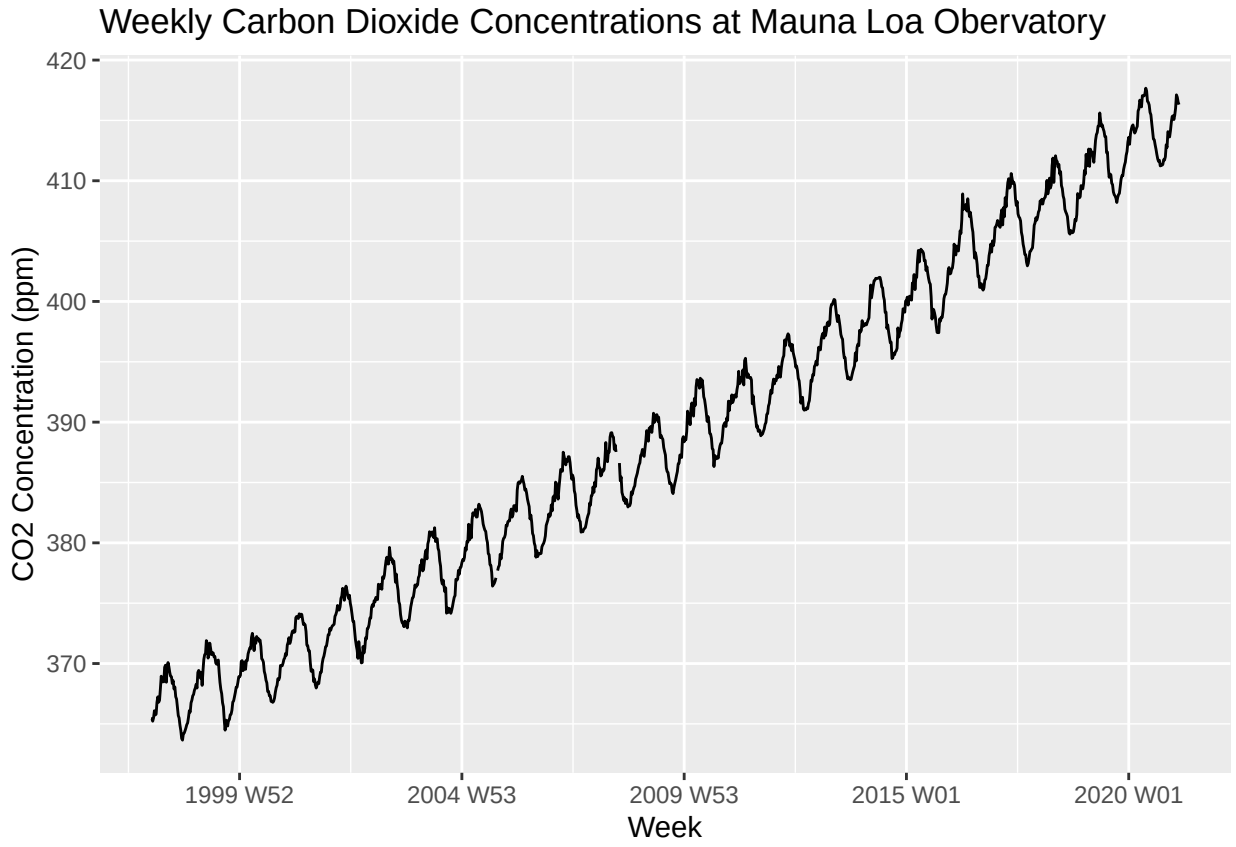
```
##     yr mo day  decimal    ppm X.days inc.1.yr.ago inc.10.yr.ago inc.since.1800
## 1 1974  5  19 1974.380 333.37      5      -999.99       -999.99          50.40
## 2 1974  5  26 1974.399 332.95      6      -999.99       -999.99          50.06
## 3 1974  6   2 1974.418 332.35      5      -999.99       -999.99          49.60
## 4 1974  6   9 1974.437 332.20      7      -999.99       -999.99          49.65
## 5 1974  6  16 1974.456 332.37      7      -999.99       -999.99          50.06
## 6 1974  6  23 1974.475 331.73      5      -999.99       -999.99          49.72
```

Create a Date new column conforming to the Date class, and remove invalid data in which missing values are coded as -999.99. We know co2 concentration cannot be negative so change any negaive values to `NA`.

```
df$date <- as.Date(with(df, paste(yr, mo, day, sep = '/')), '%Y/%m/%d')
df <- df %>% mutate(ppm = replace(ppm, ppm<0, NA))
```

The dataframe is converted to a tsible object with the week defined as the index and filtered to include data from 1998 on. A plot of the new dataset shows numerous gaps in the data.

```
co2_weekly <- tsibble(week = yearweek(df$date), value = df$ppm, index = week)
co2_weekly_98 <- co2_weekly %>% filter_index('1998'~.)
autoplot(co2_weekly_98) +
  ggtitle('Weekly Carbon Dioxide Concentrations at Mauna Loa Obervatory') +
  xlab('Week') +
  ylab('CO2 Concentration (ppm)')
```

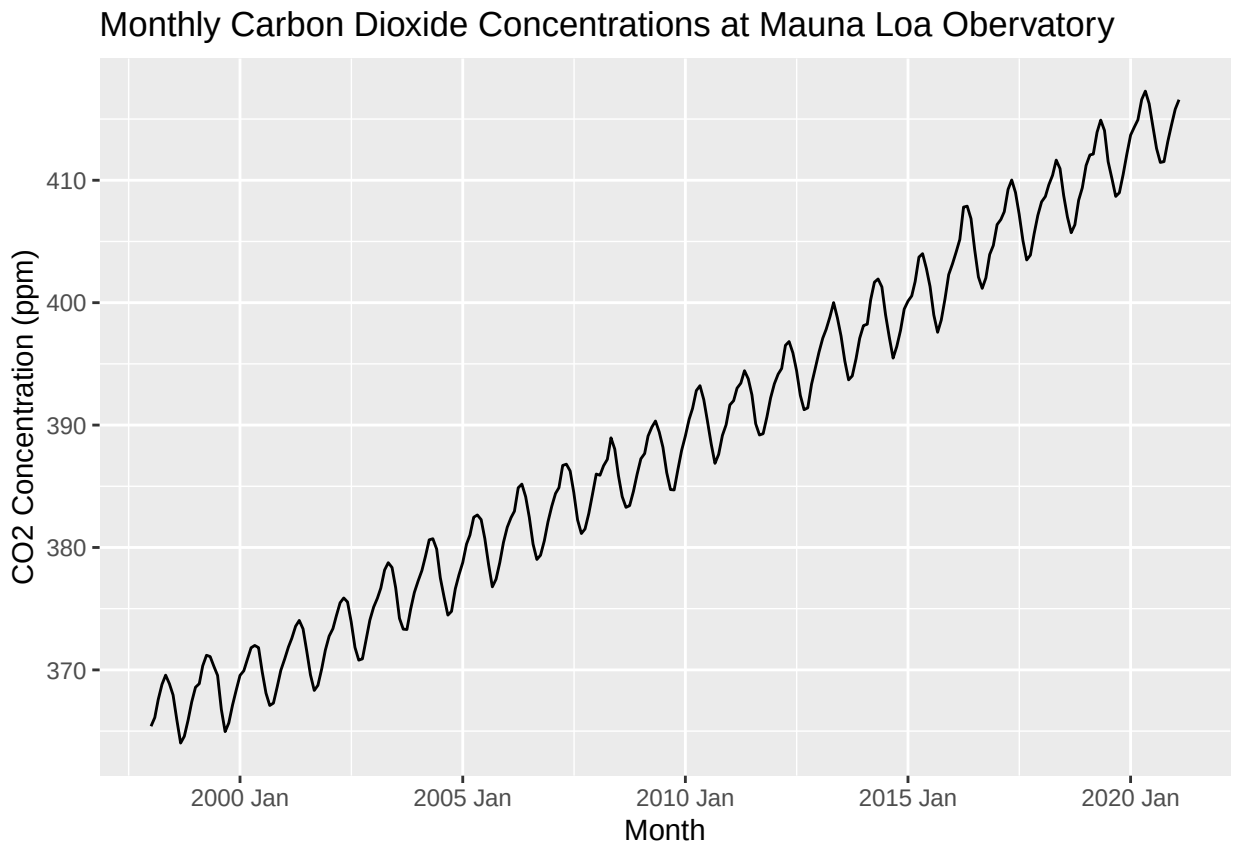



Looking at a plot of the weekly data, we can clearly see the trend and seasonal components and note that the few months before each seasonal peak have quite a bit more variation than can be discerned from the monthly aggregation.

Since the data gaps are relatively short and do not seem to mask the peaks or valleys of the seasonal cycling, it is reasonable to simply aggregate these values as monthly means. The `aggregate` function is used to perform monthly averaging across the original dataframe. A tsibble object is applied with a monthly index and filtered to remove any data before 1998. The resulting plot shows that the missing values have been aggregated into a continuous series.

```
co2_monthly <- aggregate(ppm ~ mo + yr, df, FUN = mean)
co2_monthly$date <- as.Date(with(co2_monthly, paste(yr, mo, 1, sep = '/')), '%Y/%m/%d')
co2_monthly <- tsibble(month = yearmonth(co2_monthly$date), value = co2_monthly$ppm, index = mo
co2_monthly_98 <- co2_monthly %>% filter_index('1998'~.)
```

```
autoplot(co2_monthly_98) +
  ggtitle('Monthly Carbon Dioxide Concentrations at Mauna Loa Obervatory') +
  xlab('Month') +
  ylab('CO2 Concentration (ppm)')
```



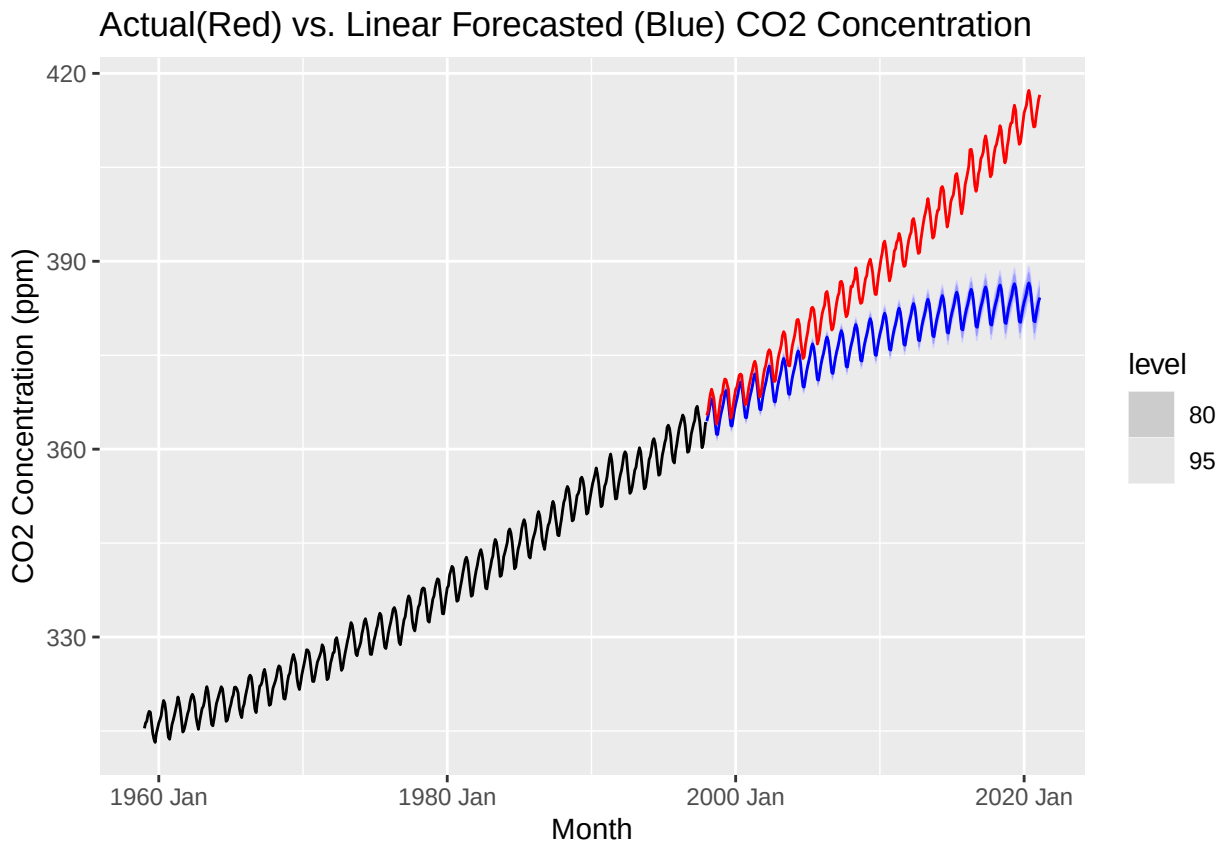

To compare the accuracy of our initial models against actual data, we first rerun the models chosen in previous sections for clarity. Fable is used instead Forecase for these models because it simplified the creation of plots.

```
co2_tsibble <- co2 %>% as_tsibble() %>% rename(month = index)
co2.training <- co2_tsibble
co2.actual <- co2_monthly_98
```

First, the data used to train the previous models (black) is plotted with it's forecast out to 2020 (blue). The forecast plot shows the estimated trend lines along with their 95% and 80% confidence intervals. The actual observed data is plotted in red, showing that the linear model has drastically diverged from the actual situation.

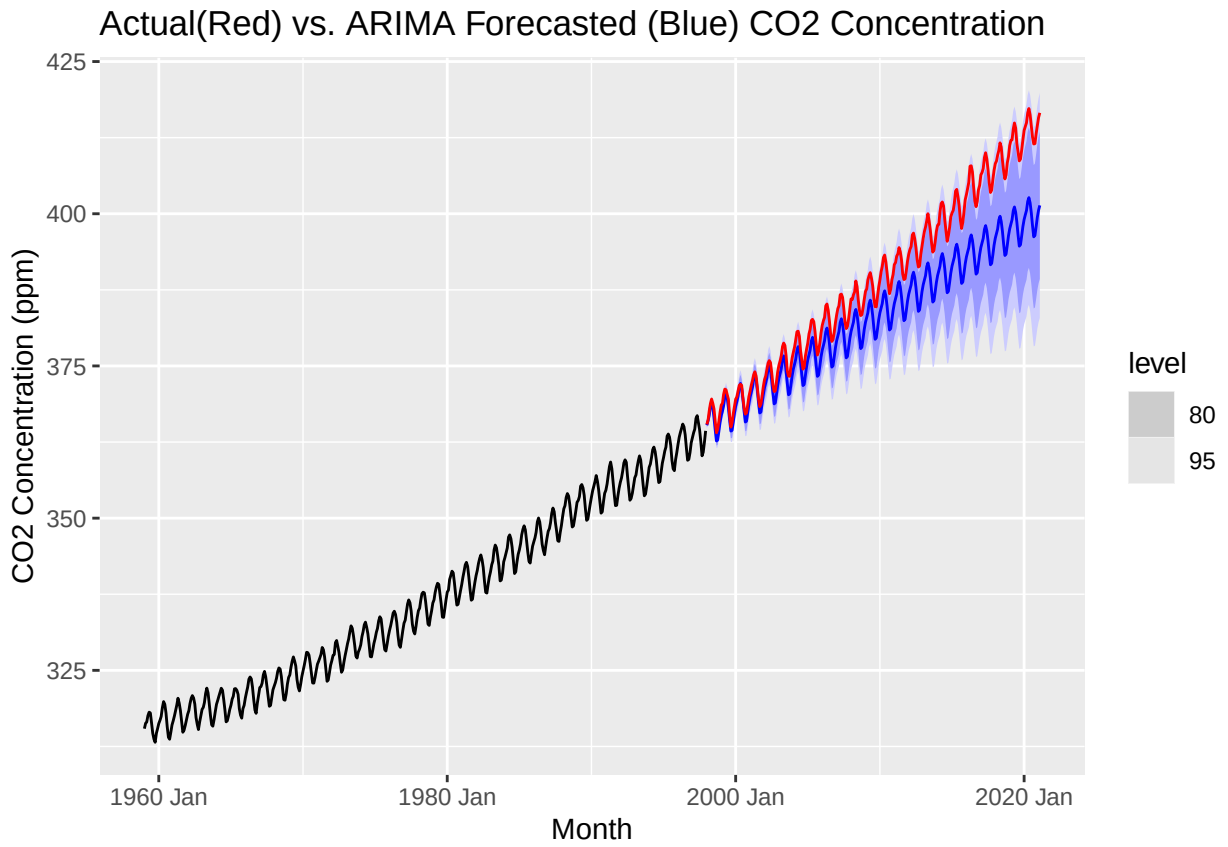
```
# Recreate models using Fable for better plotting
co2.TSLM <- co2_tsibble %>%
  model(TSLM(value ~ trend() + I(trend()^2) + I(trend()^3) + season()))
forecast(co2.TSLM, h=278) %>% autoplot() +
```

```
  autolayer(co2.actual, value, colour = 'red') +
  autolayer(co2.training, value, colour = 'black') +
  ggtitle('Actual(Red) vs. Linear Forecasted (Blue) CO2 Concentration') +
  xlab('Month') +
  ylab('CO2 Concentration (ppm)')
```




Next, the SARIMA model chosen from our previous iterative tests is plotted along with actual data. It's clear that the SARIMA model provides a much better forecast than the linear model, however the actual values are way out at the edge of the 80% confidence interval and it appears that the trends may diverge significantly in the coming years.

```
# Create ARIMA model using best parameters from iterative testing
co2.ARIMA <- co2_tsibble %>%
  model(ARIMA(value ~ 0 + pdq(2,1,3) + PDQ(2,1,2)))

forecast(co2.ARIMA, h=278) %>% autoplot() +
    autolayer(co2.actual, value, colour = 'red') +
  autolayer(co2.training, value, colour = 'black') +
    ggtitle('Actual(Red) vs. ARIMA Forecasted (Blue) CO2 Concentration') +
    xlab('Month') +
    ylab('CO2 Concentration (ppm)')
```

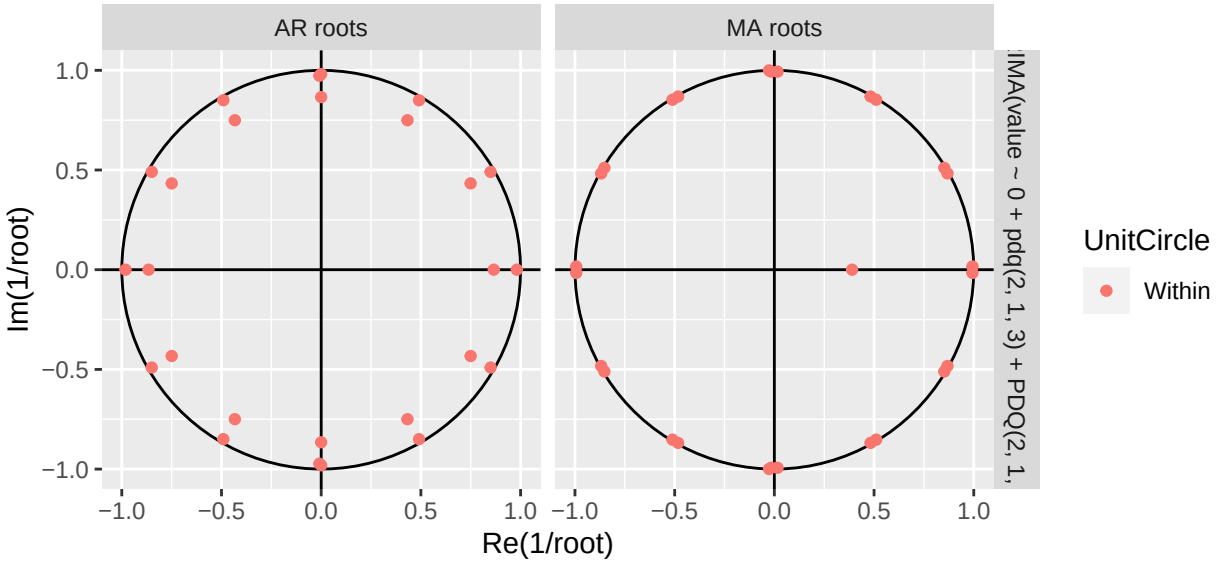



And finally the inverse AR and MA roots are plotted against unit circles to confirm that they are all less than or equal to one.

```
glance(co2.ARIMA)
```

```
## # A tibble: 1 x 8
##   .model                    sigma2 log_lik   AIC  AICc   BIC ar_roots  ma_roots
##   <chr>                      <dbl>   <dbl> <dbl> <dbl> <dbl> <list>    <list>
## 1 ARIMA(value ~ 0 + pdq(2, ~ 0.0812   -75.5  171.  172.  212. <cpl [26~ <cpl [2~
```

```
gg_arma(co2.ARIMA)
```




Accuracy metrics are generated for our two models based on in-sample data used for training and out-of-sample data used for testing. The training sets both exhibited a strong fit with low residual errors, however the test data shows significant inaccuracies for both the linear and SARIMA models.

```
rbind(co2.TSLM %>% accuracy(),
      forecast(co2.TSLM, h=278) %>% accuracy(co2_monthly_98),
      co2.ARIMA %>% accuracy(),
      forecast(co2.ARIMA, h=278) %>% accuracy(co2_monthly_98))
```

```
## # A tibble: 4 x 10
##   .model    .type        ME   RMSE    MAE      MPE   MAPE    MASE   RMSSE   ACF1
##   <chr>     <chr>     <dbl>  <dbl>  <dbl>    <dbl>  <dbl>   <dbl>   <dbl>  <dbl>
## 1 TSLM(val~ Trai~ -6.07e-16  0.497  0.385 -1.89e-4 0.114    0.304   0.353 0.821
## 2 TSLM(val~ Test   1.23e+ 1 15.2   12.3    3.07e+0 3.07   NaN     NaN     0.987
## 3 ARIMA(va~ Trai~  8.53e- 3  0.278  0.222  2.55e-3 0.0660   0.175   0.198 0.0313
## 4 ARIMA(va~ Test   6.16e+ 0  7.54   6.16   1.55e+0 1.55   NaN     NaN     0.984
```

## 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. This should include measurement and discussion how your model performs in-sample and (psuedo-) out-of-sample, comparing candidate models and explaining your choice. Generate predictions for when atmospheric CO2 is expected to be at 420 ppm and 500 ppm levels, considering prediction intervals as well as point estimates in your answer. Generate a prediction for atmospheric CO2 levels in the year 2100. How confident are you that these will be accurate predictions?

---

For our model comparisons we looked at three methods of using the training data set to explore candidate models:
1) Using the month average (from Part 4) data.
2) Using the most recent five years of weekly data.
3) Using the full weekly data.

Each of these were explored and a candidate model was selected from each based on AICc. After which we compare, the models using RMSE and the test data set.

The model developed using the month averaged data produced a model with the lowest RMSE. It also required much less run time than using the weekly data. And although it track very well with the first year of test data, it did less well on the second year. It also forecasted lower point estimate values. In addition the confidence intervals were much larger due to the higher RMSE.

For this model, the point prediction for Jan 2100 is 485 ppm with 95 confidence low of 404 ppm and and 95 high of 565 ppm. The point estimate for reaching 420 ppm CO2 is Jan 2056 and for 500 ppm is beyond 2100. However, the high 80 and 95 has 500 ppm beginning January 2084 and November 2076 respectively.

The model developed using five years of the weekly data produced a model with RMSE that was slightly higher RMSE than the model developed with month average data. This model tracked both the first and second year of test data fairly well. The forecasted point estimates are higher compared to the model from the month averaged data. For this model, Jan 2100 has CO2 concentration of 615 ppm with 95% confidence low of 537 ppm and and 95% confidence high of 694 ppm. The point estimate for reaching 420 ppm CO2 is March 2022 and for 500 ppm is May 2053.

The model developed using the full weekly data produced a substantially higher RMSE than the model developed with month average data and the model using the five years of weekly data. This model tracked both the first and second year of test data fairly well. The forecasted point estimates higher compared to the model from the month averaged data. The most estimates are similar to the previous weekly model; however, the confidence intervals are much wider.

In general, longer-term forecasts such as looking decades out would not be accurate using the models developed. The approximation of the model to the true process causes forecasting to be problematic as time frame increases. This appears to be especially true with the month average model.

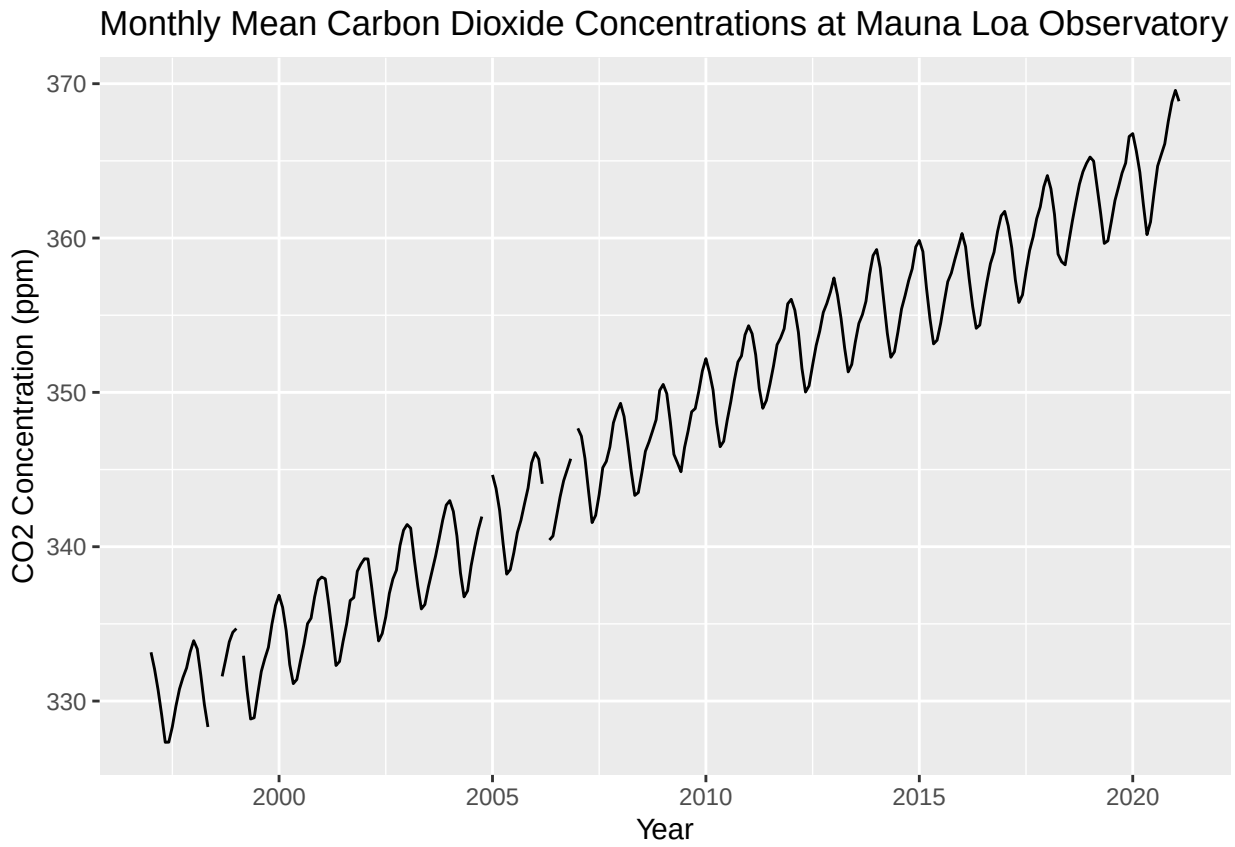
Output values, plots and discussion on the methodology and results of this exploration is provided below.

Plotting the data we see that using monthly averaging smooths out the bumps in the original (weekly) series. There are a few missing values in the first half of the series. These missing values are relatively few (seven NA out of a series with 290 observations) and the 290 observations span 24 years. In addition, they appear not be biased or intentional. Because our goal is not predicting/estimating the values of the missing observations, we leave them as NA because the ARIMA models are able to fit with the missing values.

```
df$ppm_w_na <- na_if(df$ppm, -999.99)
co2_mo_avg <- df %>%
  group_by(yr = year(date), mo = month(date)) %>%
  summarise(mo_avg = mean(ppm_w_na))
co2_mo_avg$day <- 1
co2_mo_avg$date <- as.Date(with(co2_mo_avg, paste(yr, mo, day, sep = '/')), '%Y/%m/%d')
co2_mo_avg_ts <- ts(co2_mo_avg$mo_avg, frequency = 12, start = c(1997, 1), end = c(2021, 2))
```

First, we will look at using the month average series. As mentioned earlier, the smoothing provided by the month averaging may result in a better model than the weekly series.

```
autoplot(co2_mo_avg_ts) +
  ggtitle('Monthly Mean Carbon Dioxide Concentrations at Mauna Loa Observatory') +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```

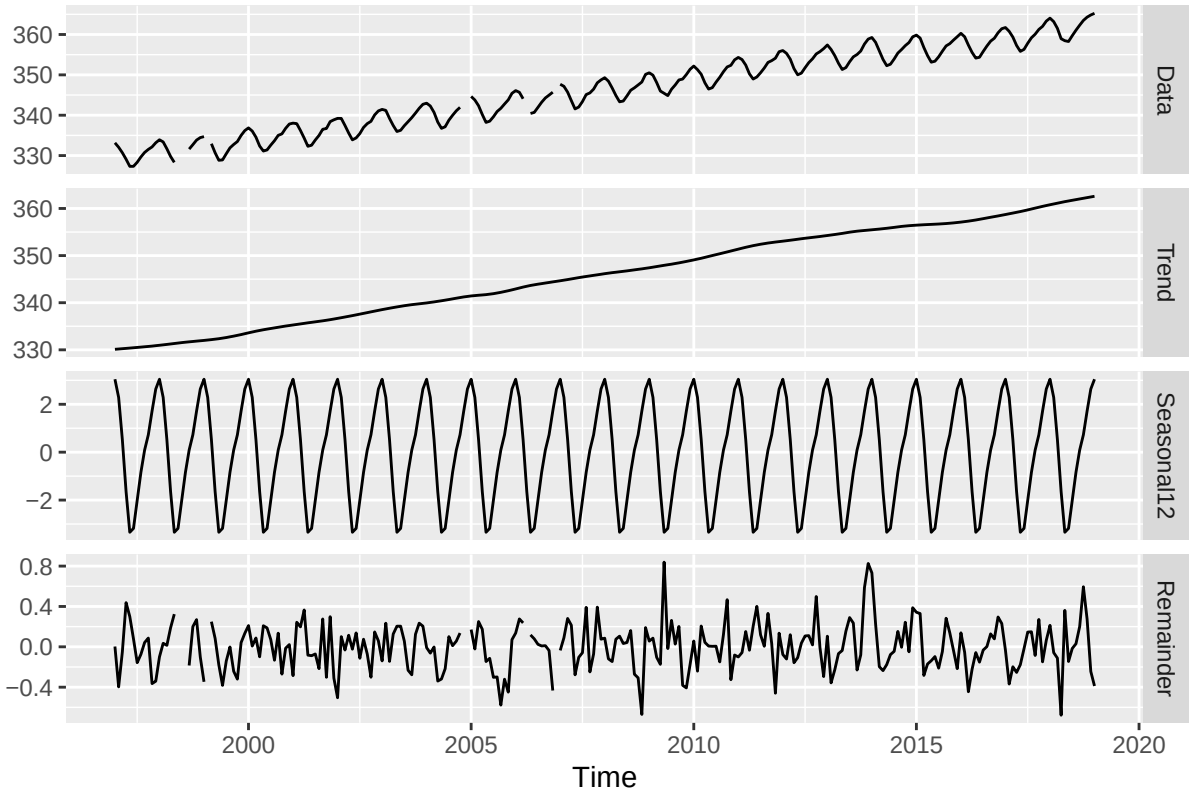



Split test and training (monthly avg) using the final two years of observations as the test set.

```
training <- subset(co2_mo_avg_ts, end=length(co2_mo_avg_ts)-25)
test<- subset(co2_mo_avg_ts, start=length(co2_mo_avg_ts)-24)
```

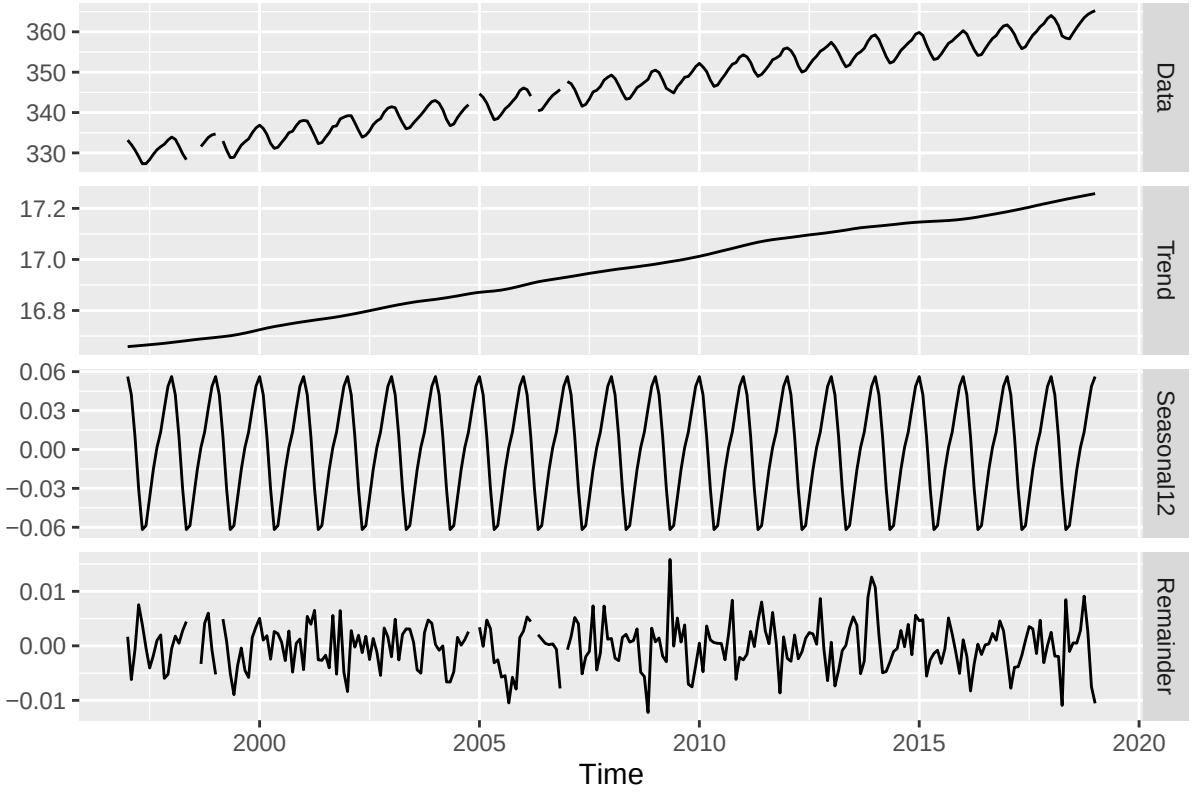
Reviewing the multi seasonal decomposition of the training set there is clearly a trend and seasonal component to the data. A log transformation does not appear necessary as the variance over time appears to be stable.

```
training %>%
  mstl(s.window="periodic", robust = TRUE) %>%
  autoplot()
```



As a check we will compare the decomposition below that has a box cox transformation with the previous decomposition that does not. THe decomposed components appear nearly identical so It doesn't appear that that a box cox transformation is necessary or helpful.

```
training %>%
  mstl(s.window="periodic", robust = TRUE, lambda= "auto") %>%
  autoplot()
```




KPSS test to determine seasonal differencing suggests one seasonal difference.

```
training %>% nsdiffs()
```

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

Time series for co2 measured over each month; therefore difference using seasonal period of 12 months. Then perform KPSS test to determine number of first differences required. This suggest no first difference is required.

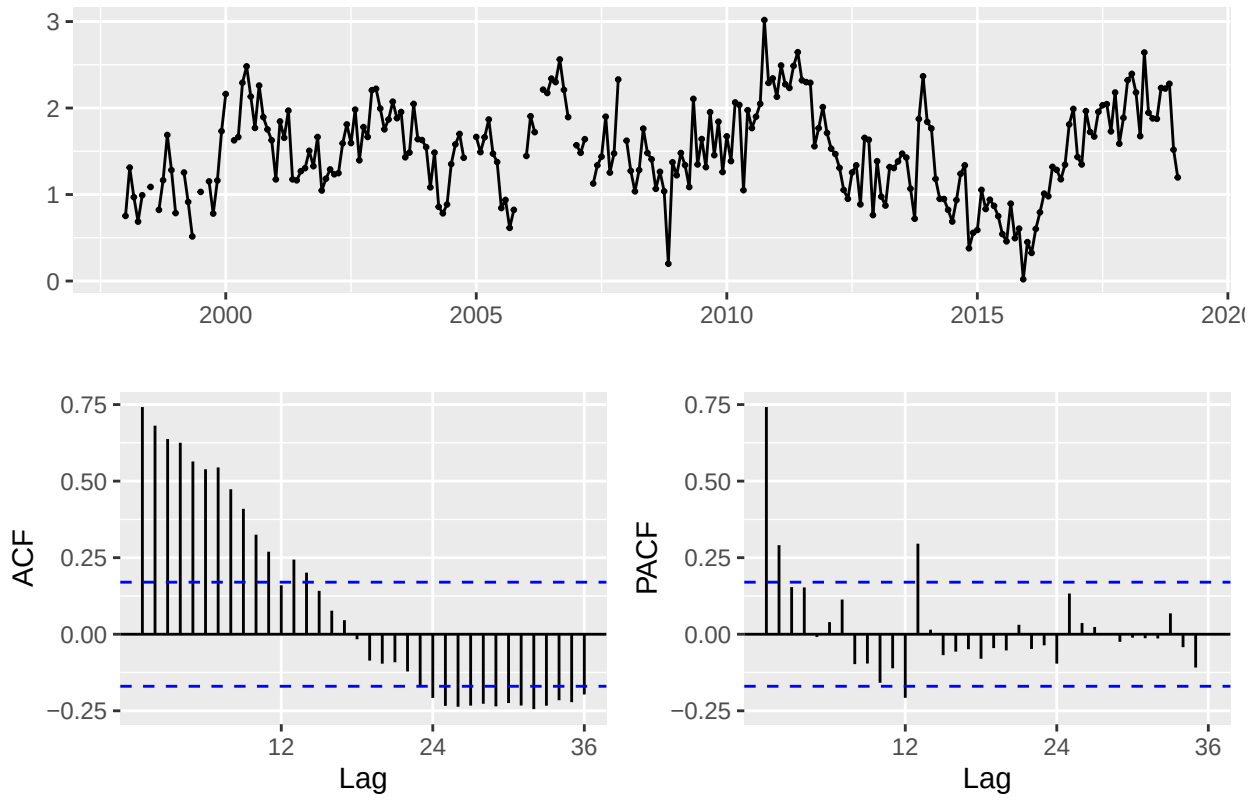
```
training %>% diff(lag=12) %>% ndiffs
```

```
## [1] 0
```

With only taking a seasonal difference, the series appears nonstationary as shown in time plot below. This suggests taking a first difference.

```
training_d <- training %>% diff(lag =12) %>% ggtsdisplay()
```

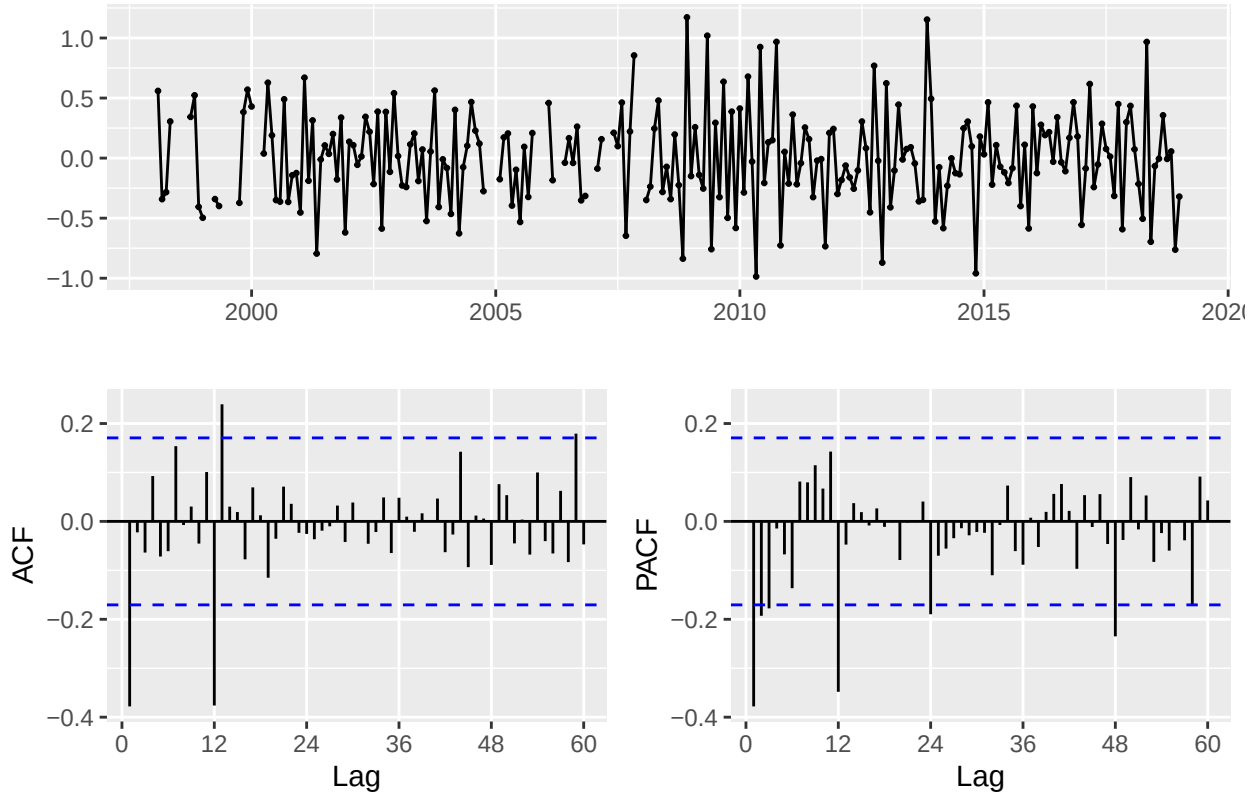
```
## Warning: Removed 14 rows containing missing values (geom_point).
```



With seasonal difference and first difference, the series appears to be stationary. PACF has spikes at seasonal lags 12 and 24, and ACF has a very strong peak at lag 12. ACF has a significant spike at lag 1 and none until lag 12. PACF has three significant spikes at the beginning. The exact ARIMA is not clear based on the plots. However, they suggest starting with something like: ARIMA$(3,1,1)(0,1,1)_{12}$ and many variations on it.

```
training_d <- training %>% diff(lag = 12) %>% diff()
training_d  %>% ggtsdisplay(lag.max = 60)
```

```
## Warning: Removed 26 rows containing missing values (geom_point).
```




Loop through various ARIMA models. Look at AICc, BIC and RMSE as selection criteria.

```
results2 <- data.frame(p=integer(),
                       q=integer(),
                       P=integer(),
                       Q=integer(),
                       AICc=double(),
                       BIC=double(),
                       RMSE=double())
for (p in 0:3){
  for (q in 0:3){
```

```
    for(P in 0:3){
      for (Q in 0:3){
        tryCatch(
          {
            mod2 <- training %>% as_tsibble() %>%
              model(ARIMA(value ~ 0 + pdq(p,1,q) + PDQ(P,1,Q)))
            if(has_name(glance(mod2),'AICc')){
          }
        results2 <- results2 %>% add_row(p=p, q = q, P=P, Q=Q, AICc = as.numeric(glance(mod2)$A
        print(paste(p, q, P, Q, as.numeric(glance(mod2)$AICc), as.numeric(glance(mod2)$BIC), a
          },
        error=function(e) {
          print(paste('error encountered for', p, q, P, Q))
          }
        )
      }
    }
  }
}
```

Review best model statistics:

```
rbind(results2[which.min(results2$AICc), ],
  results2[which.min(results2$BIC), ],
  results2[which.min(results2$RMSE), ])
```

```
##     p q P Q     AICc      BIC      RMSE
## 32  0 1 3 3 100.0846 127.7274 0.2655362
## 18  0 1 0 1 107.4695 117.9610 0.2896930
## 166 2 3 3 3 100.9634 142.0112 0.2615774
```

Of the models produced, smallest AICc is with ARIMA$(0,1,1)(2,1,4)_{12}$. We will use RMSE and also out of sample results to compare this result with models produced from the weekly series.

```
fit.011.214 <-Arima(training, order=c(0,1,1), seasonal = c(2,1,4))
```

Our ARIMA$(0,1,1)(2,1,4)_{12}$ model has an AICc $= 99.34$, BIC $= 127.6$ and RMSE $= 0.266$.

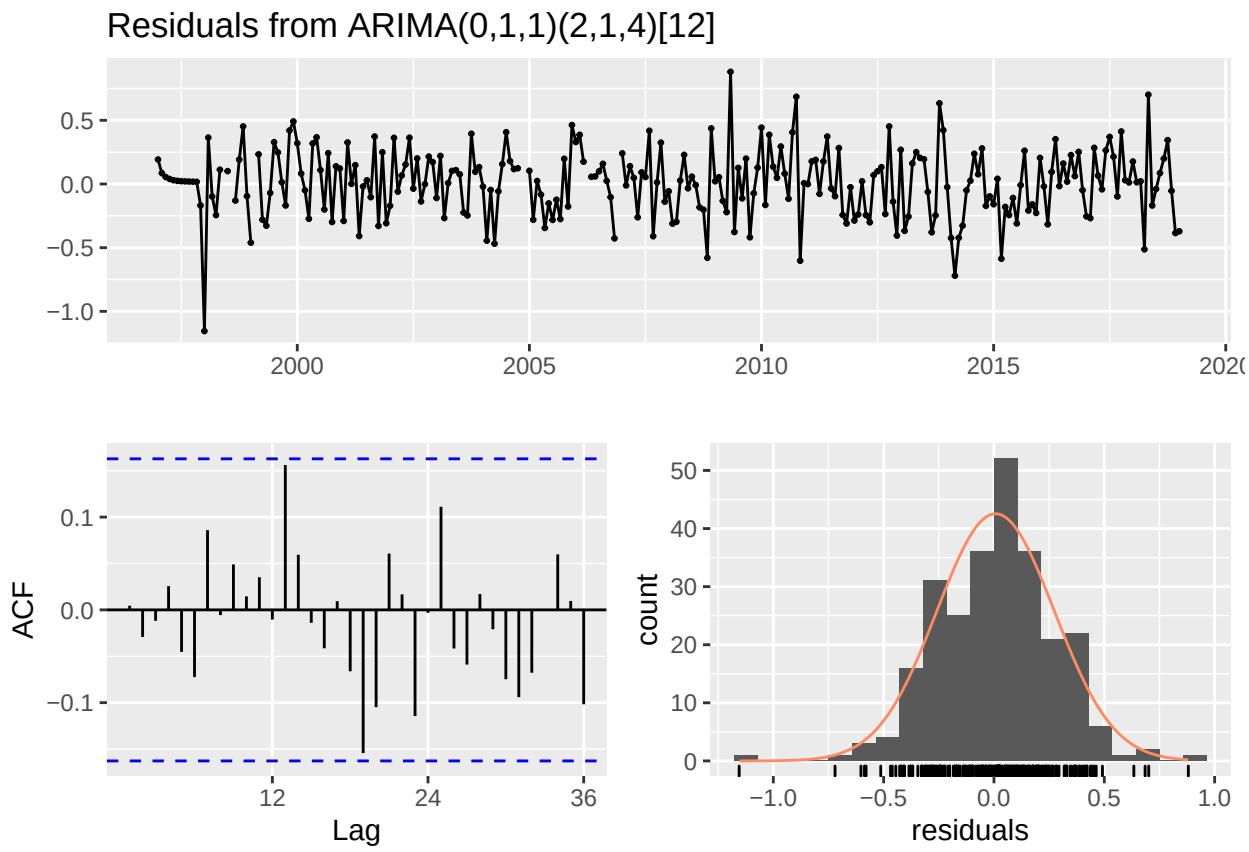
```
summary(fit.011.214)
```

```
## Series: training
## ARIMA(0,1,1)(2,1,4)[12]
##
## Coefficients:
##           ma1     sar1     sar2     sma1     sma2     sma3     sma4
```

```
##       -0.4334  -1.0889  -0.9881  0.3930  0.1378  -0.8456  -0.1960
## s.e.   0.0621   0.0552   0.0190  0.1055  0.1063   0.1009   0.0852
##
## sigma^2 estimated as 0.07662:  log likelihood=-41.67
## AIC=99.34   AICc=99.94   BIC=127.58
##
## Training set error measures:
##                       ME      RMSE       MAE         MPE       MAPE      MASE
## Training set 0.007359859 0.2658666 0.2067262 0.002199897 0.05964617 0.1380395
##                     ACF1
## Training set 0.01450724
```

The residuals provide means to see the difference between the fitted and observed values. The plots show that the model has adequately captured the information in the data. The residuals have no significant ACF and look like white noise with mean approximately zero. In addition, the residual have reasonably constant variance and fairly normally distributed.

```
checkresiduals(fit.011.214)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)(2,1,4)[12]
## Q* = 19.547, df = 17, p-value = 0.298
##
## Model df: 7.   Total lags used: 24
```

Both the Box Pierce test and Box Ljung test for this model have relatively large p-values, so we can conclude that the residuals are not distinguishable from white noise.

```
Box.test(residuals(fit.011.214))
```

```
##
##  Box-Pierce test
```

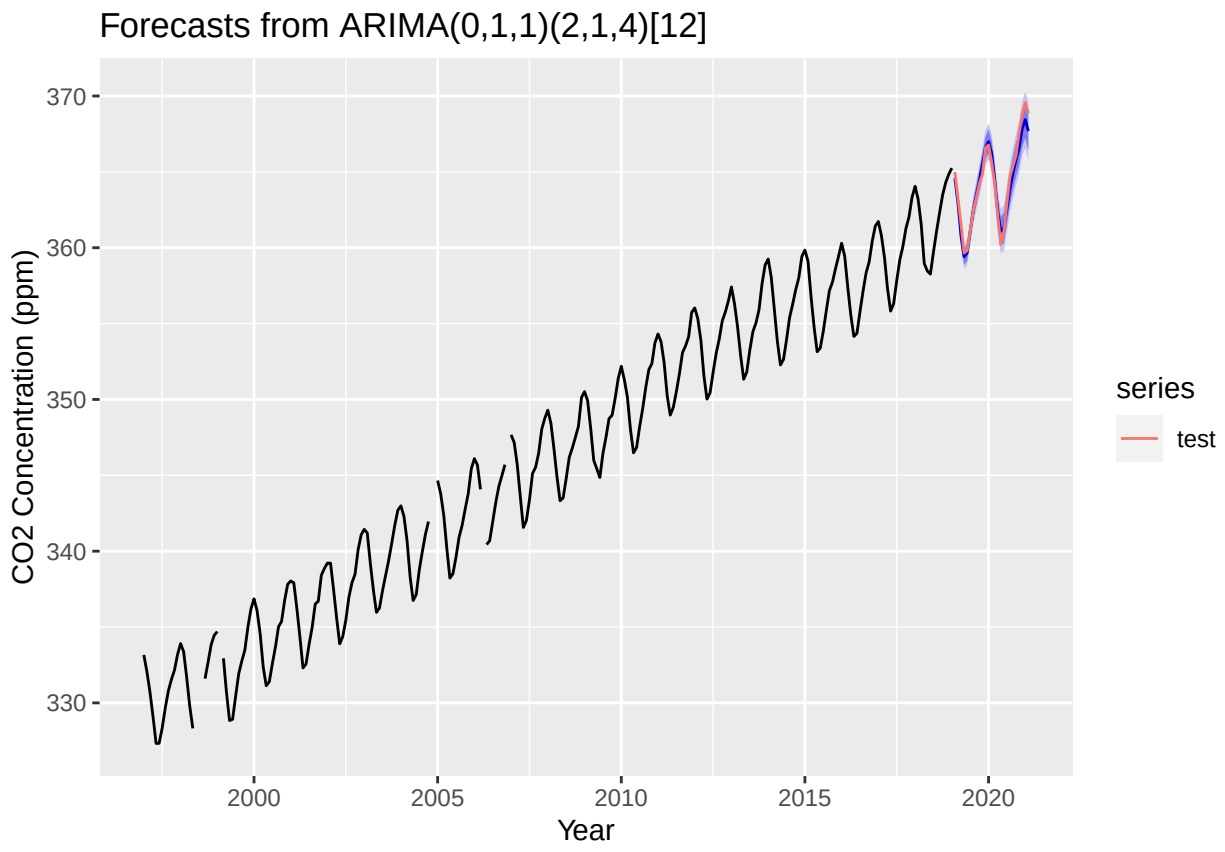
```
##
## data:  residuals(fit.011.214)
## X-squared = 0.054299, df = 1, p-value = 0.8157
```

```
Box.test(residuals(fit.011.214), type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  residuals(fit.011.214)
## X-squared = 0.054933, df = 1, p-value = 0.8147
```

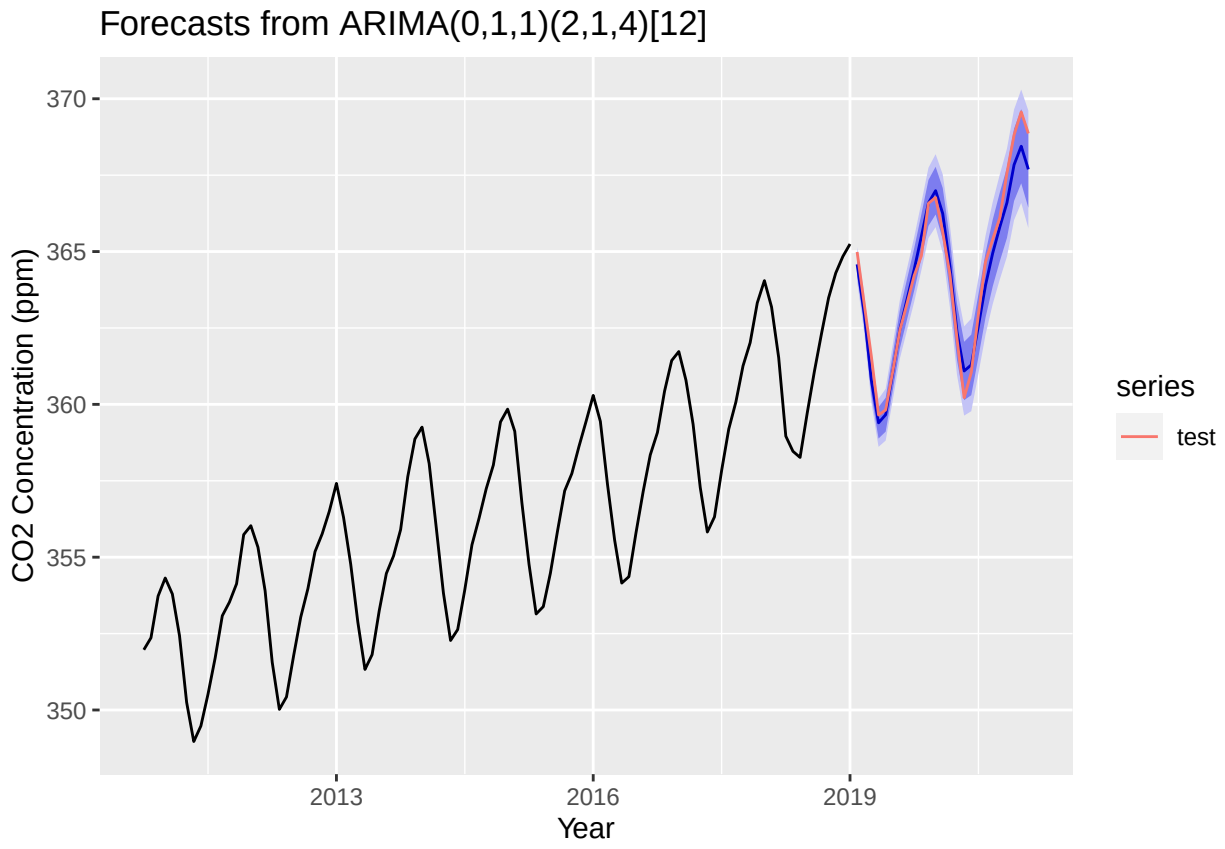
Plot of the entire monthly average series (training and test) with the forecast is shown below.

```
fit.011.214 %>%
  forecast(h=25) %>%
  autoplot() + autolayer(test) +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```



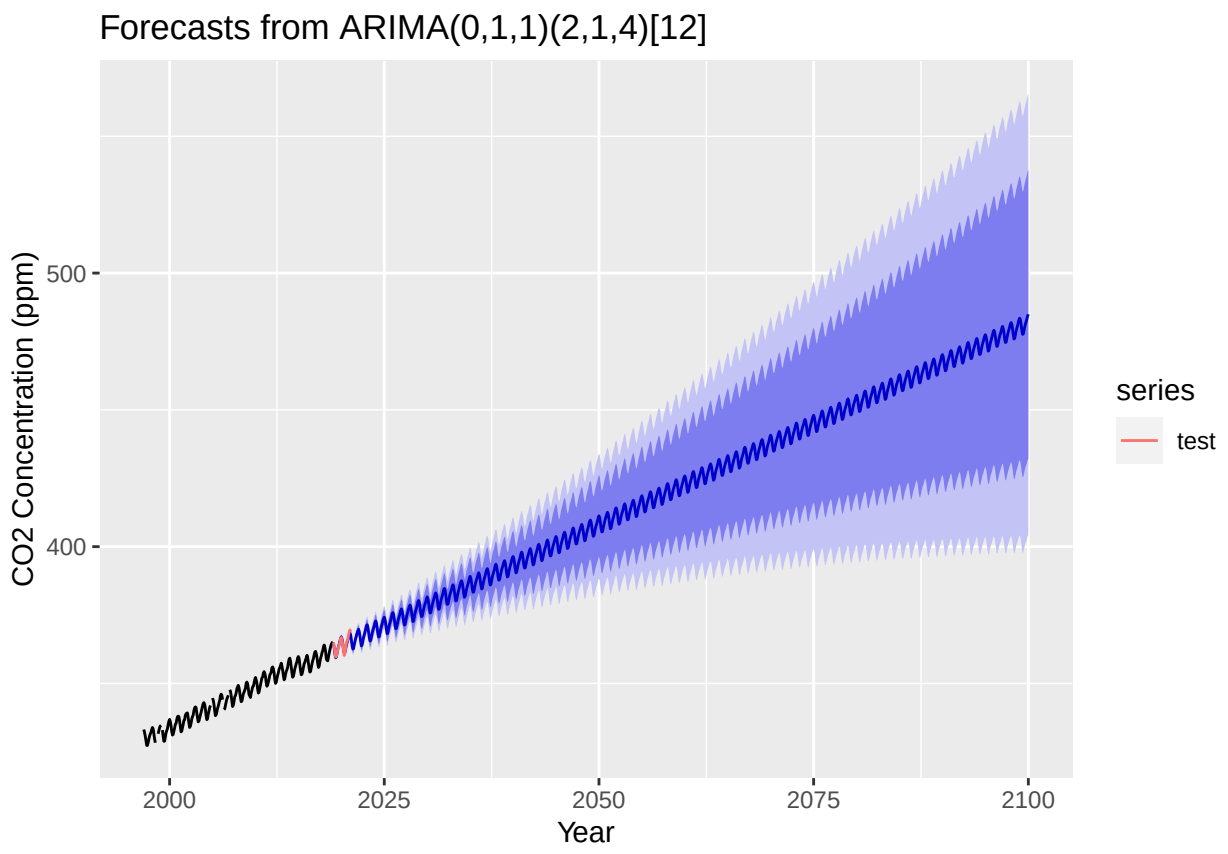
Zooming in we see that the forecast follows the test set very closely for one year out but forecasts a noticeably lower value for the following year.

```
fit.011.214 %>%
  forecast(h=25) %>%
  autoplot(include=100) + autolayer(test) +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```



The ARIMA$(0,1,1)(2,1,4)_{12}$ point prediction for Jan 2100 is 485 ppm with 95 confidence low of 404 ppm and and 95 high of 565 ppm. The point estimate for reaching 420 ppm CO2 is Jan 2056 and for 500 ppm is beyond 2100. However, the high 80 and 95 has 500 ppm beginning January 2084 and November 2076 respectively. Forecasting to 2100 we see that the confidence interval bands get wider as we move further out from our data. As shown in the comparison of the two year forecast and the test set, our model appears quite good at predicting the next year as the predictions are very tight with the test data. However, longer-term forecasts such as looking decades out would be not be accurate because the approximation of the model to the true process forecasting is problematic as time frame increases.

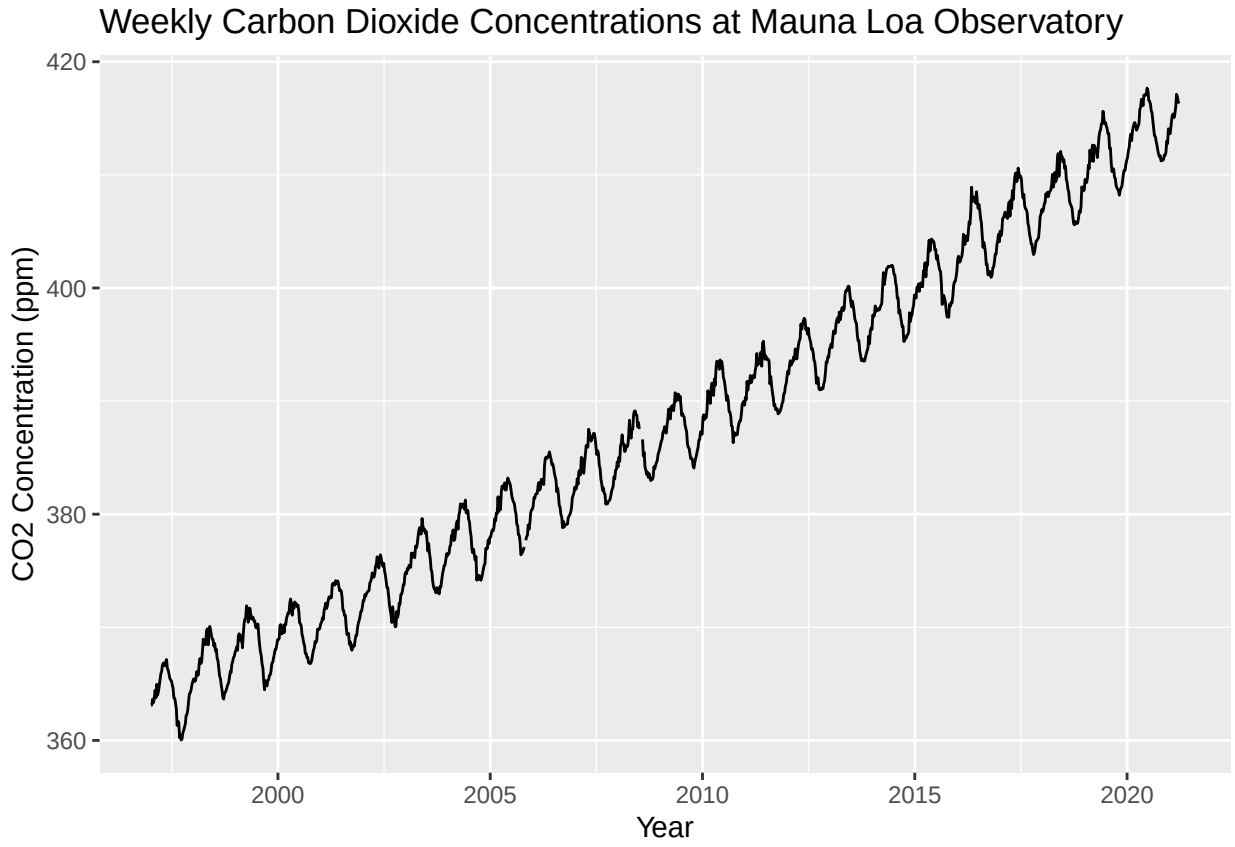
```
fit.011.214 %>%
  forecast(h=972) %>%
  autoplot() + autolayer(test) +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```




```
mo_avg_forecast <- data.frame(forecast(fit.011.214, h=972))
```

***Now we will consider modeling with weekly data.***

```
df$date <- as.Date(with(df, paste(yr, mo, day, sep = '/')), '%Y/%m/%d')
co2_weekly <-tsibble(week = yearweek(df$date), Observation = df$ppm, index = week)
co2_97_21 <- co2_weekly %>% filter_index('1997 W01'~.)
co2_97_21_ts <- ts(co2_97_21$Observation, frequency = 52,
                   start = decimal_date(ymd('1997-1-5')))

autoplot(co2_97_21_ts) +
  ggtitle('Weekly Carbon Dioxide Concentrations at Mauna Loa Observatory') +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```

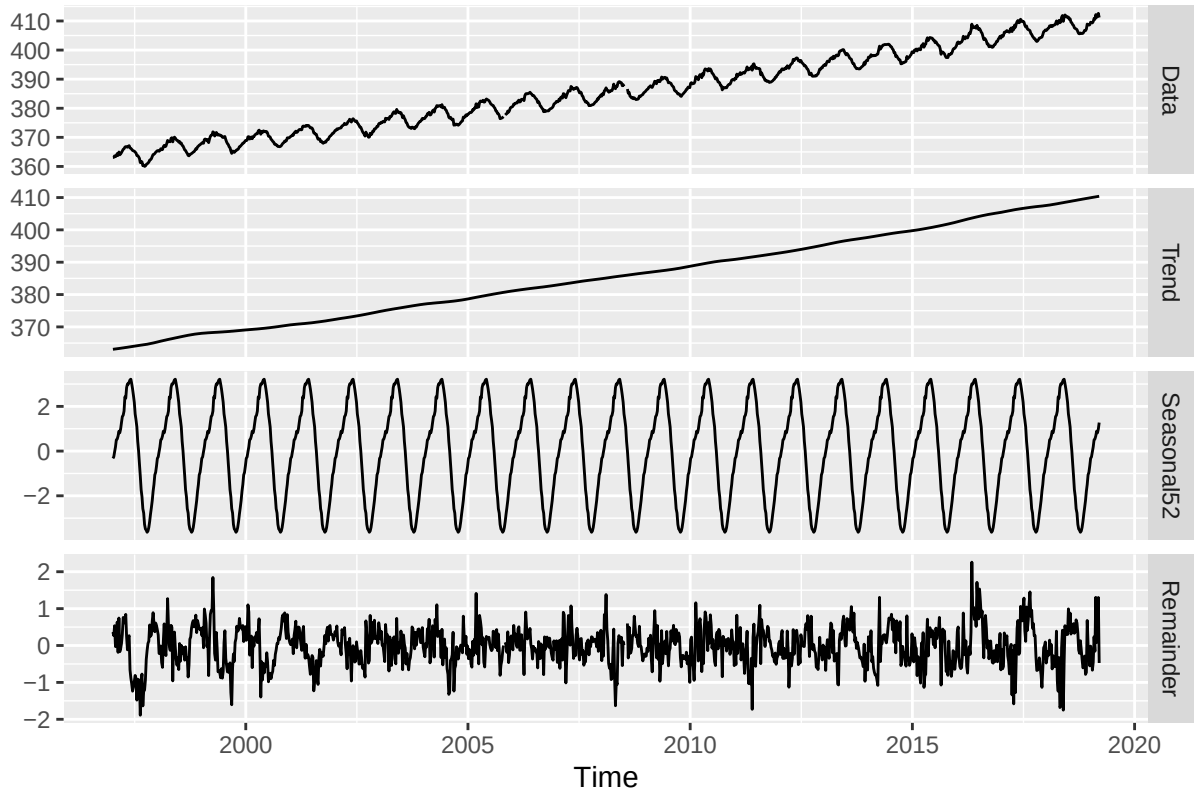



Split test and training (weekly) using the final two years of observations as the test set.

```
training_wk <- subset(co2_97_21_ts, end=length(co2_97_21_ts)-105, frequency = 365.25/7)
test_wk <- subset(co2_97_21_ts, start=length(co2_97_21_ts)-104, frequency = 365.25/7)

training_wk_52 <- subset(co2_97_21_ts, end=length(co2_97_21_ts)-105, frequency = 52)
test_wk_52 <- subset(co2_97_21_ts, start=length(co2_97_21_ts)-104, frequency = 52)
```

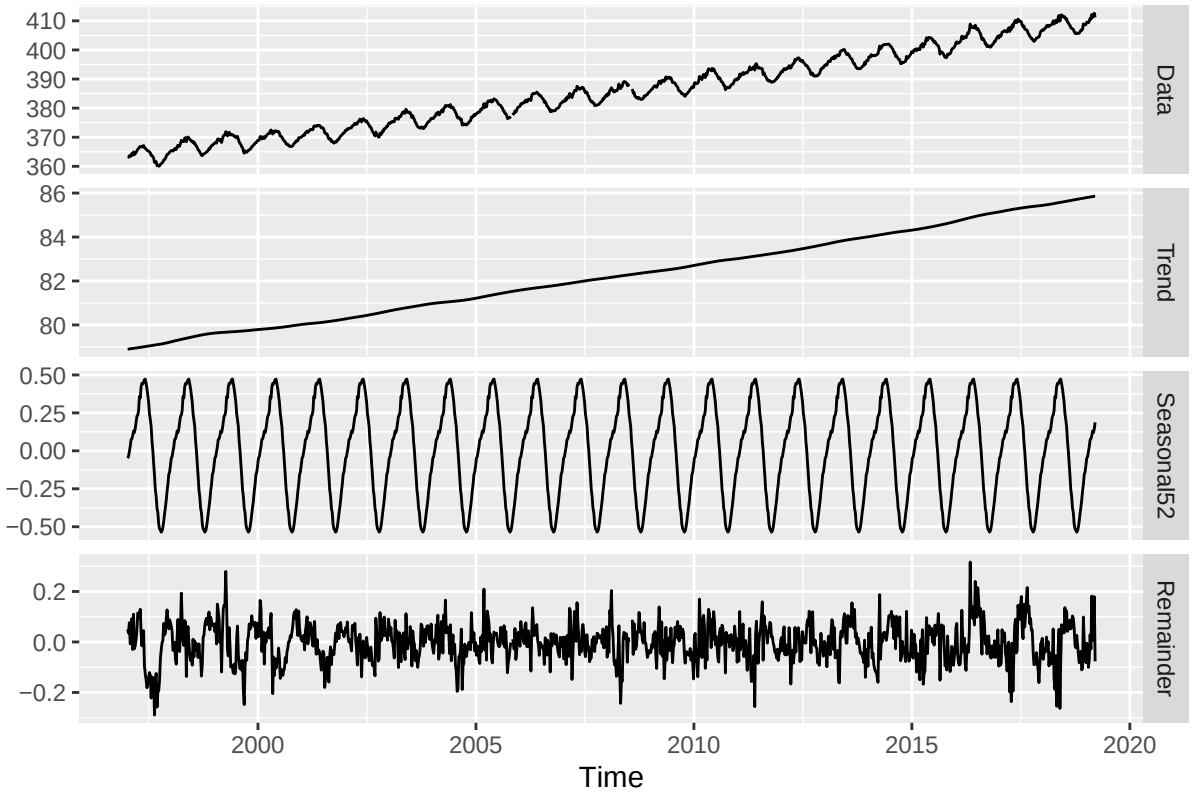
Reviewing the multi seasonal decomposition of the training set there is clearly a trend and seasonal component to the data. A log transformation does not appear necessary as the variance over time appears to be stable.

```
training_wk_52 %>%
  mstl(s.window="periodic", robust = TRUE) %>%
  autoplot()
```



A decomposition with a box cox transformation confirms that box cox transformation is not needed as there is not a difference in the results.

```
training_wk_52 %>%
  mstl(s.window="periodic", robust = TRUE, lambda = 'auto') %>%
  autoplot()
```




KPSS test to determine seasonal differencing suggests one seasonal difference.

```
training_wk %>% nsdiffs()
```

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

Time series for co2 measured over each month; therefore difference using seasonal period of 12 months. Then perform KPSS test to determine number of first differences required. This suggest no first difference is required.

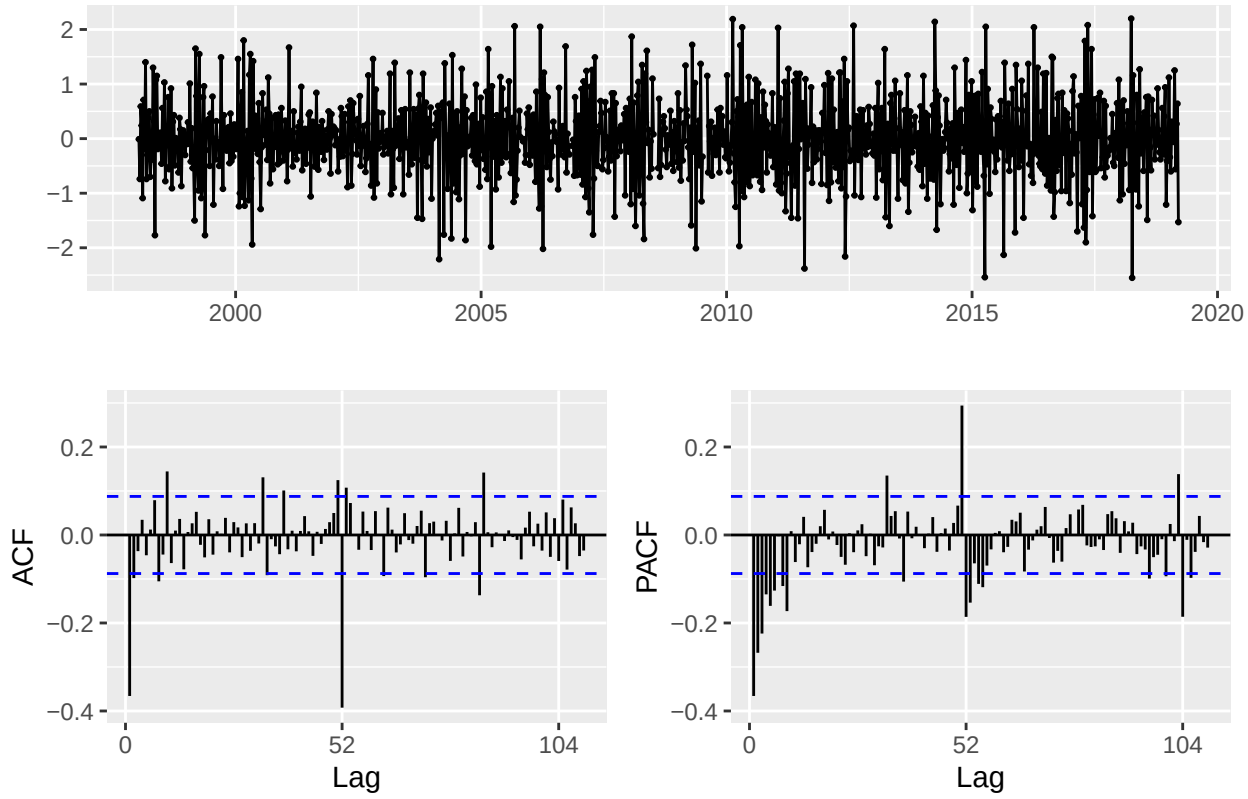
```
training_wk %>% diff(lag=52) %>% ndiffs
```

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

With seasonal difference and first difference, the series appears to be stationary. PACF has spikes at seasonal lags 52 and 104, and ACF has a very strong peak at lag 52. ACF has a significant spike at lag 1 and lag 52 suggests seasonal MA(1) component. PACF has several significant spikes at the beginning. The exact ARIMA is not clear based on the plots. However, they suggest starting with something like: ARIMA$(0,1,1)(0,1,1)_{52}$ and many variations on it.

```
training_wk_d <- training_wk %>% diff(lag = 52) %>% diff()
training_wk_d  %>% ggtsdisplay(lag.max = 110)
```

```
## Warning: Removed 12 rows containing missing values (geom_point).
```




Loop through various ARIMA models. Trend and seasonal pattern in the data is strong, therefore it seems reasonable to reduce size of training set to the last five years in order to evaluate models in a more reasonable timeframe. We will use AICc, BIC and RMSE as selection criteria.

Look at most recent five years worth of training data

```
training_wk_5yrs <- subset(training_wk, start=length(training_wk)-260)
```

```
results3 <- data.frame(p=integer(),
                       q=integer(),
                       P=integer(),
```

```
                      Q=integer(),
                      AICc=double(),
                      BIC=double(),
                      RMSE=double())
for (p in 0:3){
  for (q in 0:3){
    for(P in 0:3){
      for (Q in 0:3){
        tryCatch(
          {
            mod3 <- training_wk_5yrs %>% as_tsibble() %>%
              model(ARIMA(value ~ 0 + pdq(p,1,q) + PDQ(P,1,Q)))
            if(has_name(glance(mod3),'AICc')){
          }
        results3 <- results3 %>%
          add_row(p=p, q = q, P=P, Q=Q,
                  AICc = as.numeric(glance(mod3)$AICc),
                  BIC = as.numeric(glance(mod3)$BIC),
                  RMSE = accuracy(mod3)$RMSE)
        print(paste(p, q, P, Q,
                    as.numeric(glance(mod3)$AICc),
                    as.numeric(glance(mod3)$BIC),
                    accuracy(mod3)$RMSE))
          },
        error=function(e) {
          print(paste('error encountered for', p, q, P, Q))
          }
        )
      }
    }
  }
}
```

Review best model diagnostics:

```
rbind(results3[which.min(results3$AICc), ],
  results3[which.min(results3$BIC), ],
  results3[which.min(results3$RMSE), ])
```

```
##     p q P Q     AICc      BIC      RMSE
## 43  2 3 1 1 398.8293 424.8060 0.4938308
## 6   0 1 0 1 402.9974 412.8924 0.5557537
## 431 2 3 1 1 398.8293 424.8060 0.4938308
```

Of the models produced, smallest AICc is with ARIMA$(2,1,3)(1,1,1)_{52}$

```
fit.213.111 <-Arima(training_wk_5yrs, order=c(2,1,3), seasonal = c(1,1,1))
```

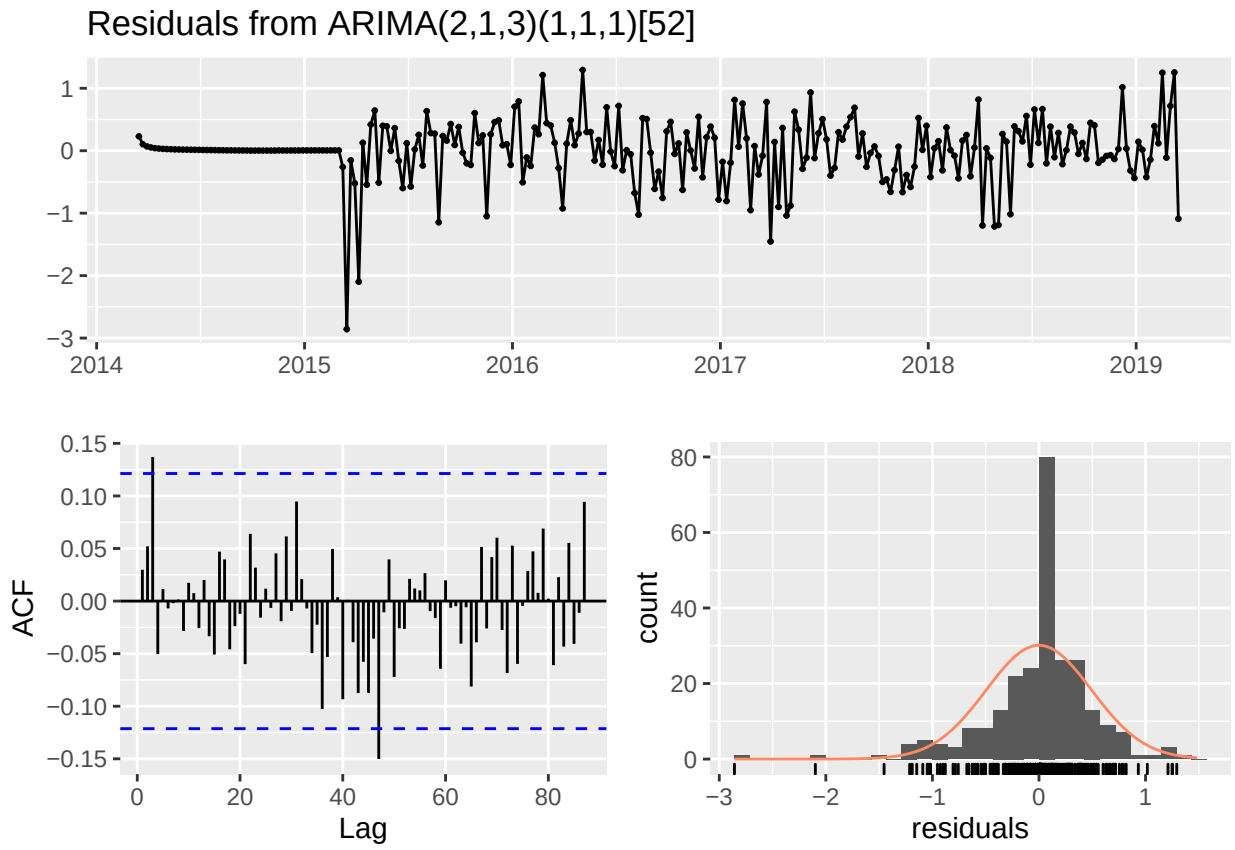
Our ARIMA$(2,1,3)(1,1,1)_{52}$ model has an AICc = 398.83, BIC = 424.81 and RMSE = 0.49. The RMSE value for the model developed using the month average was slightly lower with an RMSE = 0.266.

```
summary(fit.213.111)
```

```
## Series: training_wk_5yrs
## ARIMA(2,1,3)(1,1,1)[52]
##
## Coefficients:
##           ar1      ar2     ma1      ma2      ma3    sar1     sma1
##       -1.0011  -0.7952  0.3657  -0.0243  -0.7227  0.1832  -0.9860
## s.e.   0.1251   0.0746  0.1120   0.0759   0.0633  0.1015   0.4276
##
## sigma^2 estimated as 0.3167:  log likelihood=-191.05
## AIC=398.11   AICc=398.83   BIC=424.81
##
## Training set error measures:
##                         ME      RMSE       MAE           MPE       MAPE
## Training set -0.0005431678 0.4938308 0.3283543 -0.0002768606 0.08096009
##                    MASE       ACF1
## Training set 0.1320443 0.02979059
```

The residuals plot of ARIMA$(2,1,3)(1,1,1)_{52}$ appears somewhat like white noise except for the outliers in the beginning. The ACF plot has a couple of significant values and the residuals have a suboptimal normal looking distribution. Overall, the diagnostic plots do not appear as compelling as the results from the model generated from monthly average series, but they are not terrible either.

```
checkresiduals(fit.213.111)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,3)(1,1,1)[52]
## Q* = 42.618, df = 45, p-value = 0.5734
##
## Model df: 7.   Total lags used: 52
```

Both the Box Pierce test and Box Ljung test for this model have relatively large p-values, so we can conclude that the residuals are not distinguishable from white noise.

```
Box.test(residuals(fit.213.111))
```

```
##
```

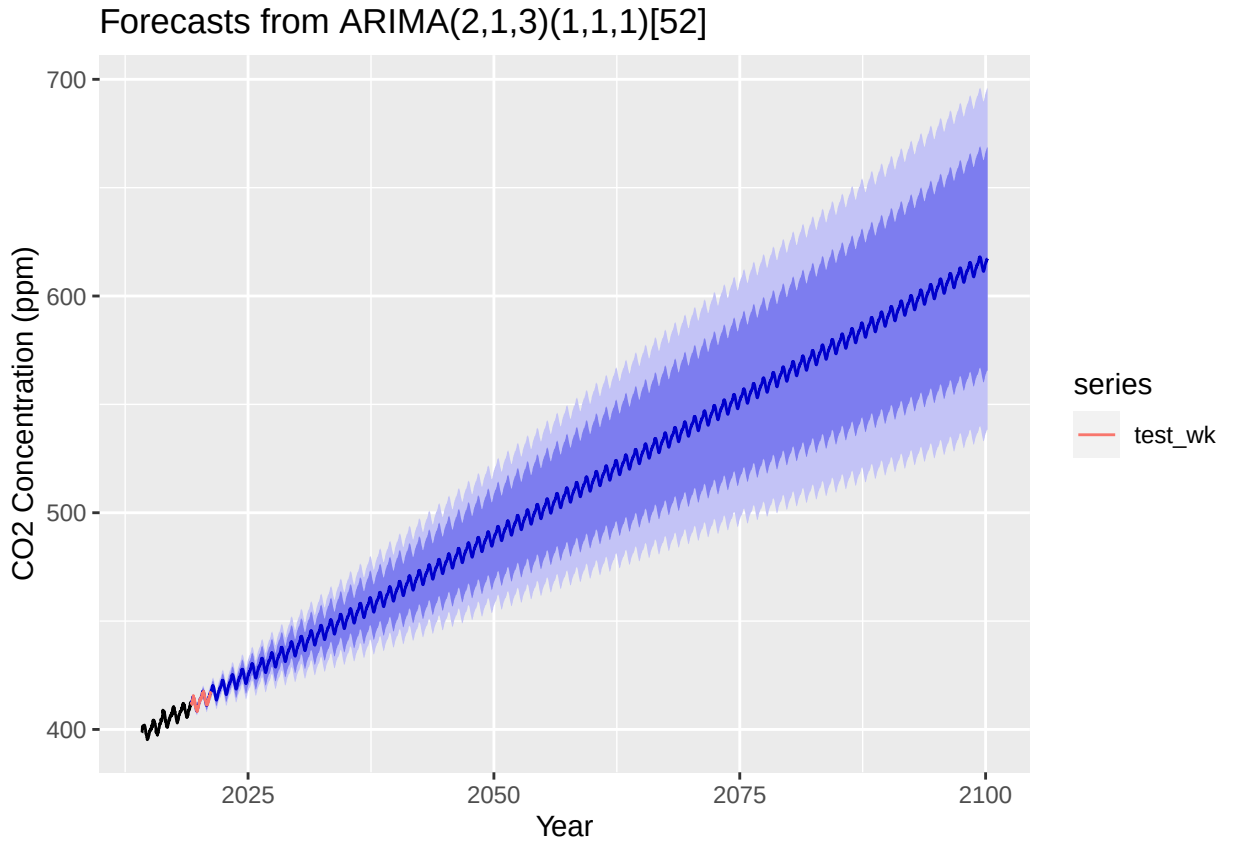
```
##  Box-Pierce test
##
## data:  residuals(fit.213.111)
## X-squared = 0.23163, df = 1, p-value = 0.6303
```

```
Box.test(residuals(fit.213.111), type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  residuals(fit.213.111)
## X-squared = 0.2343, df = 1, p-value = 0.6284
```

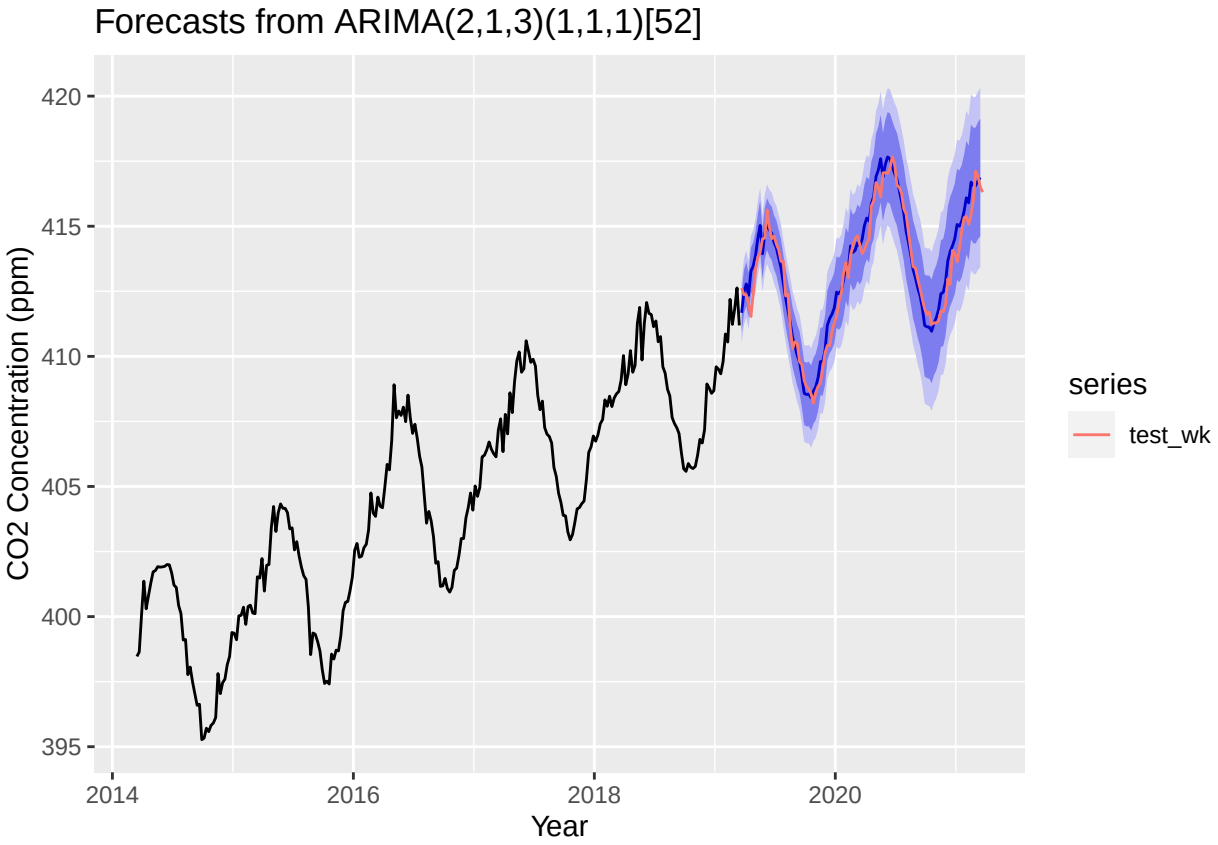
The ARIMA$(2,1,3)(1,1,1)_{52}$ point prediction for Jan 2100 is 615 ppm with 95 confidence low of 537 ppm and and 95 high of 694 ppm.
The point estimate for reaching 420 ppm CO2 is March 2022 and for 500 ppm is May 2053.

```
fit.213.111 %>%
  forecast(h=4212) %>%
  autoplot() + autolayer(test_wk) +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```



This model's forecast for the following two years appears to track the test data for both years. This is unlike our model developed with the monthly average data which tracked very well on the first year but not so much on the following year.

```
fit.213.111 %>%
  forecast(h=104) %>%
  autoplot(include=500) + autolayer(test_wk) +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```




```
weekly_forecast <-  data.frame(forecast(fit.213.111, h=4212))
```

```
weekly_forecast$dates <- date_decimal(as.numeric(rownames(weekly_forecast)))
weekly_forecast$year <- format(weekly_forecast$dates, '%Y')
weekly_forecast$week <- format(weekly_forecast$dates, '%W')
```

Next, we will consider using the entire weekly data set to develop a model.

```
results4 <- data.frame(p=integer(),
                       q=integer(),
                       P=integer(),
```

```
                      Q=integer(),
                      AICc=double(),
                      BIC=double(),
                      RMSE=double())
for (p in 0:3){
  for (q in 0:3){
    for(P in 0:3){
      for (Q in 0:3){
        tryCatch(
          {
            mod4 <- training_wk %>% as_tsibble() %>%
              model(ARIMA(value ~ 0 + pdq(p,1,q) + PDQ(P,1,Q)))
            if(has_name(glance(mod4),'AICc')){
          }
        results4 <- results4 %>% add_row(p=p, q = q, P=P, Q=Q, AICc = as.numeric(glance(mod4)$A
        print(paste(p, q, P, Q, as.numeric(glance(mod4)$AICc), as.numeric(glance(mod4)$BIC), a
          },
        error=function(e) {
          print(paste('error encountered for', p, q, P, Q))
          }
        )
      }
    }
  }
}
```

Examine models with best statistics:

```
rbind(results4[which.min(results4$AICc), ],
  results4[which.min(results4$BIC), ],
  results4[which.min(results4$RMSE), ])
```

```
##     p q P Q     AICc      BIC     RMSE
## 63  0 3 3 3 12641.40 12691.25 64.60002
## 49  0 3 0 1 12646.06 12671.03 67.67965
## 631 0 3 3 3 12641.40 12691.25 64.60002
```

Of the models produced, smallest AICc is with ARIMA$(0,1,3)(3,1,3)_{52}$

```
fit.013.313 <-Arima(training_wk, order=c(0,1,3), seasonal = c(3,1,3))
```

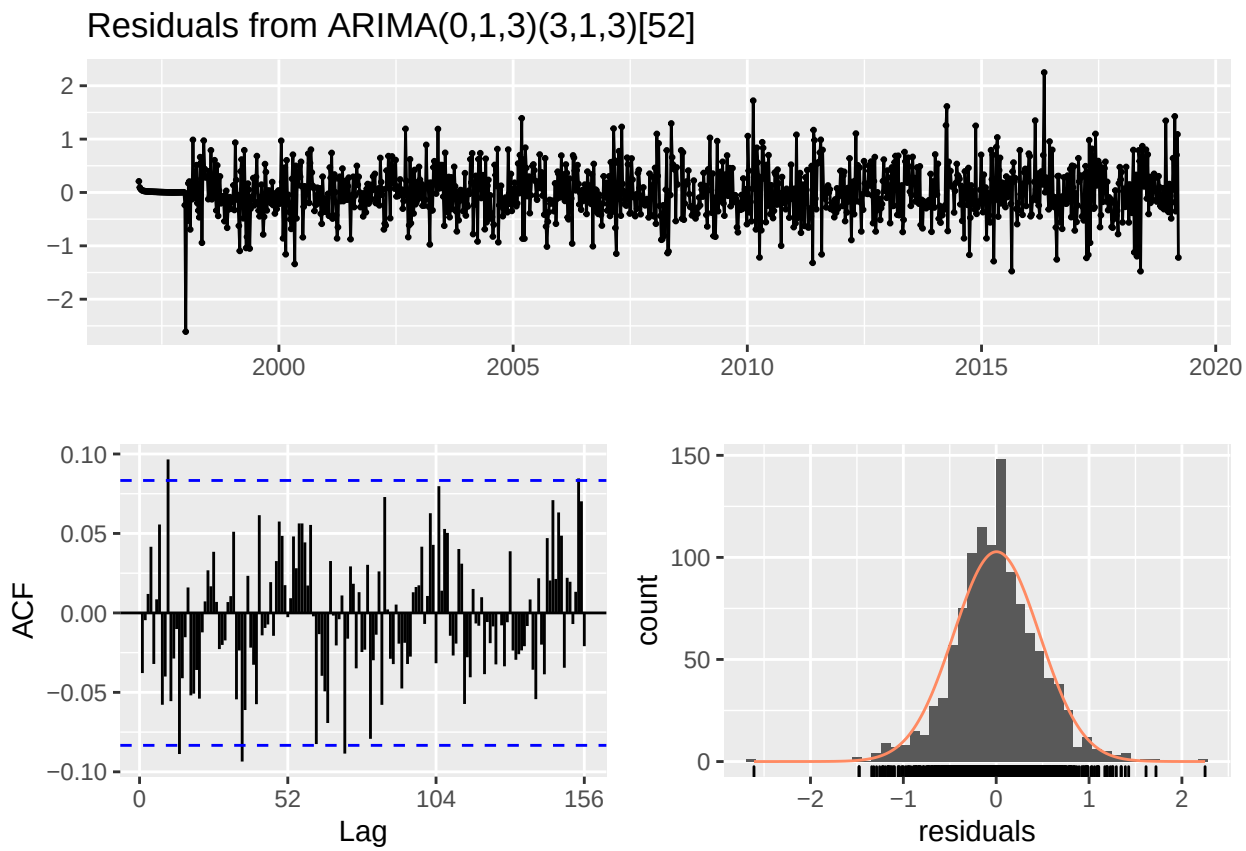
Our ARIMA$(0,1,3)(3,1,3)_{52}$ model has an AICc = 12641, BIC = 12691 and RMSE = 64.8. The RMSE value for the model developed using the monthly average and also the most recent 5 years of weekly data was much lower with an RMSE = 0.266 and RMSE = 0.493 respectively.

```
summary(fit.013.313)
```

```
## Series: training_wk
## ARIMA(0,1,3)(3,1,3)[52]
##
## Coefficients:
##           ma1      ma2      ma3     sar1    sar2     sar3     sma1     sma2
##       -0.5821  -0.1304  -0.0283  -0.2599  0.0919  -0.0681  -0.5990  -0.3381
## s.e.   0.0312   0.0336   0.0305   0.4383  0.3485   0.0468   0.4345   0.8908
##         sma3
##       0.2024
## s.e.  0.3431
##
## sigma^2 estimated as 0.2268:  log likelihood=-754.03
## AIC=1528.06   AICc=1528.26   BIC=1578.11
##
## Training set error measures:
##                       ME      RMSE       MAE          MPE       MAPE      MASE
## Training set 0.003811809 0.4632007 0.3476351 0.0006498722 0.08973413 0.1623777
##                       ACF1
## Training set -0.0002174351
```

The residual plot appears to look like white noise

```
checkresiduals(fit.013.313)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,3)(3,1,3)[52]
## Q* = 130.22, df = 95, p-value = 0.009616
##
## Model df: 9.   Total lags used: 104
```

Both the Box Pierce test and Box Ljung test for this model have relatively large p-values, so we can conclude that the residuals are not distinguishable from white noise.

```
Box.test(residuals(fit.013.313))
```

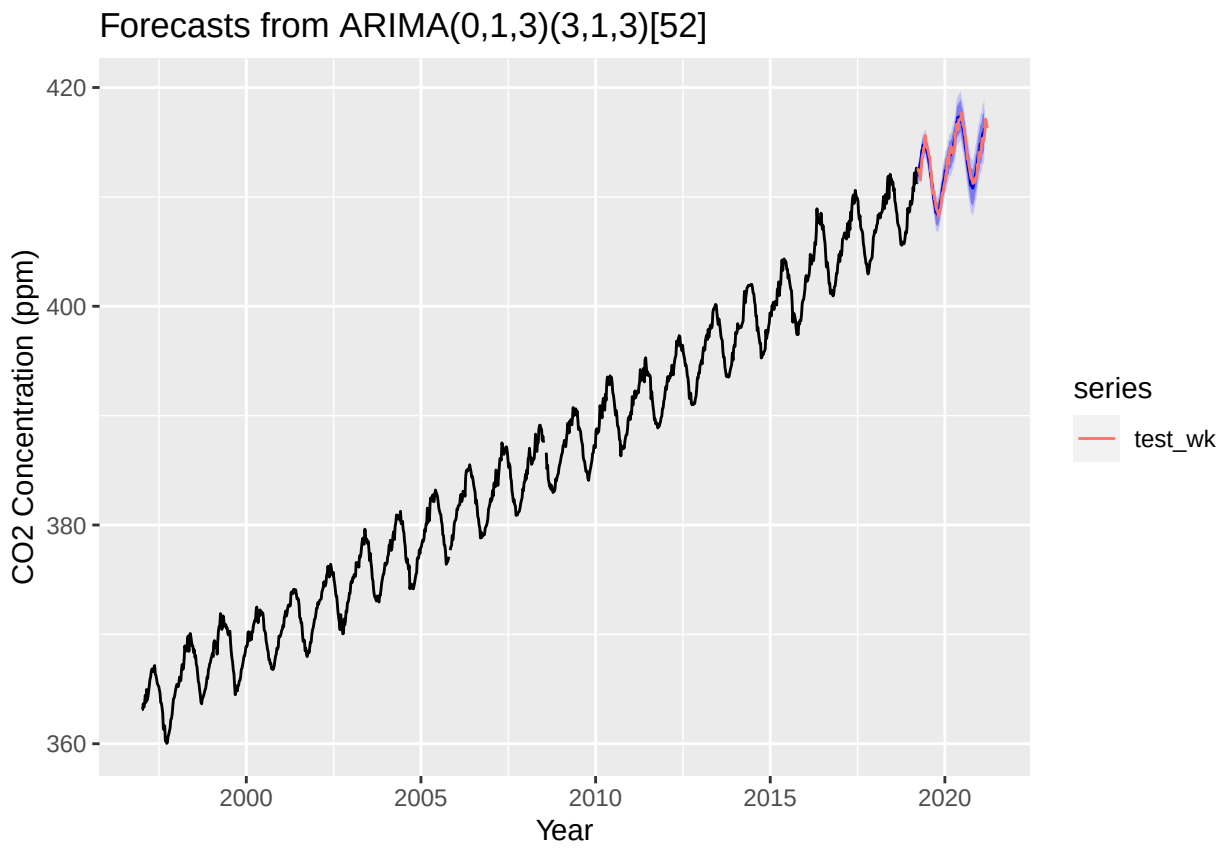
```
##
##  Box-Pierce test
##
## data:  residuals(fit.013.313)
## X-squared = 5.4417e-05, df = 1, p-value = 0.9941
```

```r
Box.test(residuals(fit.013.313), type = "Lj")
```

```
##
##  Box-Ljung test
##
## data:  residuals(fit.013.313)
## X-squared = 5.4559e-05, df = 1, p-value = 0.9941
```

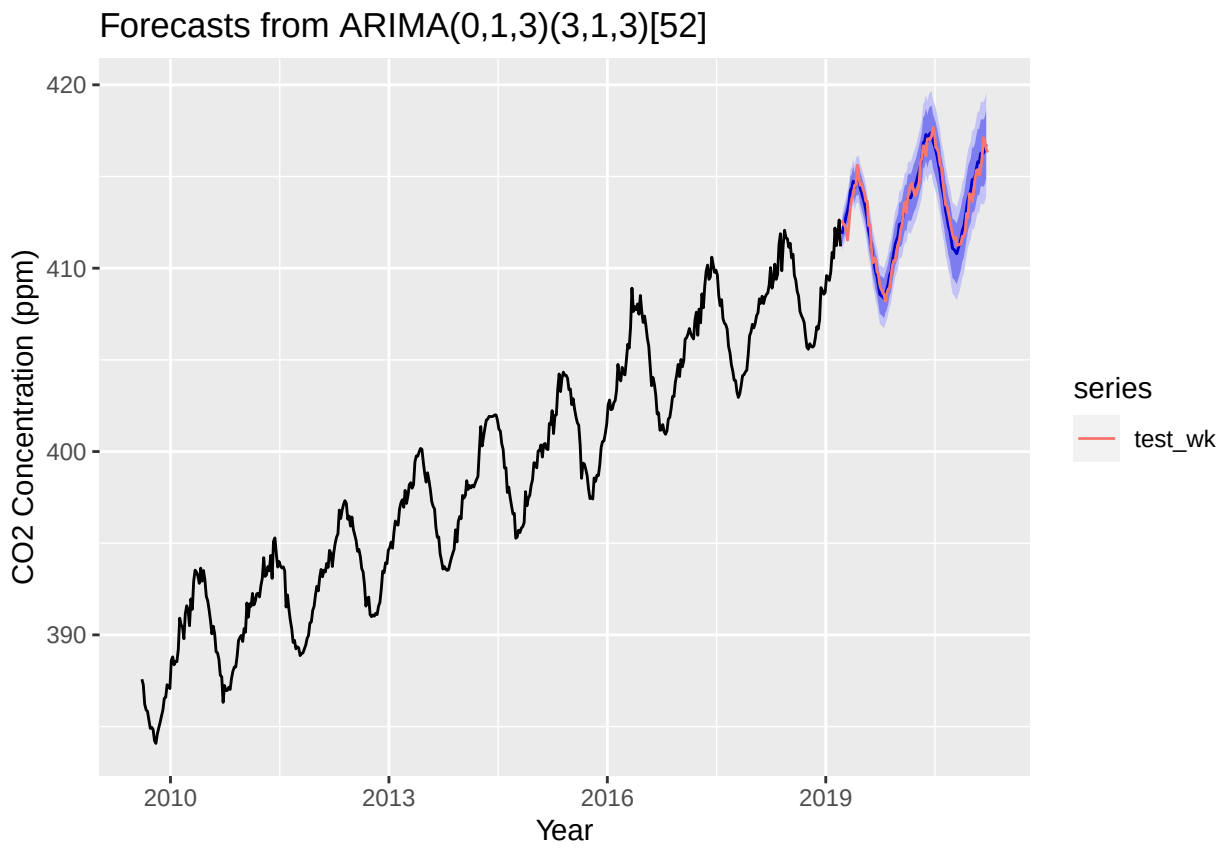
Plot of the entire weekly series (training and test) with the forecast is shown below.

```
fit.013.313 %>%
  forecast(h=100) %>%
  autoplot() + autolayer(test_wk) +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```



This model's forecast for the following two years appears to track the test data for both years.

```
fit.013.313 %>%
  forecast(h=104) %>%
  autoplot(include=500) + autolayer(test_wk) +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```



The ARIMA$(0,1,3)(3,1,3)_{52}$ point prediction for Jan 2100 is similar to the model selected using 5 years of weekly data. However, we see the effect of a much higher RMSE with this model. The confidence intervals for this model are much wider.

```
fit.013.313.forecast <- fit.013.313 %>% forecast(h=4212)
```

```
fit.013.313.forecast %>%
  autoplot() + autolayer(test_wk) +
  xlab('Year') +
  ylab('CO2 Concentration (ppm)')
```