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

Due Monday October 25 2021 11:59pm

Charlie Boatwright, Robert Hosbach

Instructions (Please Read Carefully):

- Submit by the due date. **Late submissions will not be accepted**
- No page limit, but be reasonable
- Do not modify fontsize, margin or line-spacing settings
- One student from each group should submit the lab to their student github repo by the deadline
- Submit two files:
  1. A pdf file that details your answers. Include all R code used to produce the answers
  2. The R markdown (Rmd) file used to produce the pdf file

  The assignment will not be graded unless **both** files are submitted
- Name your files to include all group members names. For example, if the students' names are Stan Cartman and Kenny Kyle, name your files as follows:
  - `StanCartman_KennyKyle_Lab2.Rmd`
  - `StanCartman_KennyKyle_Lab2.pdf`
- Although it sounds obvious, please write your name on page 1 of your pdf and Rmd files
- All answers should include a detailed narrative; make sure that your audience can easily follow the logic of your analysis. All steps used in modelling must be clearly shown and explained; do not simply 'output dump' the results of code without explanation
- If you use libraries and functions for statistical modeling that we have not covered in this course, you must provide an explanation of why such libraries and functions are used and reference the library documentation
- For mathematical formulae, type them in your R markdown file. Do not e.g. write them on a piece of paper, snap a photo, and use the image file
- Incorrectly following submission instructions results in deduction of grades
- Students are expected to act with regard to UC Berkeley Academic Integrity.

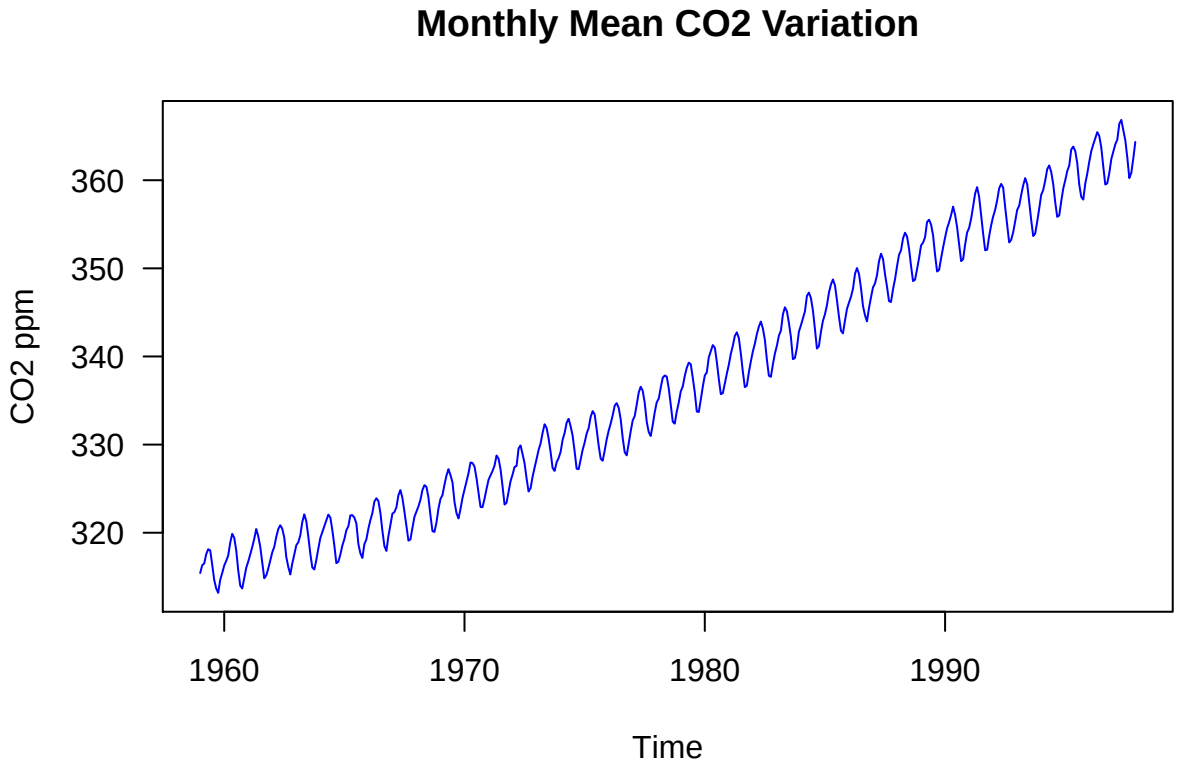
The Keeling Curve

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'.

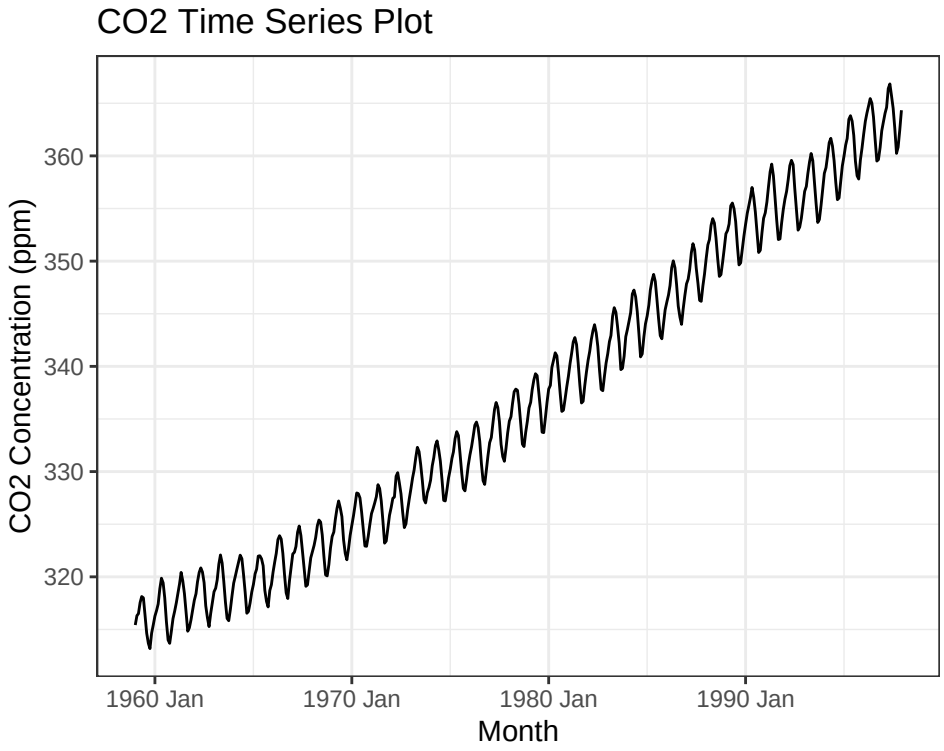
```
plot(co2, ylab = expression("CO2 ppm"), col = 'blue', las = 1)
title(main = "Monthly Mean CO2 Variation")
```



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.

```
# Convert TS into Tsibble
co2.ts <- co2 %>%
  as_tsibble()
# Plot the TS
co2.ts %>%
  autoplot() +
  labs(title = "CO2 Time Series Plot", y = "CO2 Concentration (ppm)", x = "Month") +
  theme_bw()
```




```
# Look at structure, summary, and first/last few rows
str(co2.ts)
```

```
## tbl_ts [468 x 2] (S3: tbl_ts/tbl_df/tbl/data.frame)
##  $ index: mth [1:468] 1959 Jan, 1959 Feb, 1959 Mar, 1959 Apr, 1959 May, 1959 Jun...
##  $ value: num [1:468] 315 316 316 318 318 ...
##  - attr(*, "key")= tibble [1 x 1] (S3: tbl_df/tbl/data.frame)
##   ..$ .rows: list<int> [1:1]
##   .. ..$ : int [1:468] 1 2 3 4 5 6 7 8 9 10 ...
##   .. ..@ ptype: int(0)
##  - attr(*, "index")= chr "index"
##   ..- attr(*, "ordered")= logi TRUE
##  - attr(*, "index2")= chr "index"
##  - attr(*, "interval")= interval [1:1] 1M
```

```
##   ..@ .regular: logi TRUE
```

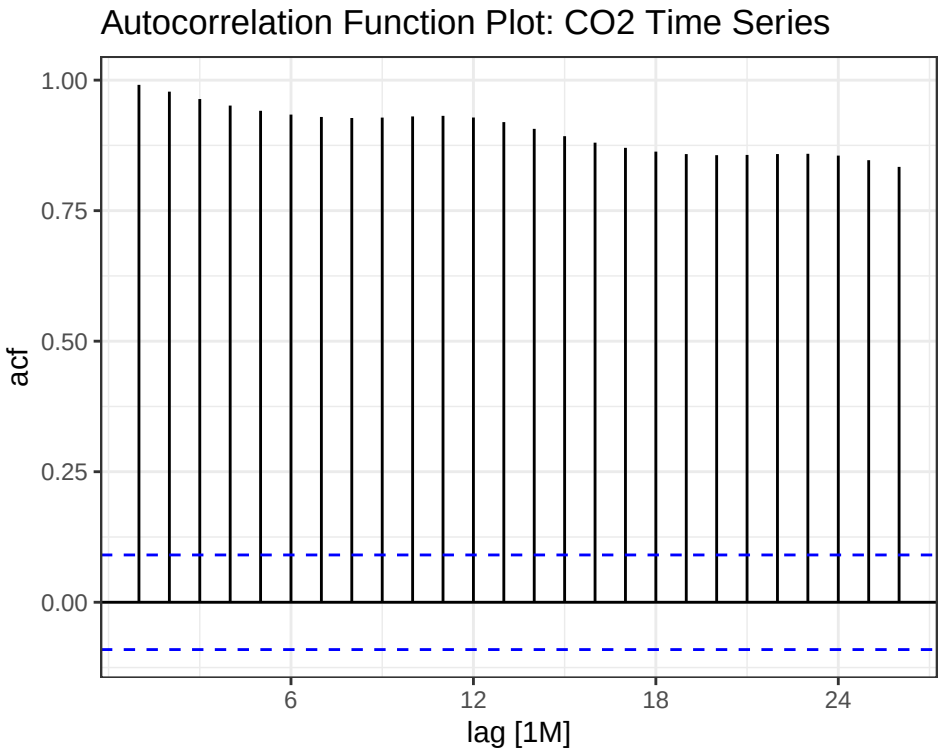
```
summary(co2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   313.2   323.5   335.2   337.1   350.3   366.8
```

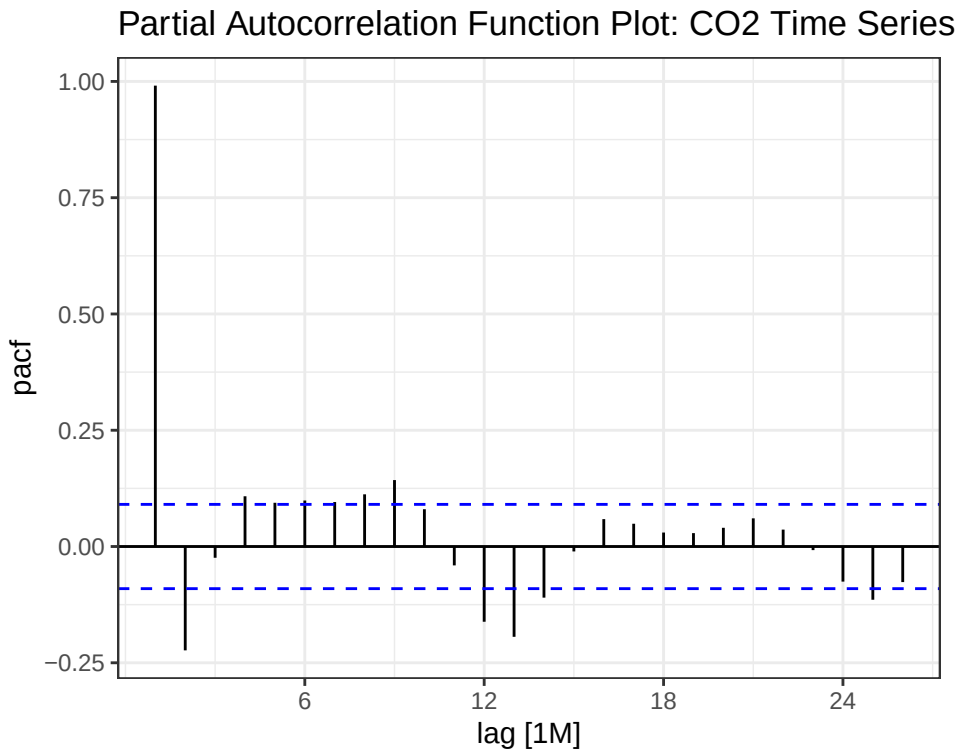
```
cbind(head(co2.ts), tail(co2.ts))
```

```
##      index  value    index  value
## 1 1959 Jan 315.42 1997 Jul 364.52
## 2 1959 Feb 316.31 1997 Aug 362.57
## 3 1959 Mar 316.50 1997 Sep 360.24
## 4 1959 Apr 317.56 1997 Oct 360.83
## 5 1959 May 318.13 1997 Nov 362.49
## 6 1959 Jun 318.00 1997 Dec 364.34
```

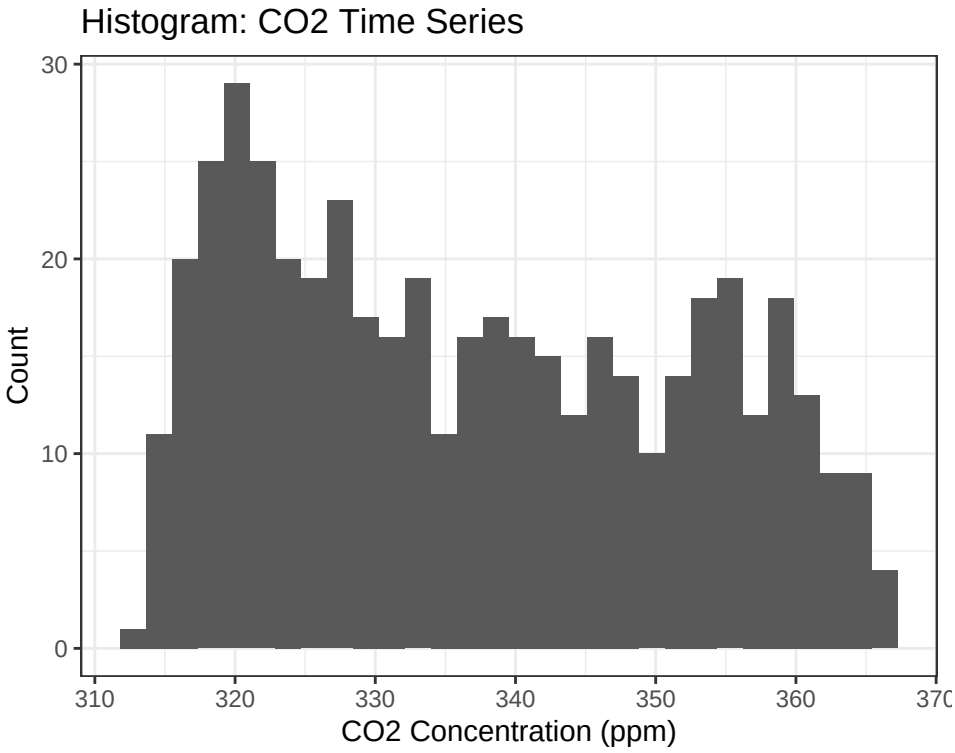
```
# ACF plot
co2.ts %>%
  ACF() %>%
  autoplot() + labs(title="Autocorrelation Function Plot: CO2 Time Series") +
  theme_bw()
```




```
# PACF plot
co2.ts %>%
  PACF() %>%
  autoplot() + labs(title="Partial Autocorrelation Function Plot: CO2 Time Series") +
  theme_bw()
```



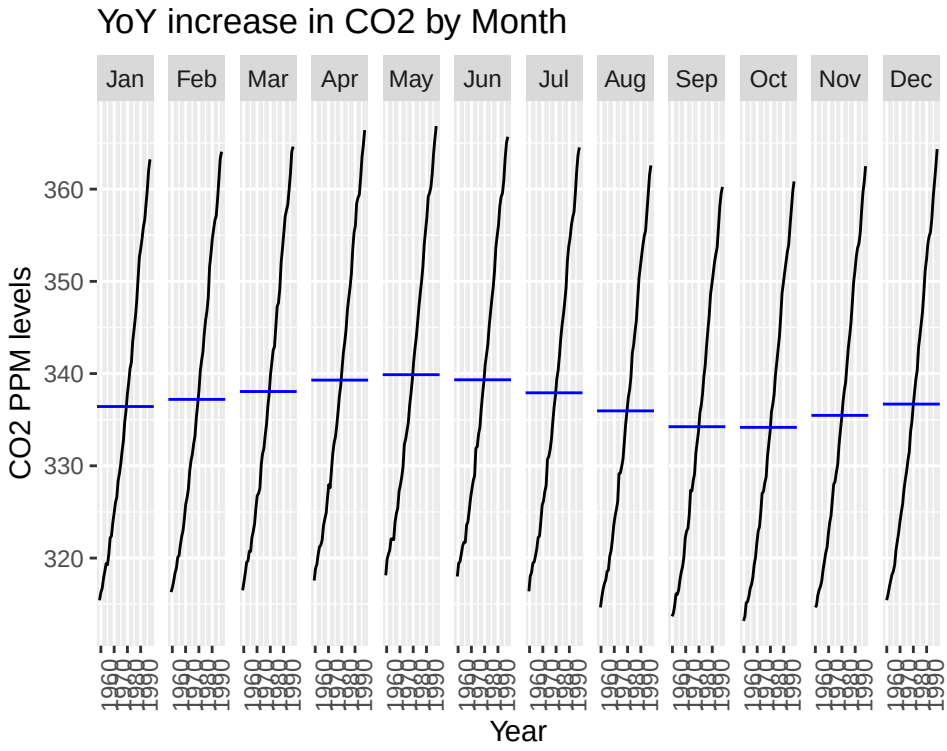

```
# Histogram of the time series
data.frame(co2 = as.numeric(co2)) %>%
  ggplot(aes(x = co2)) +
  geom_histogram(bins = 30) +
  labs(title = "Histogram: CO2 Time Series", y = "Count", x = "CO2 Concentration (ppm)") +
  theme_bw()
```



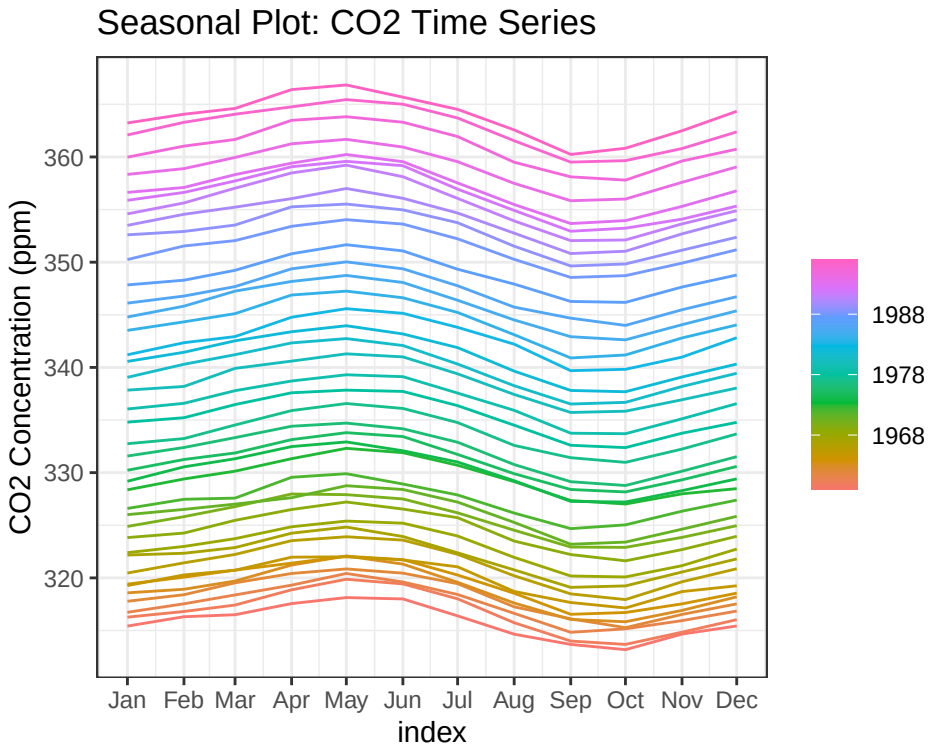
Some observations from our initial EDA:

- We have 468 monthly observations starting with January 1959 until December 1997. The CO2 concentration in the first month of the time series is 315.42 ppm and by 1997 the CO2 concentration grew to 364.34 ppm. There is no missing data in the data set and the median ppm value is 335.2.
- The plot has a distinctive deterministic positive trend as visible through both the acf and the time series itself. The plot of the time series shows consistent growth over time, and the acf has significant and slowly declining lags indicative of a trend in the data.
- The PACF shows decreasing oscillation with a sharp cutoff after the 2nd lag. However, there are still significant lags that show up roughly every 12 months (due to the seasonality of the time series).
- There is clear visual evidence of seasonality both in the plot of the time series itself, as well as in the ACF with the regular bumps at persistent lags.
- The histogram of the data is slightly right skewed, but somewhat uniform from ~330 ppm to ~360 ppm.

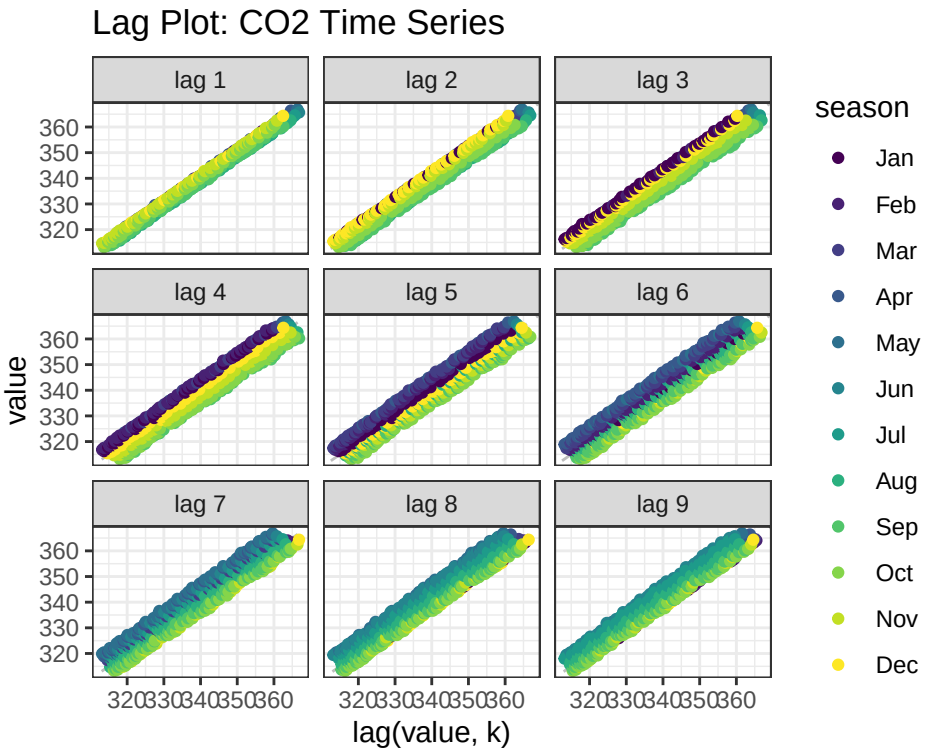
```
# Monthly subseries plot
co2.ts %>%
  gg_subseries(value) +
  labs(
    y = "CO2 PPM levels",
    title = "YoY increase in CO2 by Month",
    x = "Year"
  )
```




```
# Seasonal plot to check for seasonality
co2.ts %>%
  gg_season(value, period = "year") +
  labs(y = "CO2 Concentration (ppm)", title = "Seasonal Plot: CO2 Time Series") +
  theme_bw()
```

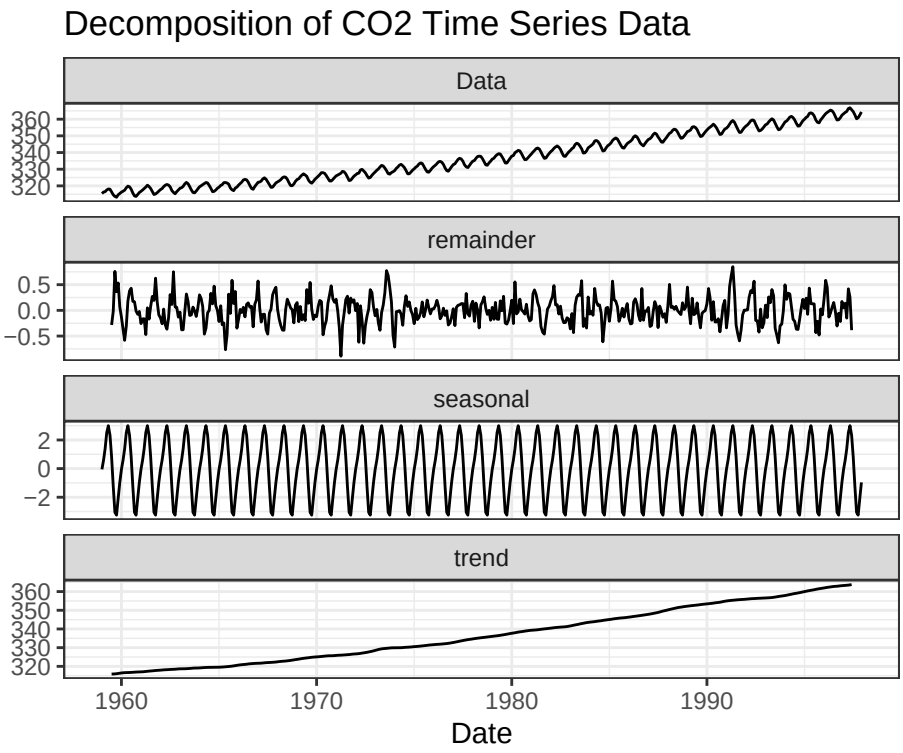


```
# Lag plot
co2.ts %>%
  gg_lag(value, geom = "point") +
  labs(x = "lag(value, k)", title = "Lag Plot: CO2 Time Series") +
  theme_bw()
```



The seasonal subseries plot and the seasonal plots reinforce that the time series contains a strong seasonal pattern as well as the consistent positive deterministic trend visible in the data. Year over year, there is consistent growth in the average ppm and there are distinctive differences in ppm by month within a year. The lag plot shows clear positive correlations for the first nine lags, regardless of the month.

```
# Decompose the TS
co2 %>%
  decompose() %>%
  autoplot() +
  theme_bw() +
  labs(x = "Date", title = "Decomposition of CO2 Time Series Data")
```



Decomposing the time series into component parts reveals both the positive deterministic trend and seasonal trend in a single plot. The deterministic trend appears to be more quadratic than linear. The seasonal component appears to have consistent variance throughout the time series.

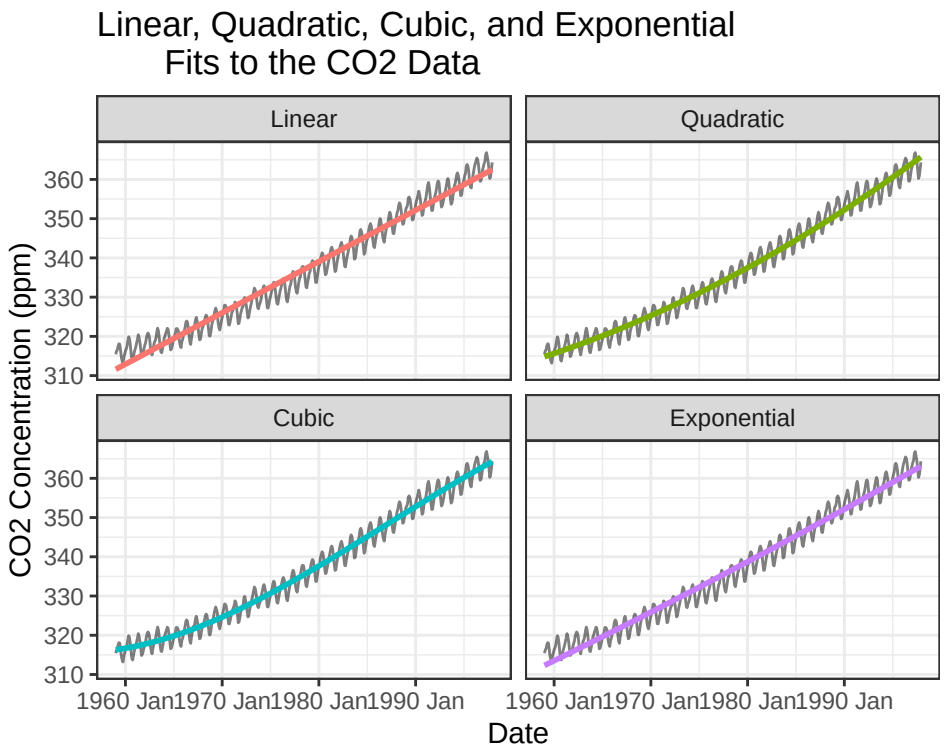
Part 2 (3 points)

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.

Fitting time trend models to the CO2 series

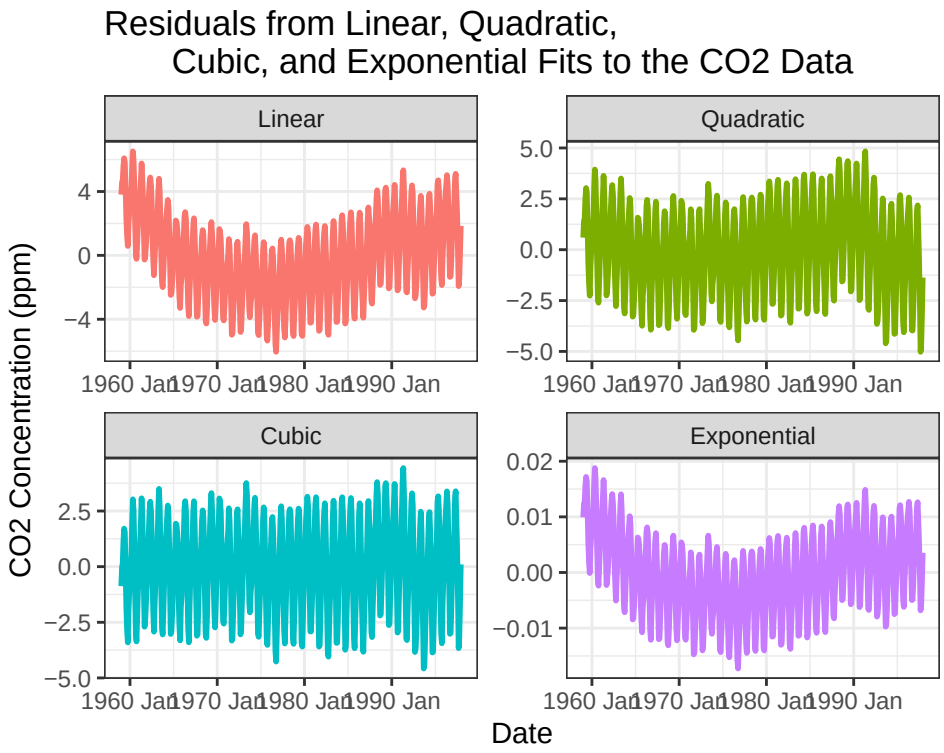
```
# Fitting the time trends
fit_trends <- co2.ts %>%
  model(
    Linear = TSLM(value ~ trend()),
    Quadratic = TSLM(value ~ trend() + I(trend() ^ 2)),
    Cubic = TSLM(value ~ trend() + I(trend() ^ 2) + I(trend() ^ 3)),
    Exponential = TSLM(log(value) ~ trend())
    )

# Plotting the fits of the time series
co2.ts %>%
  autoplot(value, alpha = 0.5) +
  geom_line(data = fitted(fit_trends) %>%
              mutate(.model = factor(.model, levels = c("Linear", "Quadratic", "Cubic", "Expone
```

```
            aes(y = .fitted, colour = .model),
            size = 1) +
  facet_wrap(. ~ .model, nrow = 3) +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Date", y = "CO2 Concentration (ppm)",
       title = "Linear, Quadratic, Cubic, and Exponential
       Fits to the CO2 Data")
```

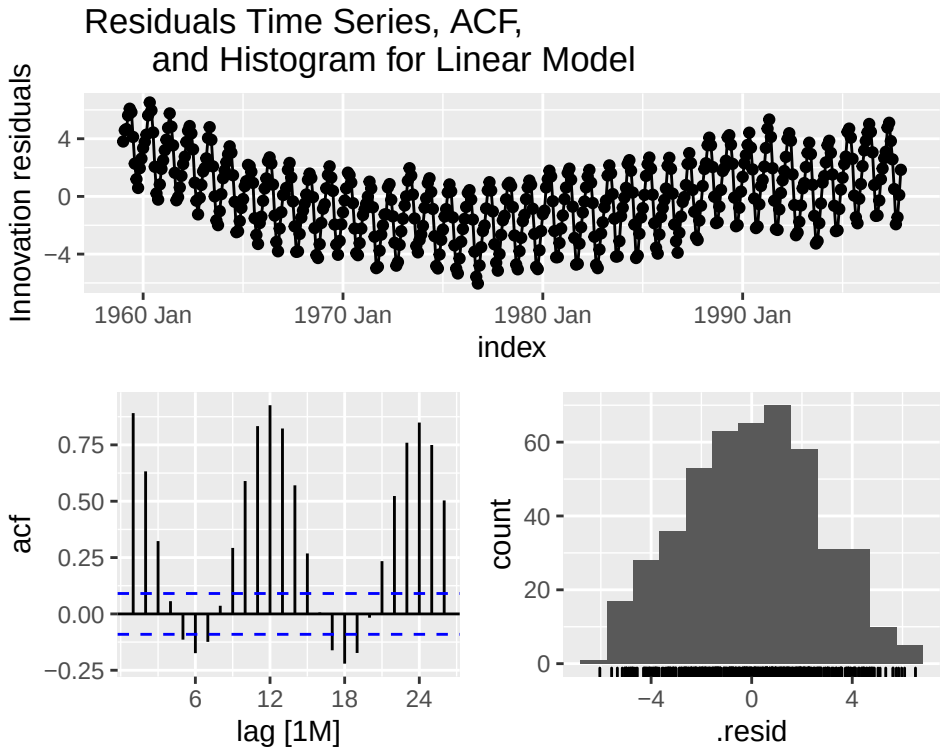


Above we fit linear, quadratic (2nd-order polynomial), cubic (3rd-order polynomial), and exponential time trends to the Keeling Curve data. For all four time trends, the seasonality of the underlying time series shows up prominently in the residuals, which is expected because the time trends we fit have no terms to account for this seasonality. The exponential fit and the linear fit have similar problems in fitting the data as neither approximate the curve visible. The cubic and quadratic fits appear to model the trend much better.

```
# Plotting the residuals
resid(fit_trends) %>%
  mutate(.model = factor(.model, levels = c("Linear", "Quadratic", "Cubic", "Exponential"))) %>
  ggplot(aes(x = index, y = .resid, colour = .model)) +
  geom_line(size = 1) +
  facet_wrap(. ~ .model, nrow = 3, scales = "free") +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Date", y = "CO2 Concentration (ppm)",
       title = "Residuals from Linear, Quadratic,
       Cubic, and Exponential Fits to the CO2 Data")
```




```
# Examine linear model's residuals more closely
co2.ts %>%
  model(Linear = TSLM(value ~ trend())) %>%
  gg_tsresiduals() +
  labs(title = "Residuals Time Series, ACF,
       and Histogram for Linear Model")
```



The residuals in the linear and exponential time trends are heteroskedastic with positive residuals at either ends of the time series (roughly start-1965 and 1990-end) and negative residuals in the middle of the time series. This is expected based on the slight curvature of the underlying trend in the CO2 time series that was noticed during the EDA. In contrast, the residuals of the quadratic and cubic time trends are more constant over the time series, with the cubic time trend having homoskedastic residuals with little discernible trends.

A deeper investigation into the linear trend's residuals shows that while the distribution of the residuals is relatively normal, there is clear autocorrelation in the residuals as shown by the ACF plot with an oscillating pattern of significant lags due primarily to the seasonality apparent in the model that we have not yet accounted for.

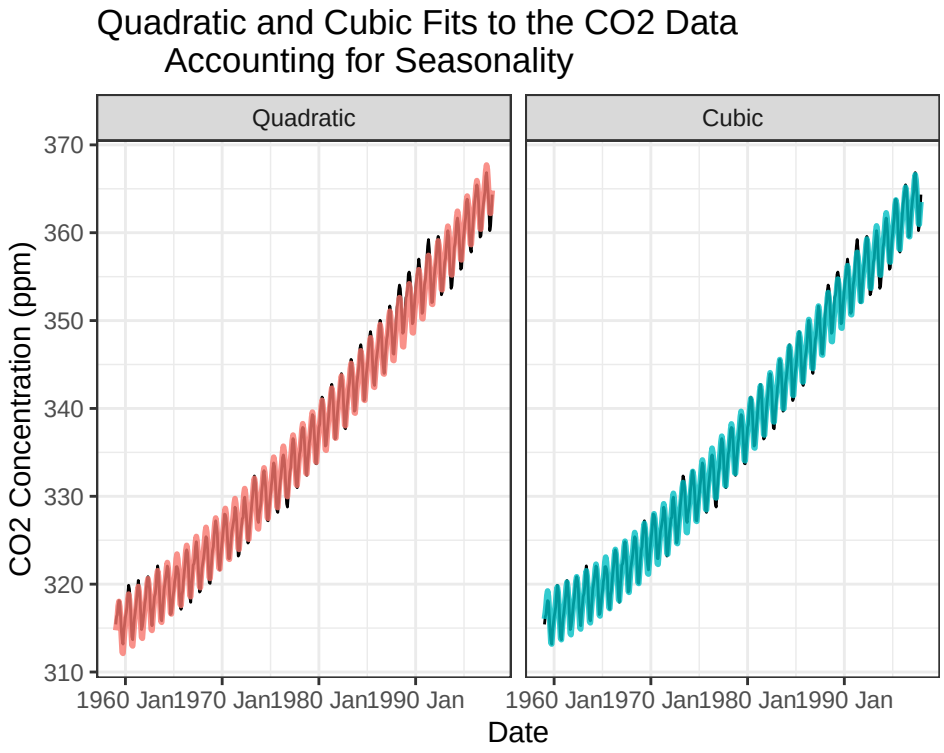
Discussion of log transformation

As per CM2009 p. 109, taking the log of the series can help make the variance more constant over time when the original series has changing variance over time. In this case, the variance in the CO2 series is relatively constant over time, so a log transformation is likely unnecessary.

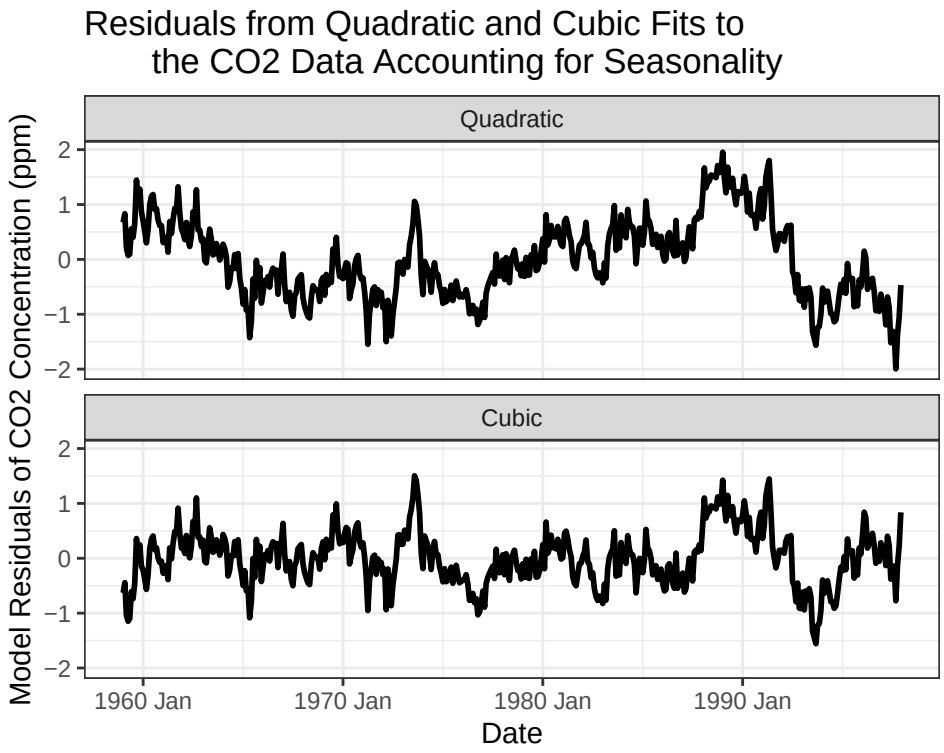
Adding a seasonal dummy variable to higher level polynomials

```
# Adding a seasonal dummy variable to fit Quadratic and Cubic linear fits
fit_seasonal_trends <- co2.ts %>%
  model(
    Quadratic = TSLM(value ~ trend() + I(trend() ^ 2) + season()),
    Cubic = TSLM(value ~ trend() + I(trend() ^ 2) + I(trend() ^ 3) + season()),
    )
# Plot the fits
```

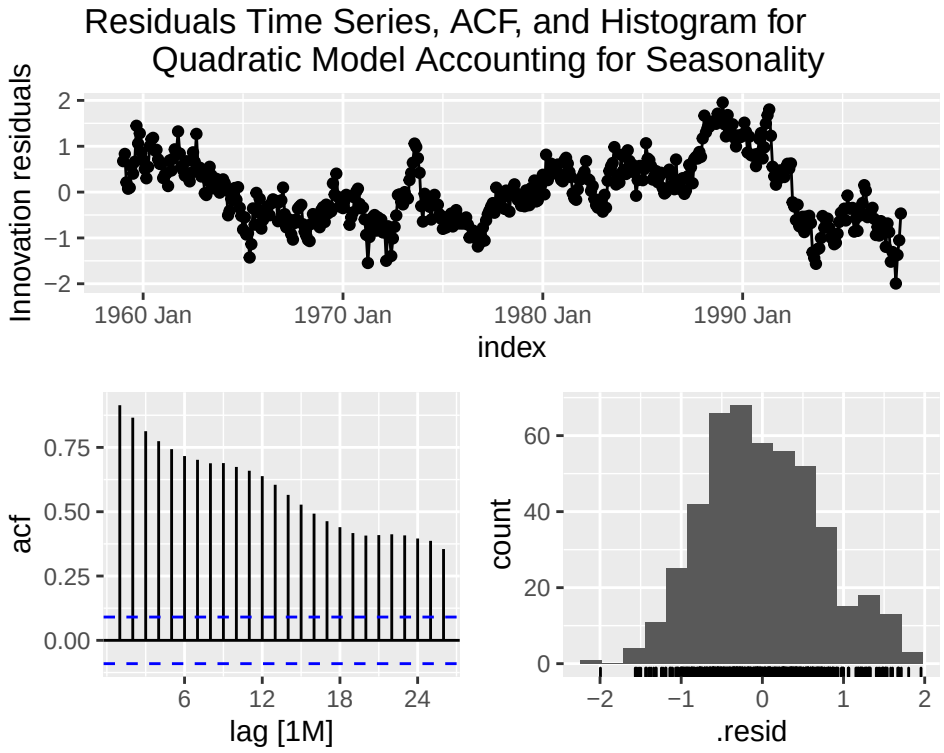
```
co2.ts %>%
  autoplot(value) +
  geom_line(data = fitted(fit_seasonal_trends) %>%
              mutate(.model = factor(.model, levels = c("Quadratic", "Cubic"))),
            aes(y = .fitted, colour = .model),
            size = 1, alpha = 0.8) +
  facet_wrap(. ~ .model, nrow = 1) +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Date", y = "CO2 Concentration (ppm)",
       title = "Quadratic and Cubic Fits to the CO2 Data
       Accounting for Seasonality")
```



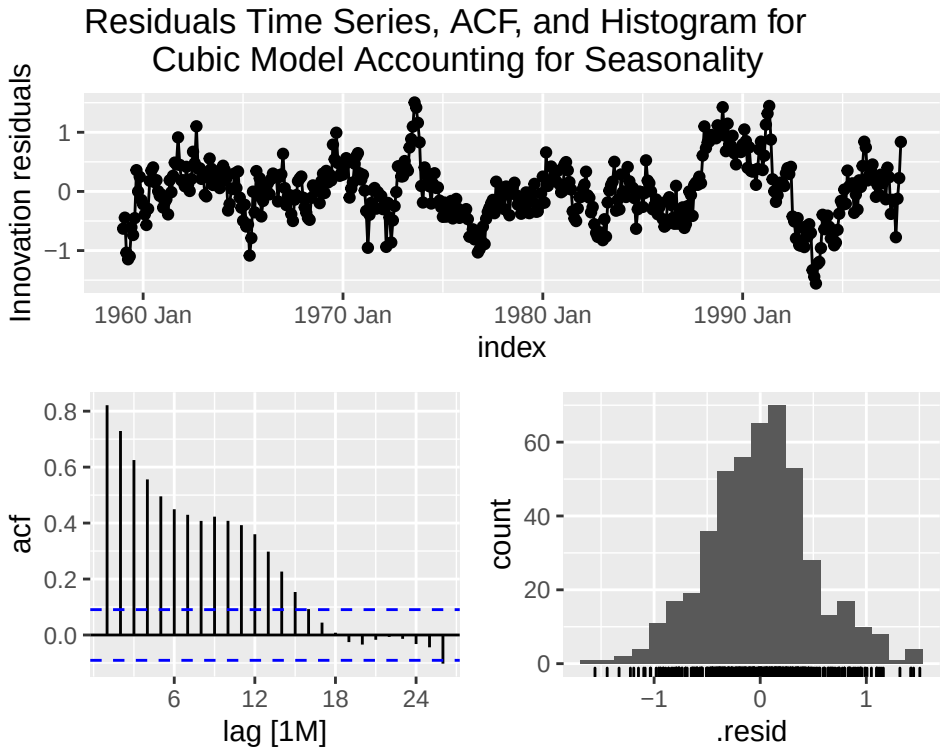

```
# Examine the residuals
residuals(fit_seasonal_trends) %>%
  mutate(.model = factor(.model, levels = c("Quadratic", "Cubic"))) %>%
  ggplot(aes(x = index, y = .resid)) +
  geom_line(size = 1) +
  facet_wrap(. ~ .model, nrow = 2) +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Date", y = "Model Residuals of CO2 Concentration (ppm)",
       title = "Residuals from Quadratic and Cubic Fits to
       the CO2 Data Accounting for Seasonality")
```




```
# Examine quadratic model's residuals more closely
co2.ts %>%
  model(
    Quadratic = TSLM(value ~ trend() + I(trend() ^ 2) + season())
    ) %>%
  gg_tsresiduals() +
  labs(title = "Residuals Time Series, ACF, and Histogram for
       Quadratic Model Accounting for Seasonality")
```



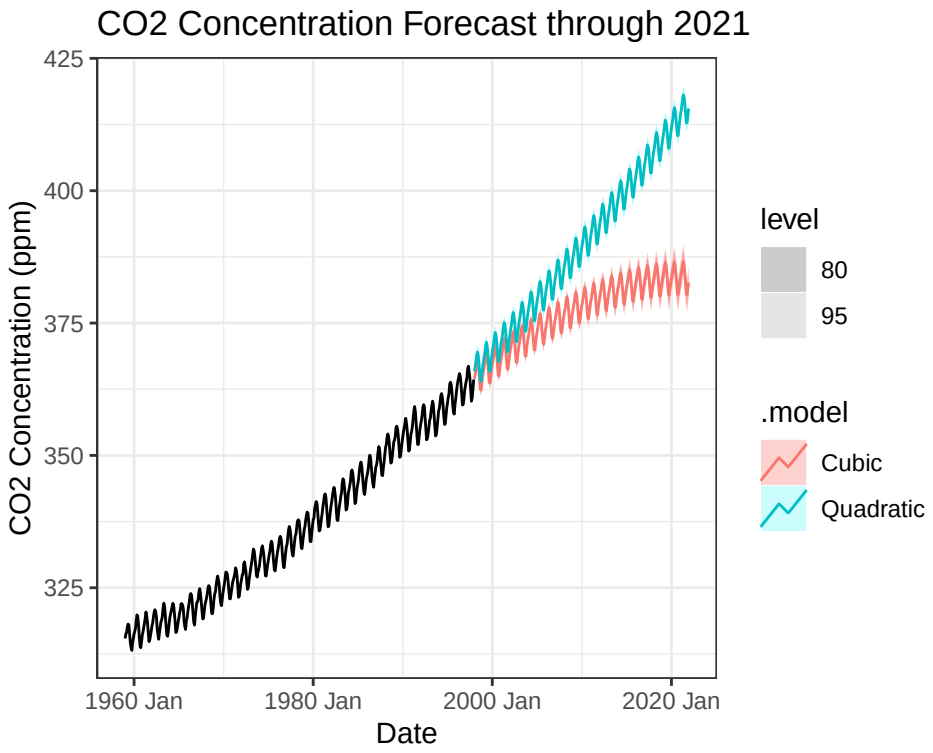

```
co2.ts %>%
  model(
    Cubic = TSLM(value ~ trend() + I(trend() ^ 2) + I(trend() ^ 3) + season())
    ) %>%
  gg_tsresiduals() +
  labs(title = "Residuals Time Series, ACF, and Histogram for
       Cubic Model Accounting for Seasonality")
```



Even after adding dummy variables to the quadratic time trend model to account for seasonality, the model's residuals show characteristic trends and strong autocorrelation through at least the first 24 lags indicating that the linear models are unable to pass the assumption of independence. We see a similar story with the cubic model, although the autocorrelation in the cubic model's residuals drops to zero after 15 lags. We use both models to forecast CO2 concentrations up through 2021 below.

Forecasting to 2021

```
# Determining the forecast lenghth
h <- 12 * (2021 - 1998 + 1)
# Plotting the forecast the based on our models
fit_seasonal_trends %>%
  forecast(h = h) %>%
  autoplot(co2.ts) +
  labs(x = "Date", y = "CO2 Concentration (ppm)",
       title = "CO2 Concentration Forecast through 2021") +
  theme_bw()
```




```
# Extracting the ultimate value of the forecast
forecast(fit_seasonal_trends, h = h) %>%
  filter_index("December 2021")
```

```
## # A fable: 2 x 4 [1M]
## # Key:     .model [2]
##   .model       index        value .mean
##   <chr>        <mth>       <dist> <dbl>
## 1 Quadratic 2021 Dec N(416, 0.82)  416.
## 2 Cubic     2021 Dec  N(383, 2.7)  383.
```

Notwithstanding the failed residual assumptions of the models with seasonal dummy variables described above, both models forecast a continuing upward trend up to the present. While the quadratic model shows an accelerating increase in CO2 concentration, the cubic model, on the other hand, shows a decelerating increase with the forecasts flattening in the mid-2010's. By the end of 2021, the quadratic and cubic models forecast atmospheric CO2 concentration of 416 ppm and 383 ppm, respectively. In comparison, the latest data from the (Primary Mauna Loa CO2 Record from the Scripps Institute of Oceanography)[https://scrippsco2.ucsd.edu/data/atmospheric_co2/primary_mlo_co2_record.html] shows that the monthly CO2 concentration has ranged from 412.9 ppm to 419 ppm in 2021 through September, with an average of 416.5 ppm. This provides evidence that the quadratic model forecast from January 1998 through December 2021 is pretty good, and confirms that the earth's carbon dioxide concentration trend observed from January 1959 through December 1997 has continued on to the present time with no substantial shocks that systematically changed the overarching trend.

In contrast, the cubic model captures a slight inflection in the CO2 series around 1990 that causes it to forecast a substantially lower CO2 concentration by 2021. While the cubic model appears to have marginally better residuals, the lack of forecast accuracy, given that we now know of the

atmospheric CO2 concentration, is highly problematic. Based on this evidence, it appears that the cubic model is over fitting the data and is too sensitive to the slight inflection at the end of the time series.

This is a scenario where the quadratic model has residuals that are nearly as good as the cubic model's residuals (in terms of producing a residual time series that resembles that of a white noise process), but has a much higher forecast accuracy when compared to the cubic model. Moreover, the parsimony of the quadratic model relative to the cubic model is preferred for such long forecasts, and in this case that logic bears out. Because of this, for part 4 of this exercise, we will only carry forward the quadratic model.

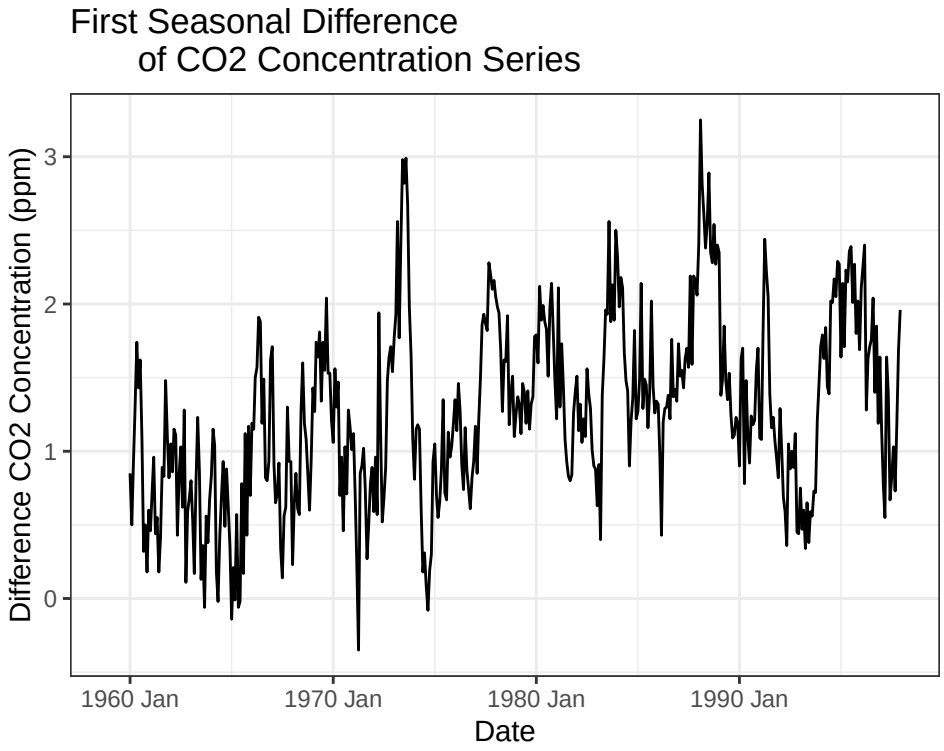
Part 3 (4 points)

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.

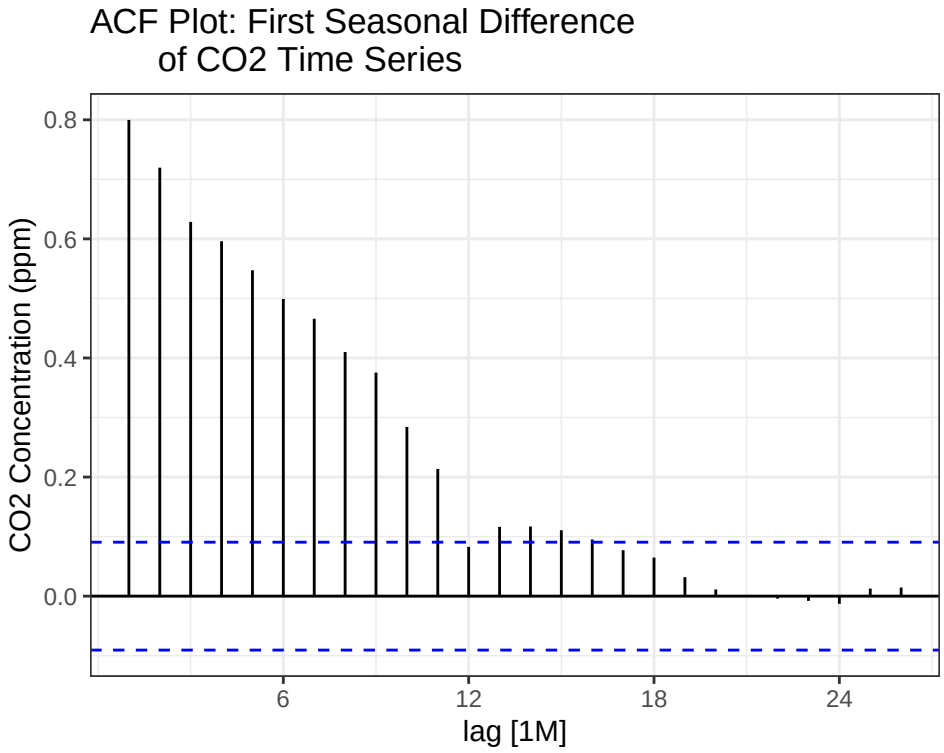
Fitting an ARIMA model

As previously discussed, the CO2 series has both a deterministic trend and a seasonal trend that we have to account for in our ARIMA model since the time series is clearly not stationary. We will first investigate if differencing the model can make the times series roughly stationary.

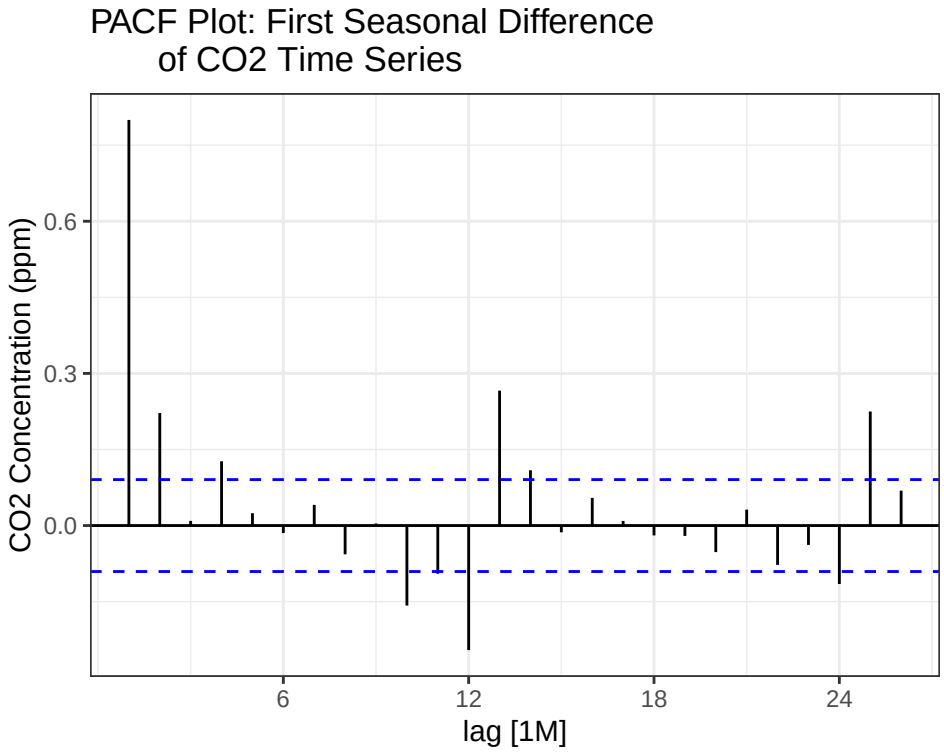
```
# Seasonal difference first due to strong seasonality in the series
# (recommendation of FPP3 to do seasonal differencing before first differencing
# when strong seasonality is present)
co2.ts %>%
  autoplot(difference(value, 12)) +
  labs(x = "Date", y = "Difference CO2 Concentration (ppm)",
       title = "First Seasonal Difference
       of CO2 Concentration Series") +
  theme_bw()
```




```
# ACF plot to check for autocorrelation
co2.ts %>%
  ACF(difference(value, 12)) %>%
  autoplot() +
  labs(y = "CO2 Concentration (ppm)", title = "ACF Plot: First Seasonal Difference
       of CO2 Time Series") +
  theme_bw()
```



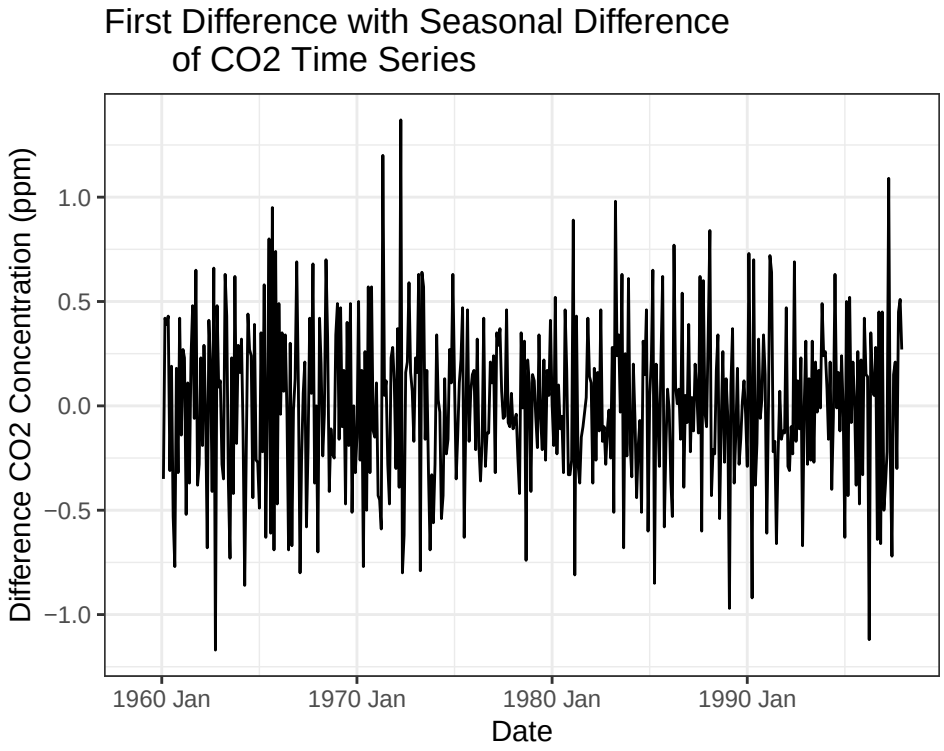

```
# PACF plot to check for autocorrelation
co2.ts %>%
  PACF(difference(value, 12)) %>%
  autoplot() +
  labs(y = "CO2 Concentration (ppm)", title = "PACF Plot: First Seasonal Difference
       of CO2 Time Series") +
  theme_bw()
```



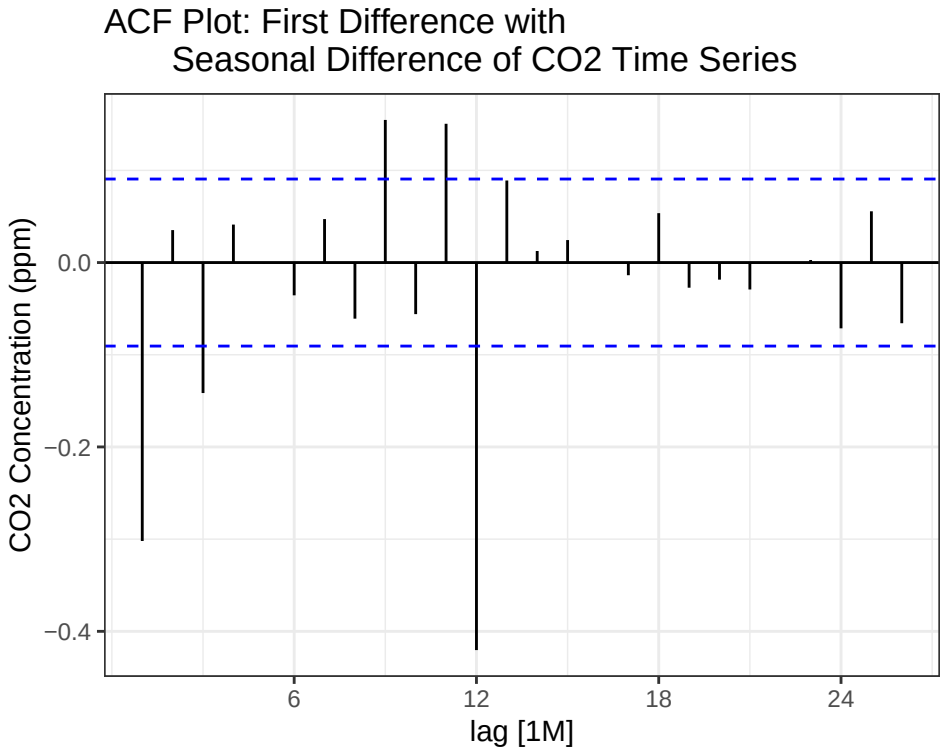
The first plot shows the time series with seasonal first differencing. The transformed series seems to successfully remove the seasonal elements from the original time series. The ACF is now gradually decreasing without seasonal bumps and the PACF still shows decreasing oscillation. However, there is still has evidence of a deterministic trend as the differenced time series retains a positive trend and the ACF has gradually decaying significant lags. Based on this evidence, we expect that the best ARIMA model fit to this CO2 series will incorporate a seasonal first-difference term *i.e.*, $D = 1$.

We investigated a second seasonal differencing, but there the resulting time series does not indicate that a second seasonal differencing is necessary. For the sake of efficiency in our response, the exploration of the second differencing was removed.

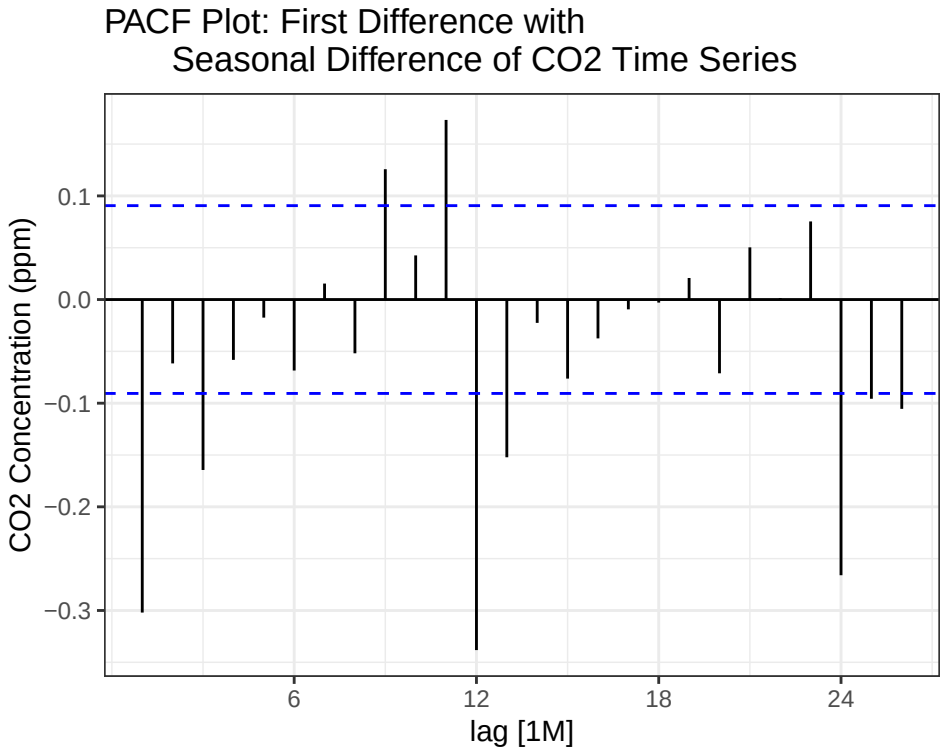
```
# Differencing on the seasonally differenced time series
co2.ts %>%
  autoplot(difference(difference(value, 12), 1)) +
  labs(x = "Date", y = "Difference CO2 Concentration (ppm)",
       title = "First Difference with Seasonal Difference
       of CO2 Time Series") +
  theme_bw()
```




```
# ACF plot to check for autocorrelation
co2.ts %>%
  ACF(difference(difference(value, 12), 1)) %>%
  autoplot() +
  labs(y = "CO2 Concentration (ppm)",
       title = "ACF Plot: First Difference with
       Seasonal Difference of CO2 Time Series") +
  theme_bw()
```




```
# PACF plot to check for autocorrelation
co2.ts %>%
  PACF(difference(difference(value, 12), 1)) %>%
  autoplot() +
  labs(y = "CO2 Concentration (ppm)",
       title = "PACF Plot: First Difference with
       Seasonal Difference of CO2 Time Series") +
  theme_bw()
```




```
forecast::ndiffs(as.ts(co2.ts))
```

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

```
forecast::nsdiffs(as.ts(co2.ts))
```

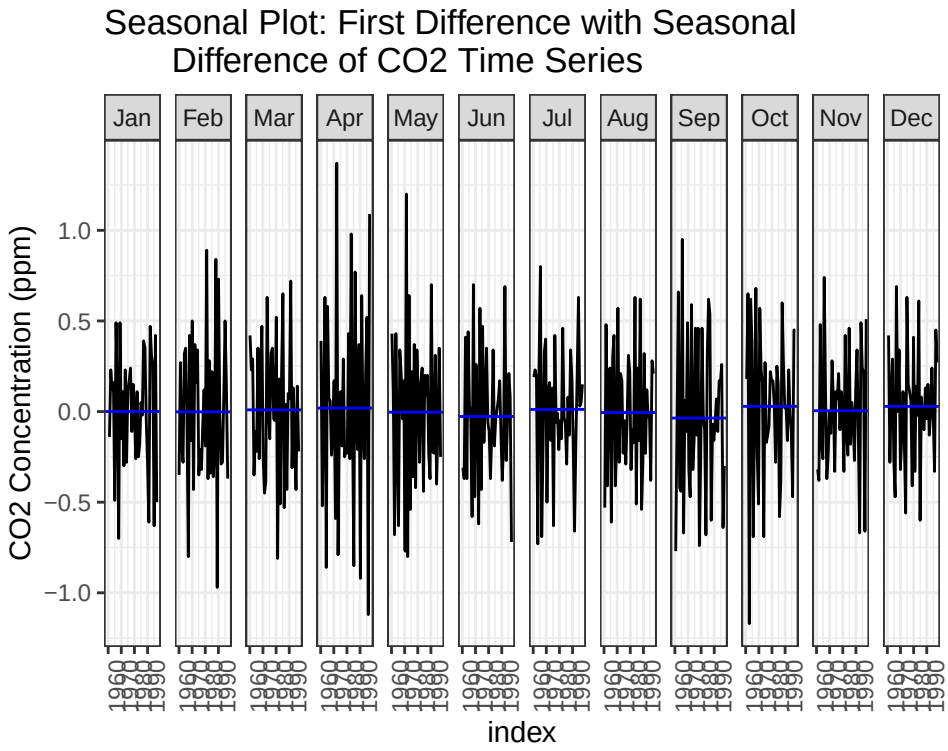
```
## [1] 1
```

The time series after we take both first difference and a first seasonal difference has constant mean and variance over the entire series. It does also appear that the time series after the differencing has less persistence, which is corroborated by the ACF plot. Based on the plots, we expect our ARIMA model to contain one difference ($d = 1$) and one seasonal difference ($D = 1$). We confirm this by also running the `forecast::ndiffs()` and `forecast::nsdiffs()` functions on the time series, which estimate how many differences and seasonal differences are required to get the series to be stationary, respectively. These functions indicate that one difference and one seasonal difference are required for this time series. The ACF and PACF of the differenced series have a few significant lags, but because there no longer appears to be additional trends or seasonal elements in the time series we can model the remaining time series with AR and MA processes.

As an exercise, we also looked at the series after first-differencing and then two seasonal differences. In this case, the ACF plot shows very little (if any) improvement over the ACF plot for the first-differenced time series with one seasonal difference indicating that second differencing is likely unnecessary.

```
# Monthly plot to check for seasonality
co2.ts %>%
  gg_subseries(difference(difference(value, 12), 1), period = "year") +
  labs(y = "CO2 Concentration (ppm)", title = "Seasonal Plot: First Difference with Seasonal
       Difference of CO2 Time Series") +
```

```
  theme_bw()+
  theme(axis.text.x=element_text(angle=90,hjust=1))
```



The monthly plot of the differenced time series reinforces that our best ARIMA model will likely contain a single difference, $d = 1$, and a single seasonal difference $D = 1$.

```
# Create function to search for the best seasonal ARIMA model (see CM 2009 p.144)
# Expect that best model(s) will all have d = 1, but let it vary to confirm
check.arimas <- function(x.ts, maxord = c(1, 1, 1, 1, 1, 1)) {
  ps <- c()
  ds <- c()
  qs <- c()
  Ps <- c()
  Ds <- c()
  Qs <- c()
  aics <- c()
  aiccs <- c()
  bics <- c()
  n <- length(x.ts)
  idx = 1
  for (p in 0:maxord[1]) for (d in 0:maxord[2]) for (q in 0:maxord[3])
    for (P in 0:maxord[4]) for (D in 0:maxord[5]) for (Q in 0:maxord[6]) {
      tryCatch({
        fit <- forecast::Arima(x.ts, order = c(p, d, q),
                    seasonal = list(order = c(P, D, Q), frequency(x.ts)),
```

```
                    method = "ML")
        ps[idx] <- p
        ds[idx] <- d
        qs[idx] <- q
        Ps[idx] <- P
        Ds[idx] <- D
        Qs[idx] <- Q
        aics[idx] <- fit$aic
        aiccs[idx] <- fit$aicc
        bics[idx] <- fit$bic
        idx = idx + 1
      }, error = function(e){
        ps[idx] <- p
        ds[idx] <- d
        qs[idx] <- q
        Ps[idx] <- P
        Ds[idx] <- D
        Qs[idx] <- Q
        aics[idx] <- 999999
        aiccs[idx] <- 999999
        bics[idx] <- 999999
        idx = idx + 1
        })
    }
  results <- data.frame(
    p = ps,
    d = ds,
    q = qs,
    P = Ps,
    D = Ds,
    Q = Qs,
    AIC = aics,
    AICc = aiccs,
    BIC = bics
  )
  results[order(results$AIC), ]
}

# Iterate through seasonal ARIMA models to find the best
# best.arimas <- check.arimas(co2, maxord = c(3,1,3,3,1,3))
# We commented this out because the function takes a long time to run.
# We saved the full list of arima models
# into a csv and are only sharing the top 10 by BIC and AIC here.
# The csv attached has the full list.
best.arimas <- read.csv("w271_lab2_part3.csv")
best.arimas %>% arrange(AIC) %>% head(10)
```

Determining AR and MA parameters for the best ARIMA model

```
##      X p d q P D Q      AIC     AICc      BIC
## 1  181 0 1 1 2 1 2 173.5011 173.6886 198.2229
## 2  213 0 1 2 2 1 2 174.0323 174.2829 202.8744
## 3  365 1 0 3 2 1 2 176.6324 177.0360 213.7349
## 4  228 0 1 3 0 1 1 176.8645 176.9982 197.4660
## 5  245 0 1 3 2 1 2 177.5217 177.8446 210.4841
## 6  869 3 1 3 2 1 2 177.6093 178.2053 222.9326
## 7  238 0 1 3 1 1 3 177.6845 178.0074 210.6469
## 8  820 3 1 1 2 1 2 177.8093 178.2138 214.8920
## 9  615 2 1 1 0 1 1 177.8278 177.9614 198.4293
## 10 230 0 1 3 0 1 3 177.9601 178.2107 206.8022
```

```
best.arimas %>% arrange(BIC) %>% head(10)
```

```
##      X p d q P D Q      AIC     AICc      BIC
## 1  164 0 1 1 0 1 1 178.1557 178.2089 190.5166
## 2  408 1 1 1 0 1 1 178.0672 178.1561 194.5484
## 3  196 0 1 2 0 1 1 179.0924 179.1813 195.5736
## 4  165 0 1 1 0 1 2 179.8808 179.9696 196.3619
## 5  172 0 1 1 1 1 1 179.9235 180.0124 196.4047
## 6  378 1 1 0 0 1 1 184.1817 184.2350 196.5426
## 7  228 0 1 3 0 1 1 176.8645 176.9982 197.4660
## 8  181 0 1 1 2 1 2 173.5011 173.6886 198.2229
## 9  615 2 1 1 0 1 1 177.8278 177.9614 198.4293
## 10 432 1 1 2 0 1 1 178.6271 178.7607 199.2286
```

To choose the best model, we first decided to compare the model with the lowest AIC against the model with the lowest BIC. The model with the lowest AIC is ARIMA(0,1,1)(2,1,2)[12] and the model with the lowest BIC is ARIMA(0,1,1)(0,1,1)[12].

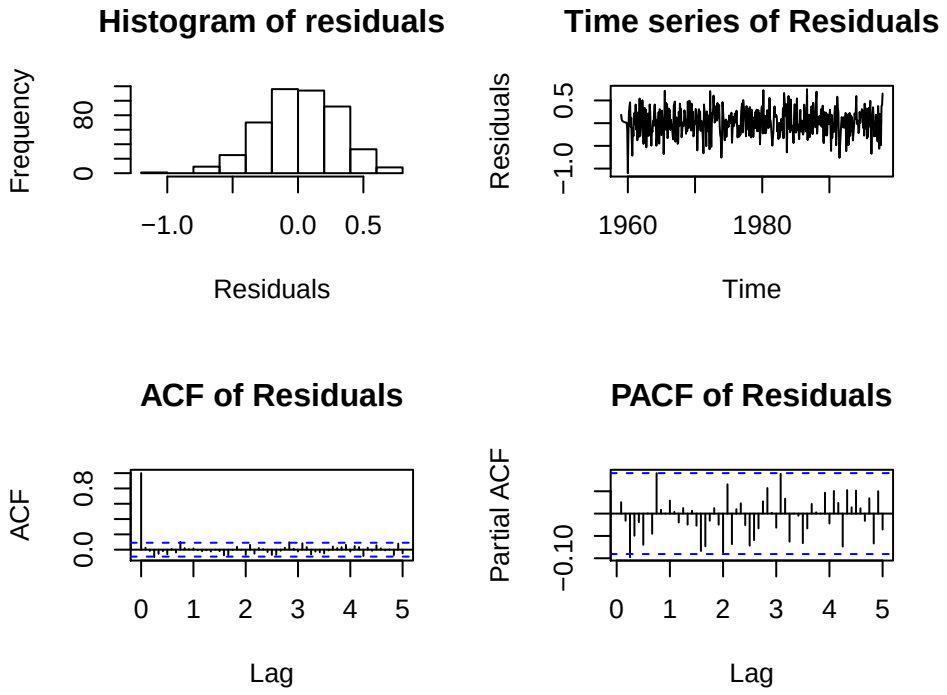
Based on having the lowest BIC and being the more parsimonious model, the ARIMA(0,1,1)(0,1,1)[12] model is initially preferred for this forecasting exercise. However, residual diagnostics must be conducted to ensure the model represents the original time series well.

```
# Generate model
co2.arima <- forecast::Arima(co2, order = c(0, 1, 1),
                             seasonal = list(order = c(0, 1, 1), 12),
                             method = "ML")
co2.arima
```

```
## Series: co2
## ARIMA(0,1,1)(0,1,1)[12]
##
## Coefficients:
##           ma1     sma1
##       -0.3501  -0.8507
## s.e.   0.0496   0.0256
##
## sigma^2 estimated as 0.08581:  log likelihood=-86.08
```

```
## AIC=178.16   AICc=178.21   BIC=190.52
```

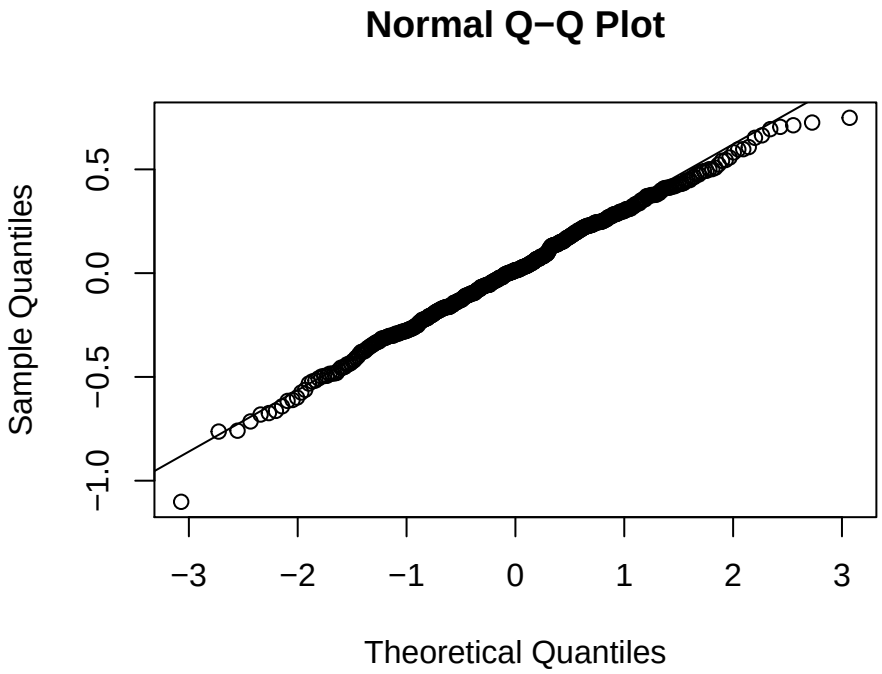
```
par(mfrow = c(2, 2))
hist(co2.arima$residuals, main = "Histogram of residuals", xlab = "Residuals")
plot.ts(co2.arima$residuals, main = "Time series of Residuals", ylab = "Residuals")
acf(co2.arima$residuals, main='ACF of Residuals', lag.max = 60)
pacf(co2.arima$residuals, main='PACF of Residuals', lag.max = 60)
```




```
# Formally test residuals
shapiro.test(co2.arima$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  co2.arima$residuals
## W = 0.99611, p-value = 0.3077
```

```
par(mfrow = c(1, 1))
qqnorm(co2.arima$residuals)
qqline(co2.arima$residuals)
```




```
# Box-Ljung test
Box.test(co2.arima$residuals, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  co2.arima$residuals
## X-squared = 0.30892, df = 1, p-value = 0.5783
```

The residuals diagnostics show the following:

- The model residuals are approximately normally distributed, and there are only three significant lags (lags 3, 9, and 35) in the ACF plot that goes out to lag 60. Additionally, the PACF plot shows only lag 3 as significant.
- The Shapiro-Wilk Normality Test produces a p-value of 0.308, which indicates that there is insufficient evidence to reject the null hypothesis that the residuals are distributed normally.
- The Box-Ljung test produces a p-value of 0.578, which indicates that there is insufficient evidence to reject the null hypothesis that the residuals are independent.

Overall, the seasonal ARIMA(0,1,1)(0,1,1)[12] model is the best model in terms of BIC (over the parameter field that we searched), is more parsimonious than the model with the best AIC, and has model residuals that appear to be generated by a white noise process (at least, there is insufficient evidence to reject this).

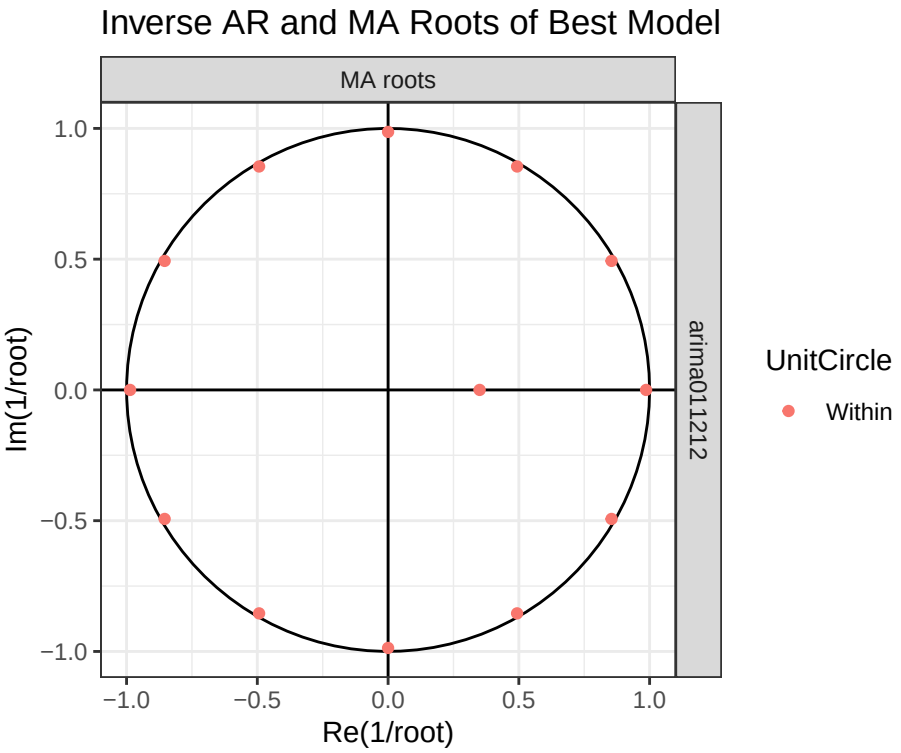
Forecasting ARIMA(0,1,1)(0,1,1)[12] through 2021

```
# Calculate number of forecast periods to get us through 2021
# Forecast will start in January 1998
h <- 12 * (2021 - 1998 + 1)
```

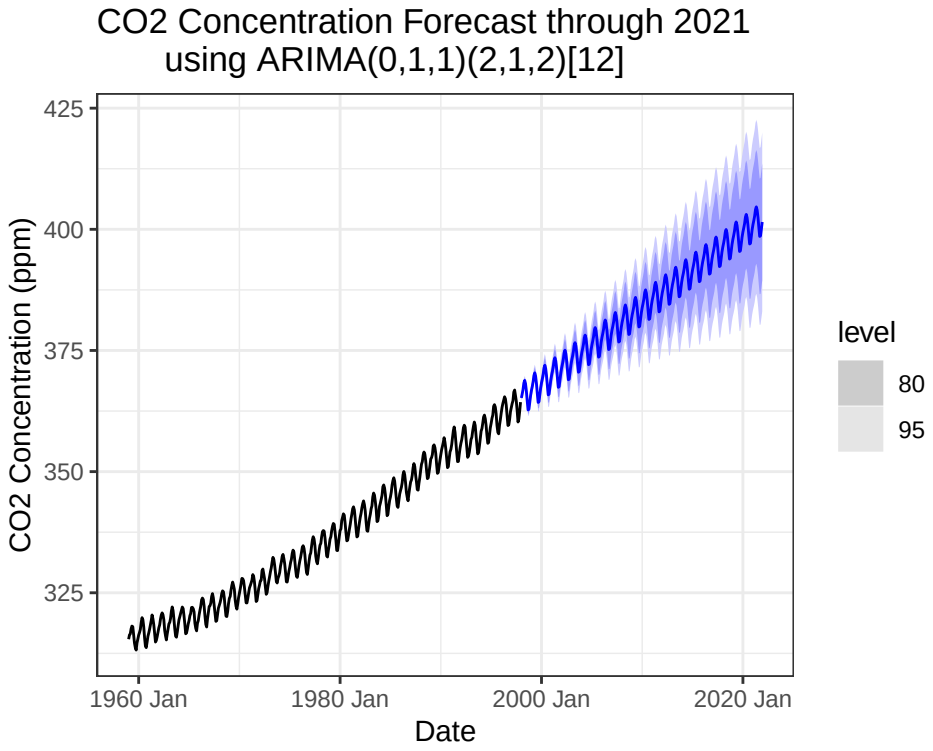
```r
# Generate seasonal ARIMA model using Fable and ensure it's the same as before
fabmod <- co2.ts %>%
  model(arima011212 = ARIMA(value ~ pdq(0, 1, 1) + PDQ(0, 1, 1)))
coefficients(fabmod) # The coefficients are the same!
```

```
## # A tibble: 2 x 6
##   .model      term  estimate std.error statistic   p.value
##   <chr>       <chr>    <dbl>     <dbl>     <dbl>     <dbl>
## 1 arima011212 ma1     -0.350    0.0496     -7.05 6.56e- 12
## 2 arima011212 sma1    -0.851    0.0257    -33.2  1.96e-123
```

```r
# Look at ARMA roots
fabmod %>%
  gg_arma() +
  theme_bw() +
  labs(title = "Inverse AR and MA Roots of Best Model")
```




```r
# Forecast plot
fabletools::forecast(fabmod, h = h) %>%
  autoplot(co2.ts) +
  labs(title = "CO2 Concentration Forecast through 2021
       using ARIMA(0,1,1)(2,1,2)[12]",
       x = "Date", y = "CO2 Concentration (ppm)") +
  theme_bw()
```



We first look at the inverse AR and MA roots of the seasonal ARIMA model and see that all of the MA roots are *very* close to the unit circle. This suggests some instability and potential bias in the forecast. The time series plot shows that the forecast captures the essence of the CO2 series trend as well as the inherent seasonality. Moreover, the confidence intervals around the central estimates get larger the further out the forecast is, which is expected because there is more uncertainty the further out you forecast.

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 in the table and skip the first set of rows
co2_weekly <- read.table("co2_weekly_mlo.txt", header = FALSE, skip = 45, sep = "")
colnames(co2_weekly) <- c("year","month","day","decimal","ppm","days","prior_year","10_year","

# Transforming the date column
co2_weekly <- co2_weekly %>%
  mutate(date = as.Date(paste(year,month,day, sep = "-"))) %>%
  na_if(-999.99) %>%
  select(date, ppm) %>%
  as_tsibble(index = date)
```

```r
# Sanity check to ensure data set is read in properly
# A visual inspection of the data set shows there should be 18 missing ppm values
summary(co2_weekly)
```

```
##       date                 ppm
##  Min.   :1974-05-19   Min.   :326.7
##  1st Qu.:1986-02-24   1st Qu.:347.5
##  Median :1997-12-03   Median :365.3
##  Mean   :1997-12-03   Mean   :368.4
##  3rd Qu.:2009-09-11   3rd Qu.:388.5
##  Max.   :2021-06-20   Max.   :420.0
##                       NA's   :18
```

```r
str(co2_weekly)
```

```
## tbl_ts [2,458 x 2] (S3: tbl_ts/tbl_df/tbl/data.frame)
##  $ date: Date[1:2458], format: "1974-05-19" "1974-05-26" ...
##  $ ppm : num [1:2458] 333 333 332 332 332 ...
##  - attr(*, "key")= tibble [1 x 1] (S3: tbl_df/tbl/data.frame)
##   ..$ .rows: list<int> [1:1]
##   .. ..$ : int [1:2458] 1 2 3 4 5 6 7 8 9 10 ...
##   .. ..@ ptype: int(0)
##  - attr(*, "index")= chr "date"
##   ..- attr(*, "ordered")= logi TRUE
##  - attr(*, "index2")= chr "date"
##  - attr(*, "interval")= interval [1:1] 7D
##   ..@ .regular: logi TRUE
```

```r
# Look at missing values
co2_weekly %>% filter(is.na(ppm))
```

```
## # A tsibble: 18 x 2 [7D]
##    date         ppm
##    <date>     <dbl>
##  1 1975-10-05    NA
##  2 1975-12-07    NA
##  3 1975-12-14    NA
##  4 1975-12-21    NA
##  5 1975-12-28    NA
##  6 1976-06-27    NA
##  7 1982-03-21    NA
##  8 1982-04-11    NA
##  9 1982-04-18    NA
## 10 1983-08-07    NA
## 11 1984-04-01    NA
## 12 1984-04-08    NA
## 13 1984-04-15    NA
## 14 1984-04-22    NA
## 15 2005-10-16    NA
```

```
## 16 2008-06-29    NA
## 17 2008-07-06    NA
## 18 2008-07-13    NA
```

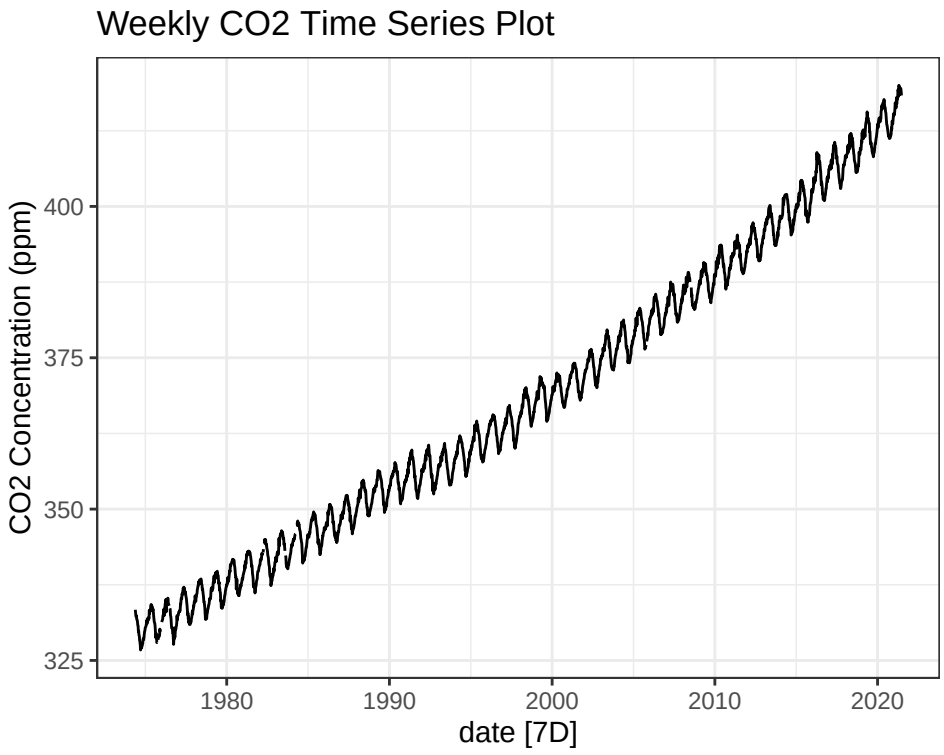
Summary of the data:

- The weekly time series runs from May 19, 1974 to June 20, 2021 and contains 2,458 values.
- There are 18 missing values in the series that were coded as -999.99, and they are not randomly scattered throughout the series. For instance, we are missing the four weeks that make up December 1975 and four straight weeks in April 1984 (among 10 others). This has implications for how we will deal with these missing values.

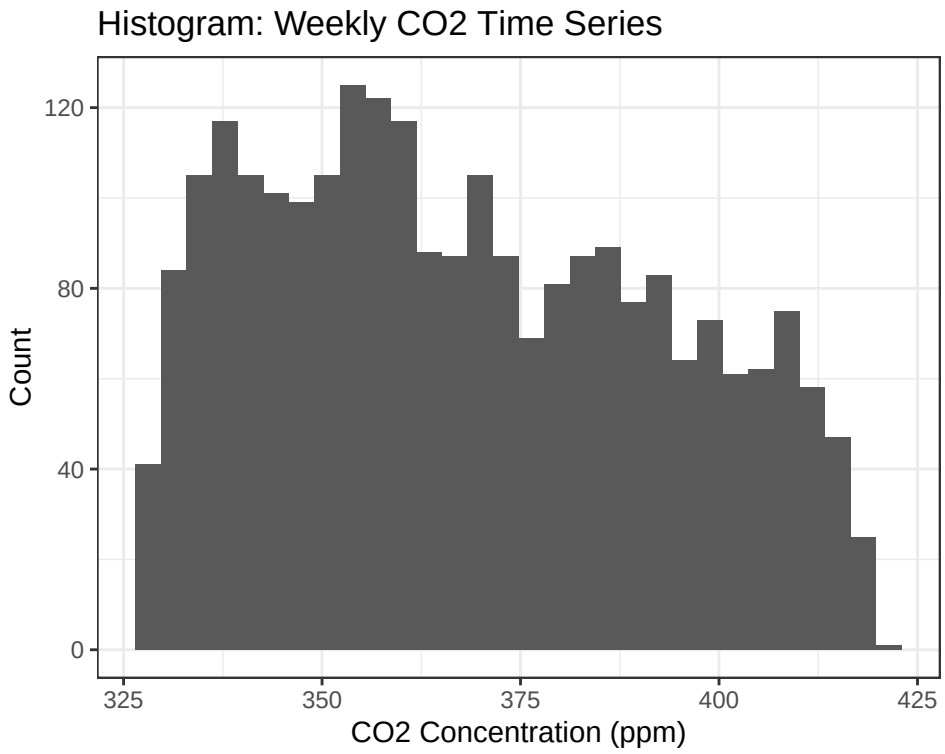
```
# Perform ETSDA
cbind(head(co2_weekly), tail(co2_weekly))
```

```
##         date    ppm       date    ppm
## 1 1974-05-19 333.37 2021-05-16 419.09
## 2 1974-05-26 332.95 2021-05-23 418.92
## 3 1974-06-02 332.35 2021-05-30 419.55
## 4 1974-06-09 332.20 2021-06-06 419.47
## 5 1974-06-16 332.37 2021-06-13 419.06
## 6 1974-06-23 331.73 2021-06-20 418.33
```

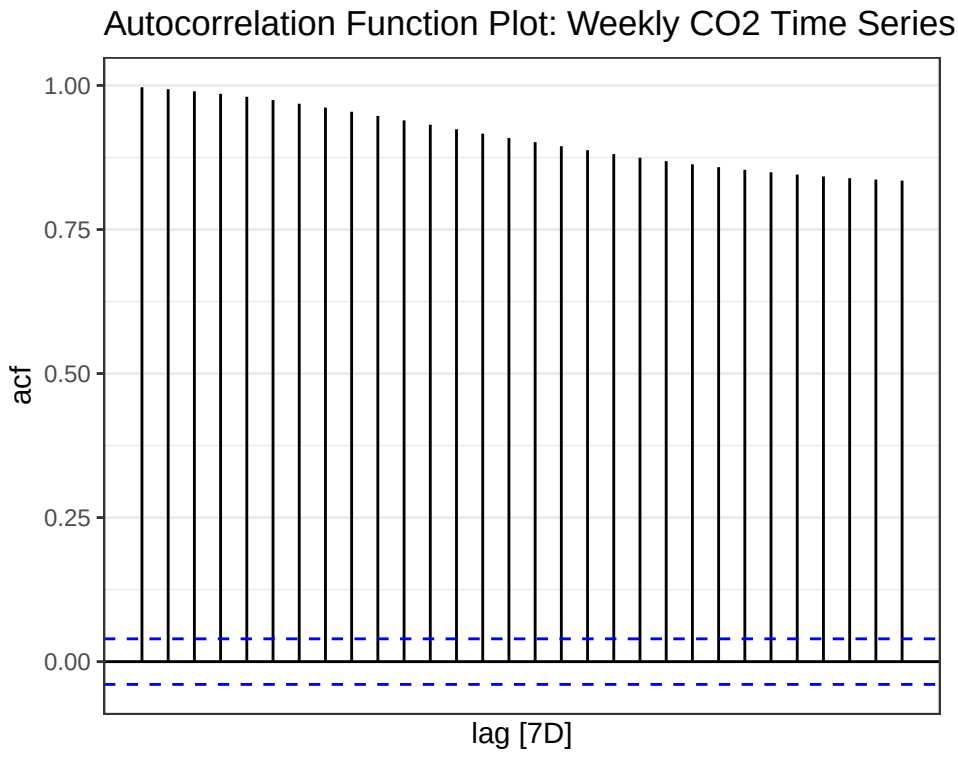
```
# Time plot to see dynamics
co2_weekly %>%
  autoplot() +
  labs(title = "Weekly CO2 Time Series Plot", y = "CO2 Concentration (ppm)") +
  theme_bw()
```



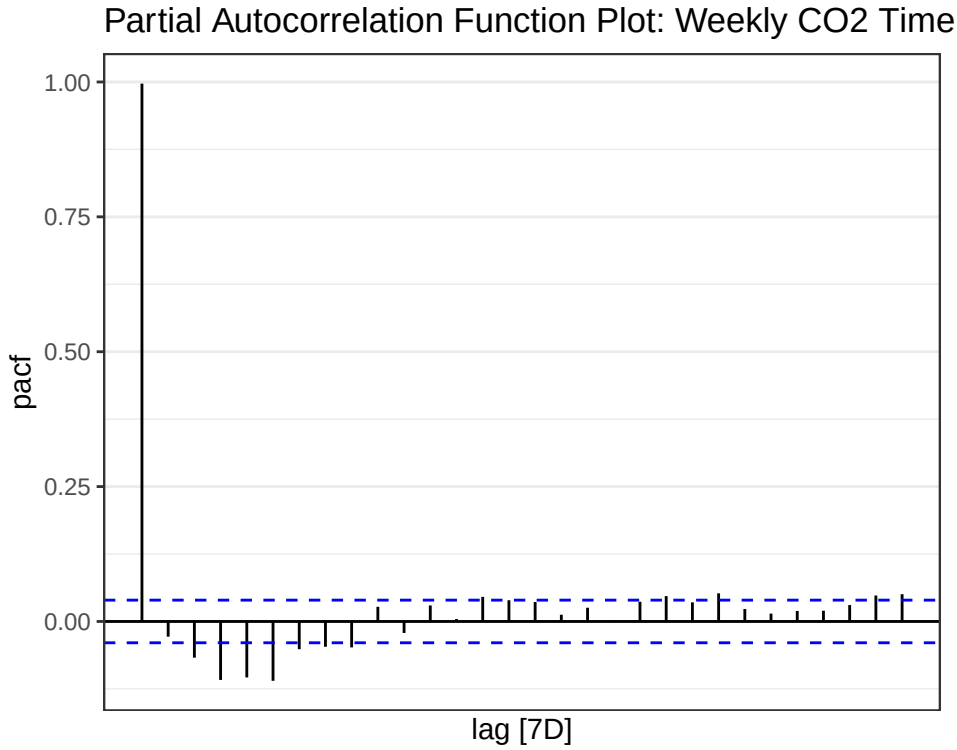
```r
# Histogram of values
data.frame(co2 = as.numeric(co2_weekly$ppm)) %>%
  ggplot(aes(x = co2)) +
  geom_histogram(bins = 30) +
  labs(title = "Histogram: Weekly CO2 Time Series", y = "Count", x = "CO2 Concentration (ppm)") +
  theme_bw()
```




```r
# ACF plot
co2_weekly %>%
  ACF() %>%
  autoplot() + labs(title="Autocorrelation Function Plot: Weekly CO2 Time Series") +
  theme_bw()
```

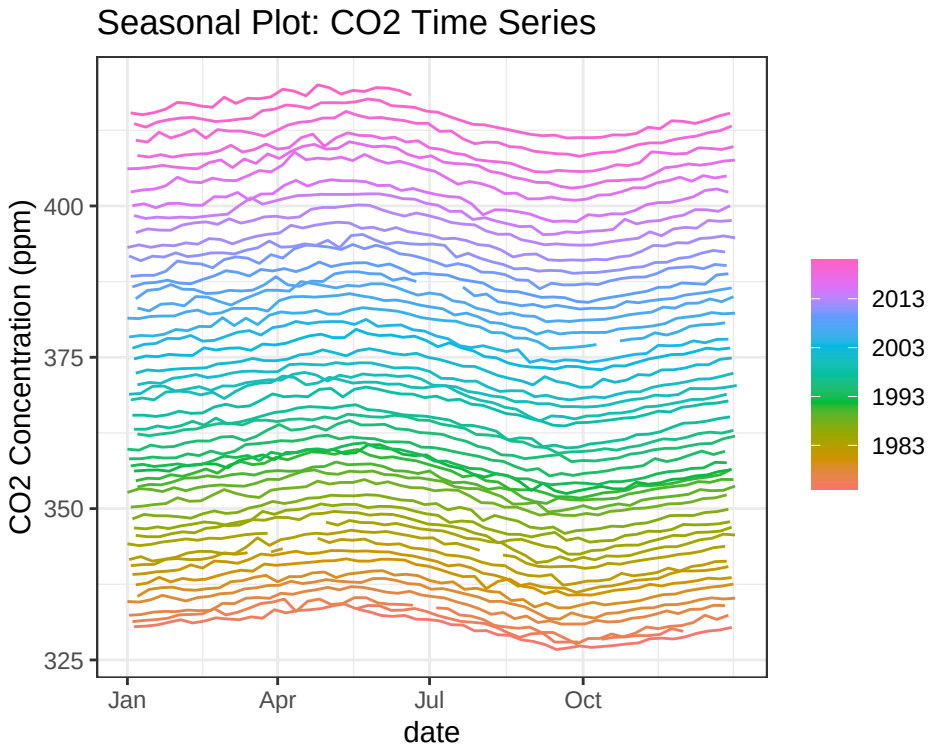



```
# PACF plot
co2_weekly %>%
  PACF() %>%
  autoplot() + labs(title="Partial Autocorrelation Function Plot: Weekly CO2 Time Series") +
  theme_bw()
```

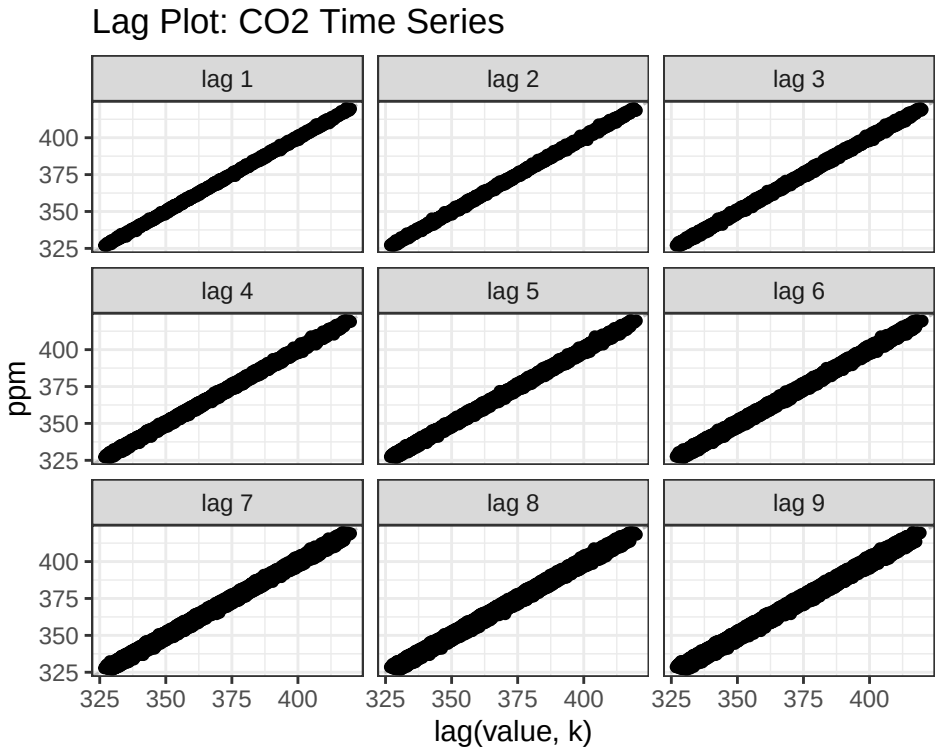


- The first 6 values and last 6 values in the data set appear reasonable relative to our expectations from the monthly data.
- The time series appears to be well-aligned in terms of trend and seasonality with the previous time series of monthly values.
- The distribution of ppm values in the time series is slightly right skewed like the monthly data.
- The waviness of the seasonal plot is indicative of the seasonality in the data, and the fact that the lines in each successive year tend to be higher for later years is expected.
- The ACF shows a long, slow decline.
- The PACF drops off initially, but then seems to oscillate around 0. There are undulations in the plot that are indicative of the seasonality in the data.

```
# Seasonal plot to check for seasonality
co2_weekly %>%
  gg_season(ppm, period = "year") +
  labs(y = "CO2 Concentration (ppm)", title = "Seasonal Plot: CO2 Time Series") +
  theme_bw()
```



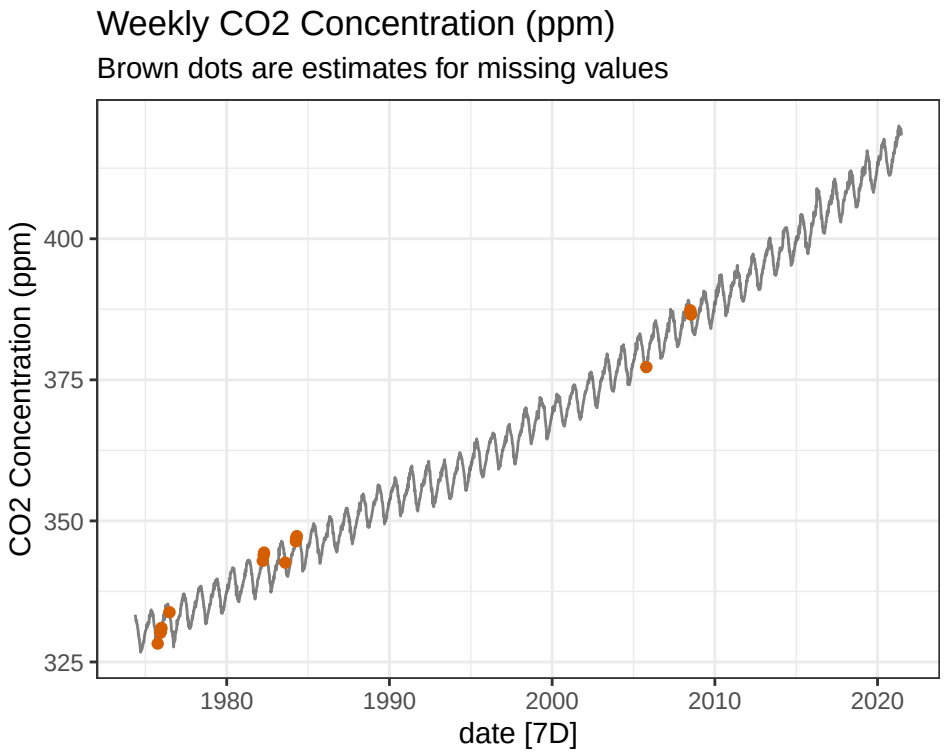

```
# Lag plot
co2_weekly %>%
  gg_lag(ppm, geom = "point") +
  labs(x = "lag(value, k)", title = "Lag Plot: CO2 Time Series") +
  theme_bw()
```



The lag plot shows very strong correlation in the time series at least through 9 lags, and the year-over-year seasonal plot shows strong evidence of seasonality in the data.

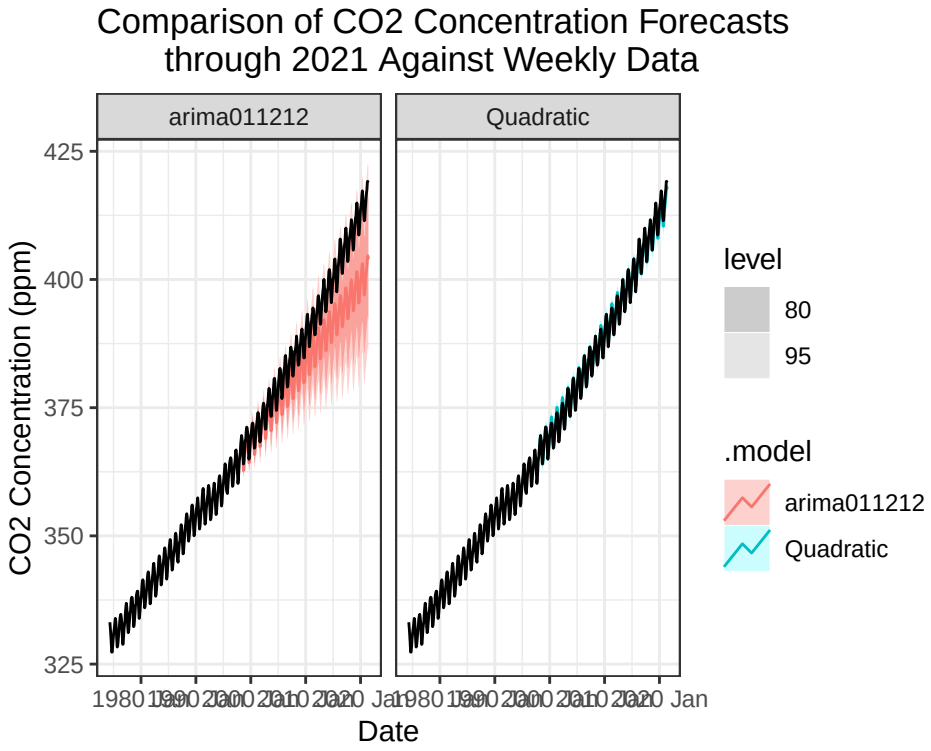
As previously mentioned, there are time frames in the series where we are missing successive values. Therefore, imputation by means of simple interpolation may not be appropriate. Instead, we auto-fit an ARIMA model to the time series (including the missing values) and then use that model to predict (*i.e.*, fill in) the missing values.

```
co2_weekly_fill <- co2_weekly %>%
  model(ARIMA(ppm)) %>%
  interpolate(co2_weekly) %>%
  as_tsibble()

co2_weekly_fill %>%
  autoplot(ppm, alpha = 0.5) +
  geom_point(
    data = left_join(
      co2_weekly %>% filter(is.na(ppm)),
      co2_weekly_fill,
      by = c("date" = "date")) %>%
      dplyr::select(date, ppm.y),
    aes(y = ppm.y), colour="#D55E00") +
  labs(title = "Weekly CO2 Concentration (ppm)",
       subtitle = "Brown dots are estimates for missing values",
       y = "CO2 Concentration (ppm)") +
  theme_bw()
```



Based on the plot with estimated ppm values, the predictions from the ARIMA model appear reasonable in light of the overall time series. Next, we aggregate this filled-in weekly series into a monthly series so that we can evaluate the model forecasts from parts 2 and 3.

```
h <- (12 * (2021 - 1998 + 1)) - 6 # Need to get to June 2021

# Convert weekly series to month-average series
co2_monthly <- co2_weekly_fill %>%
  index_by(index = yearmonth(date)) %>%
  summarise(ppm = mean(ppm))

# Generate object with both forecasts, using `ppm` as dependent variable name
poly_arima_models <- co2.ts %>%
  rename(ppm = value) %>%
  model(arima011212 = ARIMA(ppm ~ pdq(0, 1, 1) + PDQ(0, 1, 1)),
        Quadratic = TSLM(ppm ~ trend() + I(trend() ^ 2) + season()))

fabletools::forecast(poly_arima_models, h = h) %>%
  autoplot(co2_monthly) +
  facet_grid(. ~ .model) +
  labs(title = "Comparison of CO2 Concentration Forecasts
       through 2021 Against Weekly Data",
       x = "Date", y = "CO2 Concentration (ppm)") +
  theme_bw()
```



We see from the plot that the ARIMA(0,1,1)(0,1,1)[12] model underestimates the CO2 concentration with its forecast to the present, whereas the quadratic model actually appears to fit the data quite well.

```
# Examining the error terms between the two models
poly_arima_models %>%
  fabletools::forecast(h = h) %>%
  accuracy(
    data = co2_monthly
  ) %>% arrange(RMSE)
```

```
## # A tibble: 2 x 10
##   .model      .type      ME  RMSE   MAE     MPE  MAPE  MASE RMSSE  ACF1
##   <chr>       <chr>   <dbl> <dbl> <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Quadratic   Test  -0.0770 0.783 0.628 -0.0241 0.160 0.426 0.499 0.846
## 2 arima011212 Test   6.00   7.36  6.00   1.50   1.50  4.07  4.69  0.984
```

We can check the accuracy of our models against a slew of forecast accuracy measures using the `accuracy()` function. Focusing in on the root-mean squared error (RMSE) of the forecast period, we see that the RMSE of the quadratic model (0.78) is an order of magnitude lower than that of the ARIMA model (7.36). A similar case can be made for the mean absolute error (MAE) values, the mean absolute percentage error (MAPE) values and the mean percentage error (MPE) values between the models. The Quadratic model has improved accuracy over the ARIMA model by any number of accuracy metrics. This appears to be a case where a simpler linear time series model with quadratic and seasonal terms actually outperforms the more complex seasonal ARIMA model, lending credence to the idea that more complex time series models are not always better.

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?

For this exercise we'll continue to use the month-averaged data from part (4). We first split the model into the training set (everything up to June 2019) and the test set (July 2019 to June 2021). Similar to our approach in part (3), we used a function to fit a series of seasonal ARIMA models across a parameter range that specified d and D as 1, and allowed all other parameters to range from 0 to 3. The AIC, AICc, and BIC for each of the models that converged are read in below with the full data table of fitted models saved to the attached csv. We first sort by AIC and then by BIC to see which models are best according to those metrics.

```
# Split the data into training and test sets
co2_train <- co2_monthly %>%
  filter_index(. ~ "Jun 2019")
co2_test <- co2_monthly %>%
  filter_index("Jul 2019" ~ .)

# Iterate through seasonal ARIMA models to find the best
# best.arimas.noaa <- check.arimas(as.ts(co2_train), maxord = c(3, 1, 3, 3, 1, 3))
best.arimas.noaa <- read.csv("w271_lab2_part5.csv")
best.arimas.noaa %>% arrange(AIC) %>% head(10)
```

```
##       X p d q P D Q      AIC     AICc      BIC
## 1   408 1 1 1 0 1 1 317.0207 317.0970 334.1046
## 2   193 0 1 2 0 1 1 317.1044 317.1807 334.1883
## 3   161 0 1 1 0 1 1 317.1642 317.2099 329.9772
## 4   409 1 1 1 0 1 2 318.0965 318.2113 339.4515
## 5   416 1 1 1 1 1 1 318.0996 318.2143 339.4545
## 6   169 0 1 1 1 1 1 318.1227 318.1991 335.2067
## 7   162 0 1 1 0 1 2 318.1358 318.2121 335.2197
## 8   194 0 1 2 0 1 2 318.1782 318.2929 339.5331
## 9   201 0 1 2 1 1 1 318.1809 318.2957 339.5359
## 10 187 0 1 1 3 1 3 318.3633 318.6402 352.5312
```

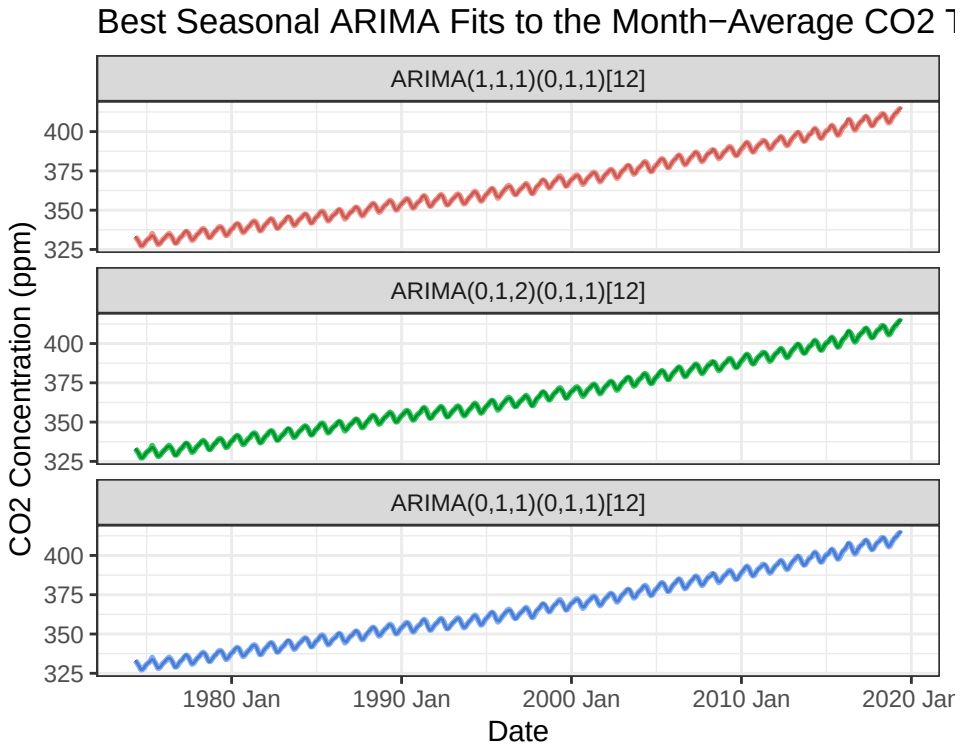
```
best.arimas.noaa %>% arrange(BIC) %>% head(10)
```

```
##       X p d q P D Q      AIC     AICc      BIC
## 1   161 0 1 1 0 1 1 317.1642 317.2099 329.9772
## 2   408 1 1 1 0 1 1 317.0207 317.0970 334.1046
## 3   193 0 1 2 0 1 1 317.1044 317.1807 334.1883
## 4   169 0 1 1 1 1 1 318.1227 318.1991 335.2067
## 5   162 0 1 1 0 1 2 318.1358 318.2121 335.2197
## 6   409 1 1 1 0 1 2 318.0965 318.2113 339.4515
## 7   416 1 1 1 1 1 1 318.0996 318.2143 339.4545
## 8   194 0 1 2 0 1 2 318.1782 318.2929 339.5331
## 9   201 0 1 2 1 1 1 318.1809 318.2957 339.5359
```

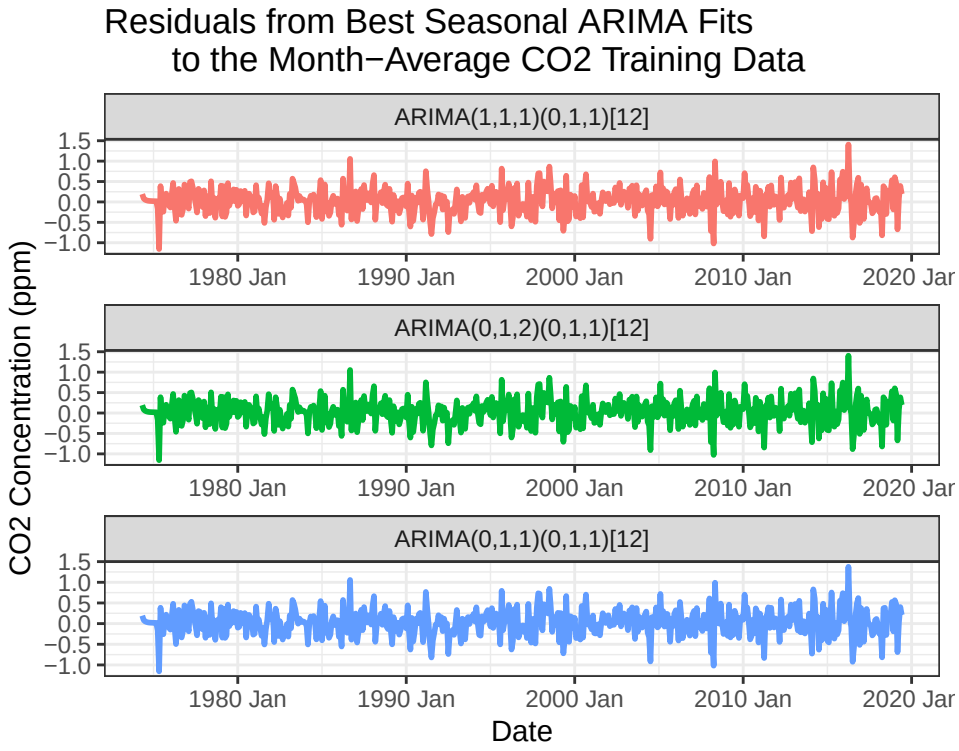
```
## 10 225 0 1 3 0 1 1 318.8662 318.9809 340.2211
```

We can see that the three top models according to AIC—ARIMA(1,1,1)(0,1,1)[12], ARIMA(0,1,2)(0,1,1)[12], and ARIMA(0,1,1)(0,1,1)[12]—are all within 0.1435 AIC units, whereas the 4th-best model has an AIC that is 0.9323 AIC units above the 3rd-best model. The same three models are also the top three models according to BIC, albeit in a different order. According to BIC, the ARIMA(0,1,1)(0,1,1)[12] seasonal ARIMA model is the favorite (the BIC value of 329.98 is over 4 BIC units lower than the next best model), additionally, it's the most parsimonious of the top 3 models. The next step is to explore the fits and residuals of our three best models

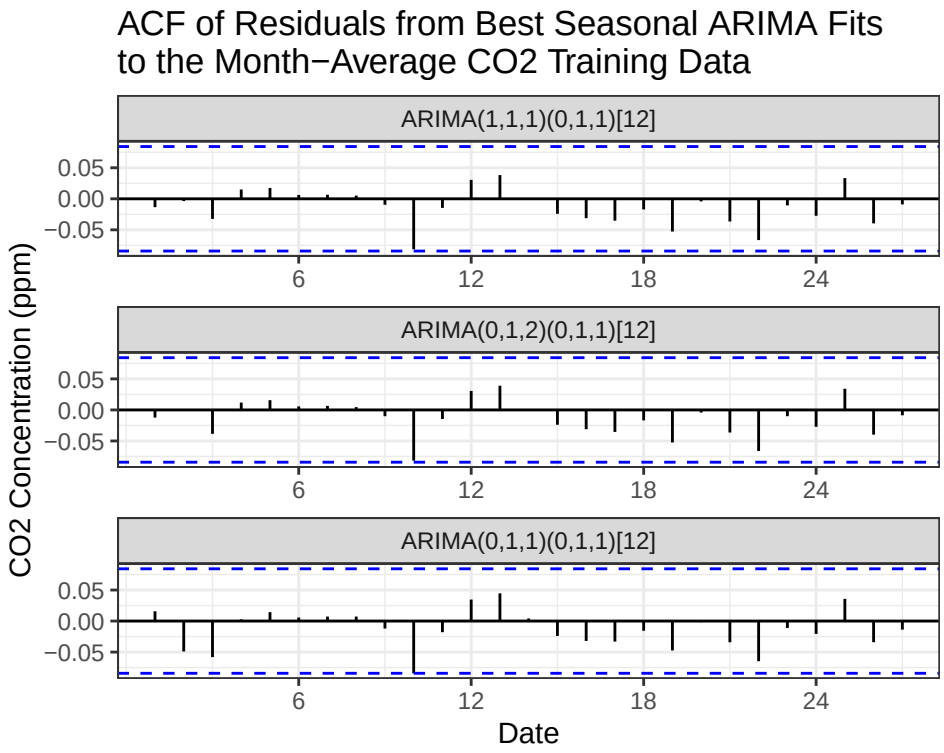
```
three_arima_models <- co2_train %>%
  model(arima111011 = ARIMA(ppm ~ pdq(1, 1, 1) + PDQ(0, 1, 1)),
        arima012011 = ARIMA(ppm ~ pdq(0, 1, 2) + PDQ(0, 1, 1)),
        arima011011 = ARIMA(ppm ~ pdq(0, 1, 1) + PDQ(0, 1, 1)))

co2_train %>%
  autoplot(ppm) +
  geom_line(data = fitted(three_arima_models) %>%
              mutate(.model = ifelse(.model == "arima111011", "ARIMA(1,1,1)(0,1,1)[12]",
                                     ifelse(.model == "arima012011", "ARIMA(0,1,2)(0,1,1)[12]"
                                            "ARIMA(0,1,1)(0,1,1)[12]"))) %>%
              mutate(.model = factor(
                .model, levels = c("ARIMA(1,1,1)(0,1,1)[12]",
                                   "ARIMA(0,1,2)(0,1,1)[12]",
                                   "ARIMA(0,1,1)(0,1,1)[12]"))),
            aes(x = index, y = .fitted, colour = .model),
            size = 1, alpha = 0.8) +
  facet_wrap(. ~ .model, nrow = 3) +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Date", y = "CO2 Concentration (ppm)",
       title = "Best Seasonal ARIMA Fits to the Month-Average CO2 Training Data")
```



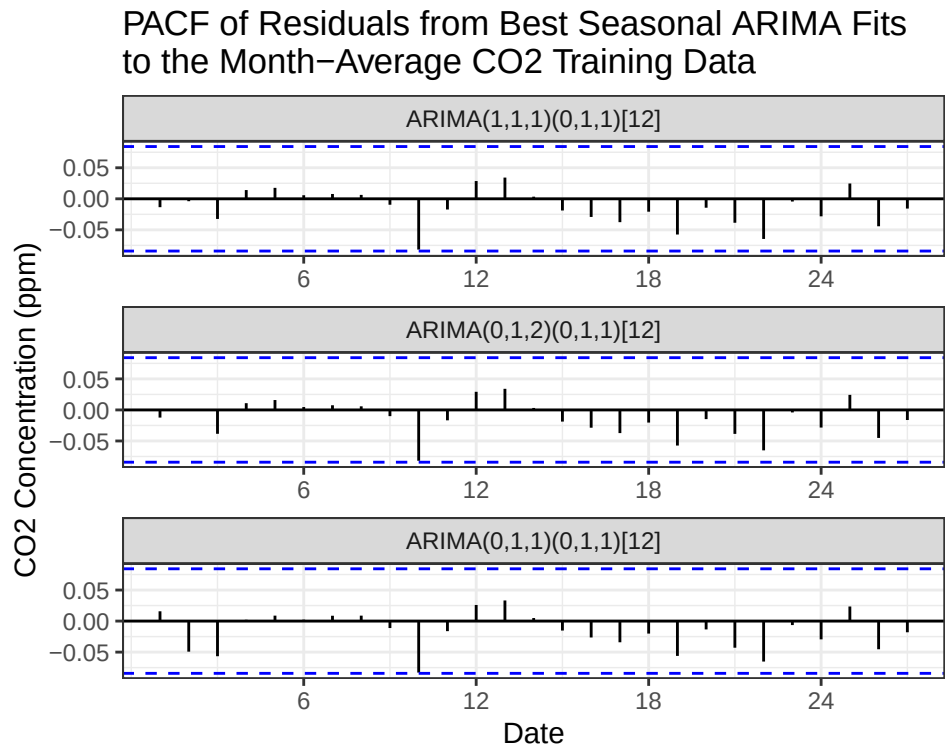

```
resid(three_arima_models) %>%
  mutate(.model = ifelse(.model == "arima111011", "ARIMA(1,1,1)(0,1,1)[12]",
                                      ifelse(.model == "arima012011", "ARIMA(0,1,2)(0,1,1)[12]"
                                             "ARIMA(0,1,1)(0,1,1)[12]"))) %>%
  mutate(.model = factor(.model, levels = c("ARIMA(1,1,1)(0,1,1)[12]",
                                    "ARIMA(0,1,2)(0,1,1)[12]",
                                    "ARIMA(0,1,1)(0,1,1)[12]"))) %>%
  ggplot(aes(x = index, y = .resid, colour = .model)) +
  geom_line(size = 1) +
  facet_wrap(. ~ .model, nrow = 3, scales = "free") +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Date", y = "CO2 Concentration (ppm)",
       title = "Residuals from Best Seasonal ARIMA Fits
       to the Month-Average CO2 Training Data")
```



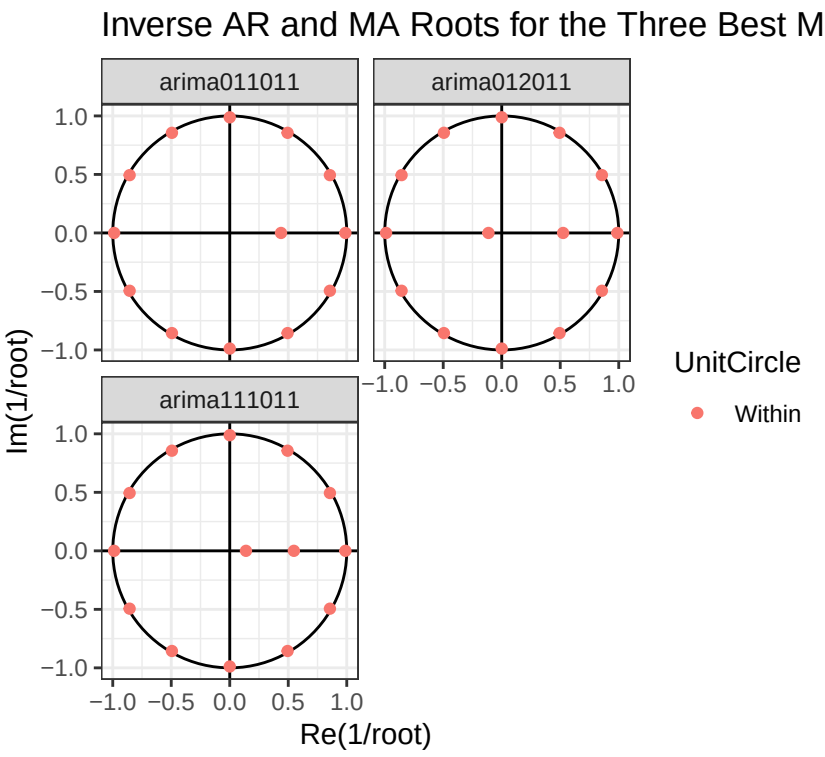

```
augment(three_arima_models) %>%
  mutate(.model = ifelse(.model == "arima111011", "ARIMA(1,1,1)(0,1,1)[12]",
                                    ifelse(.model == "arima012011", "ARIMA(0,1,2)(0,1,1)[12]"
                                           "ARIMA(0,1,1)(0,1,1)[12]"))) %>%
  mutate(.model = factor(.model, levels = c("ARIMA(1,1,1)(0,1,1)[12]",
                                    "ARIMA(0,1,2)(0,1,1)[12]",
                                    "ARIMA(0,1,1)(0,1,1)[12]"))) %>%
  ACF(.resid) %>%
  autoplot() +
  facet_wrap(. ~ .model, nrow = 3, scales = "free") +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Date", y = "CO2 Concentration (ppm)",
       title = "ACF of Residuals from Best Seasonal ARIMA Fits\nto the Month-Average CO2 Train
```




```
augment(three_arima_models) %>%
  mutate(.model = ifelse(.model == "arima111011", "ARIMA(1,1,1)(0,1,1)[12]",
                                      ifelse(.model == "arima012011", "ARIMA(0,1,2)(0,1,1)[12]"
                                             "ARIMA(0,1,1)(0,1,1)[12]"))) %>%
  mutate(.model = factor(.model, levels = c("ARIMA(1,1,1)(0,1,1)[12]",
                                    "ARIMA(0,1,2)(0,1,1)[12]",
                                    "ARIMA(0,1,1)(0,1,1)[12]"))) %>%
  PACF(.resid) %>%
  autoplot() +
  facet_wrap(. ~ .model, nrow = 3, scales = "free") +
  theme_bw() +
  theme(legend.position = "none") +
  labs(x = "Date", y = "CO2 Concentration (ppm)",
       title = "PACF of Residuals from Best Seasonal ARIMA Fits\nto the Month-Average CO2 Trai
```




```
three_arima_models %>%
  gg_arma() +
  facet_wrap(.model ~ ., nrow = 2) +
  theme_bw() +
  labs(title = "Inverse AR and MA Roots for the Three Best Models")
```



The first plot shows how the three models compare to the data they were trained on. Based on the fit as well as the plots of the residuals, there is no discernible difference between the three models. Similarly the ACF and PACF of the residuals reveals effectively no difference between the models and none of the lags are significant indicating they are results from a random noise process. Finally, the inverse AR and MA roots for all three models are *very* close to the unit circle. While this suggests instability and potential bias in forecasts, none of the models appears to be strictly better than the others on this aspect.

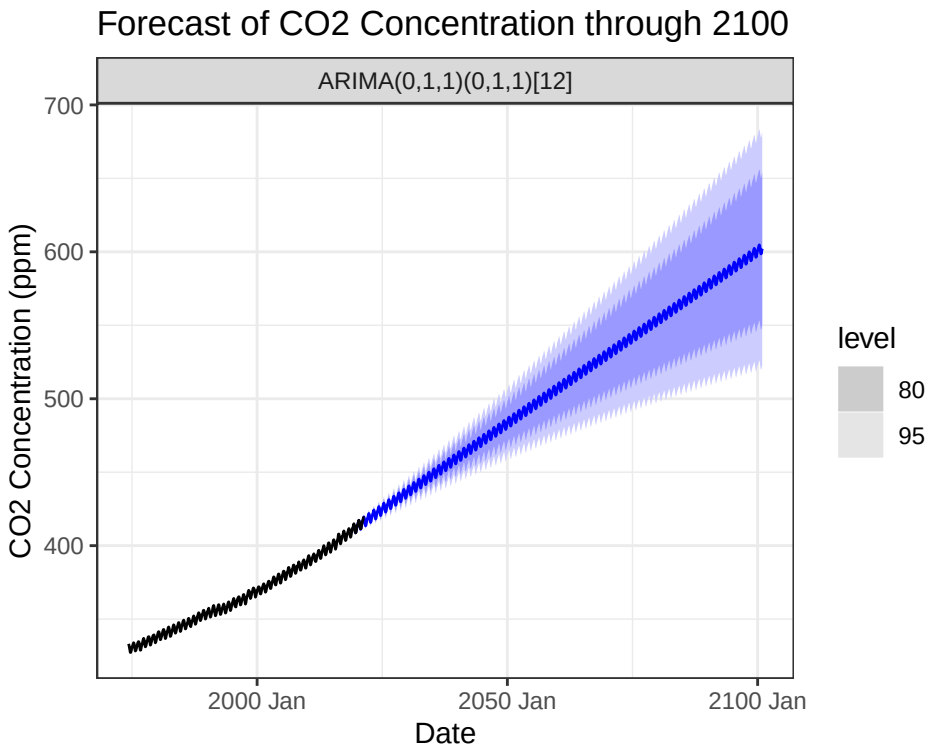
The final point of comparison will measure the accuracy of the models against our testing data using RMSE as the primary measure of accuracy.

```
three_arima_models %>%
 fabletools::forecast(h = 24) %>%
 accuracy(
   data = co2_test
 ) %>% arrange(RMSE)
```

```
## # A tibble: 3 x 10
##   .model      .type       ME  RMSE   MAE       MPE   MAPE  MASE RMSSE   ACF1
##   <chr>       <chr>    <dbl> <dbl> <dbl>     <dbl>  <dbl> <dbl> <dbl>  <dbl>
## 1 arima011011 Test  -0.0271  0.250 0.199 -0.00671  0.0481   NaN   NaN 0.0539
## 2 arima012011 Test   0.00223 0.251 0.198  0.000374 0.0478   NaN   NaN 0.0630
## 3 arima111011 Test   0.00553 0.252 0.198  0.00117  0.0478   NaN   NaN 0.0647
```

The ARIMA(0,1,1)(0,1,1)[12] model achieves the lowest RMSE (which tracks the mean of the time series) on the test data, with the other two models having slightly higher RMSE values. Conversely, the ARIMA(0,1,1)(0,1,1)[12] has a slightly higher mean absolute error (which tracks the median of the time series) than the other two models. But, of the three models the ARIMA(0,1,1)(0,1,1)[12] is the most parsimonious, which makes it our top choice for conducting this 80-year forecast.

```
# Determine the forecasting length
h = 6 + ((2100 - 2020 + 1) * 12)

# Fit the model with our parameters
best_model <- co2_train %>%
  model(arima011011 = ARIMA(ppm ~ pdq(0, 1, 1) + PDQ(0, 1, 1)))

# Generate Predictions
forecast_2100 <- fabletools::forecast(best_model, h = h)

# Plot Predictions
forecast_2100 %>%
  mutate(.model = "ARIMA(0,1,1)(0,1,1)[12]") %>%
  autoplot(co2_monthly) +
  facet_grid(. ~ .model) +
  labs(title = "Forecast of CO2 Concentration through 2100",
       x = "Date", y = "CO2 Concentration (ppm)") +
  theme_bw()
```




```
# Isolate where it reaches 450ppm for the first time
forecast_2100 %>%
  filter_index("May 2034") %>%
  hilo(level = 95) %>%
  unpack_hilo("95%")
```

```
## # A tsibble: 1 x 6 [?]
## # Key:       .model [1]
##   .model          index        ppm .mean `95%_lower` `95%_upper`
##   <chr>           <mth>     <dist> <dbl>       <dbl>       <dbl>
## 1 arima011011 2034 May N(450, 24)  450.        440.        460.
```

```
# Isolate the forecast for Dec 2100
forecast_2100 %>%
  filter_index("Dec 2100") %>%
  hilo(level = 95) %>%
  unpack_hilo("95%")
```

```
## # A tsibble: 1 x 6 [?]
## # Key:       .model [1]
##   .model          index          ppm .mean `95%_lower` `95%_upper`
##   <chr>           <mth>       <dist> <dbl>       <dbl>       <dbl>
## 1 arima011011 2100 Dec N(602, 1640)  602.        523.        682.
```

The estimated model predicts that the earths CO2 concentration will first reach 450 ppm by May 2034, with a 95% confidence interval for that month between 440 ppm and 460 ppm. The model predicts that by December 2100, the earth's CO2 concentration will be 602 ppm with a 95% confidence interval of 522 to 681.

We are trying to use this model to make predictions for 80 years into the future. This is problematic for multiple reasons:

- Using such a long forecast horizon is simply unreasonable. The time series we trained the ARIMA model on only has approximately 45 years of data; so, in this exercise we are forecasting over a window that is nearly twice the length of the original time series our model was trained on.
- There are geologic/natural and political influences that will likely cause the steady trend we see in the training data to deviate (for better or worse) within the next 80 years. We cannot capture such influences in our forecast.
- For predictions in the year 2100, we see that the 95% confidence intervals are roughly 160 ppm wide (or 25% of the predicted atmospheric CO2 concentration at that time), indicating minimal confidence in the predictions being accurate. Additionally, as we saw earlier, the ARIMA model did a worse job at forecasting from 1998 to the present, whereas the quadratic model did a much better job of accurately forecasting the time series over that time period. This does not mean the quadratic model would clearly be better for an 80-year forecast such as this, but it does further draw into question the validity of this ARIMA model's forecasts for this time series.