ECON 144: Project 2

Amos Tong, Shreesh Agarwal, Kevin Tian, and Matthew Craig

Contents


```
rm(list=ls(all=TRUE))
library(cat)
library(png)
library(SnowballC)
library(LDAvis)
library(dplyr)
library(stringi)
library(plyr)
```

```r
library(foreign)
library(xts)
library(tis)
library(jsonlite)
library(FNN)
library(hexbin)
library(RColorBrewer)
library(MASS)
library(quantmod)
library(foreign)
library(MASS)
library(TTR)
library(vars)
library(readtext)
library(tidyr)
library(scales)
library(fitdistrplus)
library(xtable)
library(effects)
library(broom)
library(stats)
library(sandwich)
library(stargazer)
library(leaps)
library(tidyverse)
library(moments)
library(lmtest)
library(tseries)
library(fabletools)
library(restriktor)
library(tseries)
library(forecast)
library(fpp3)
library(tseries)
library(seasonal)
library(moments)
library(ggplot2)
library(feasts)
library('KFAS')
library('FKF')
```

I. Introduction

Datasets chosen for this project:

1. Vehicle Miles Traveled (01-01-2000 to 12-31-2019)

> Vehicle Miles Traveled (monthly millions of miles, not seasonally adjusted) is sourced from the U.S. Federal Highway Administration. A vehicle mile traveled represents one vehicle traveling one mile on public roads anywhere within the 50 states.

2. Rail Passenger Miles (01-01-2000 to 12-31-2019)

Rail passenger-miles (monthly miles, not seasonally adjusted) represent the movement of 1 passenger for 1 mile. The data is gathered by the U.S. Department of Transportation, Federal Railroad Administration, and published by the U.S. Bureau of Transportation Statistics.

II. Results

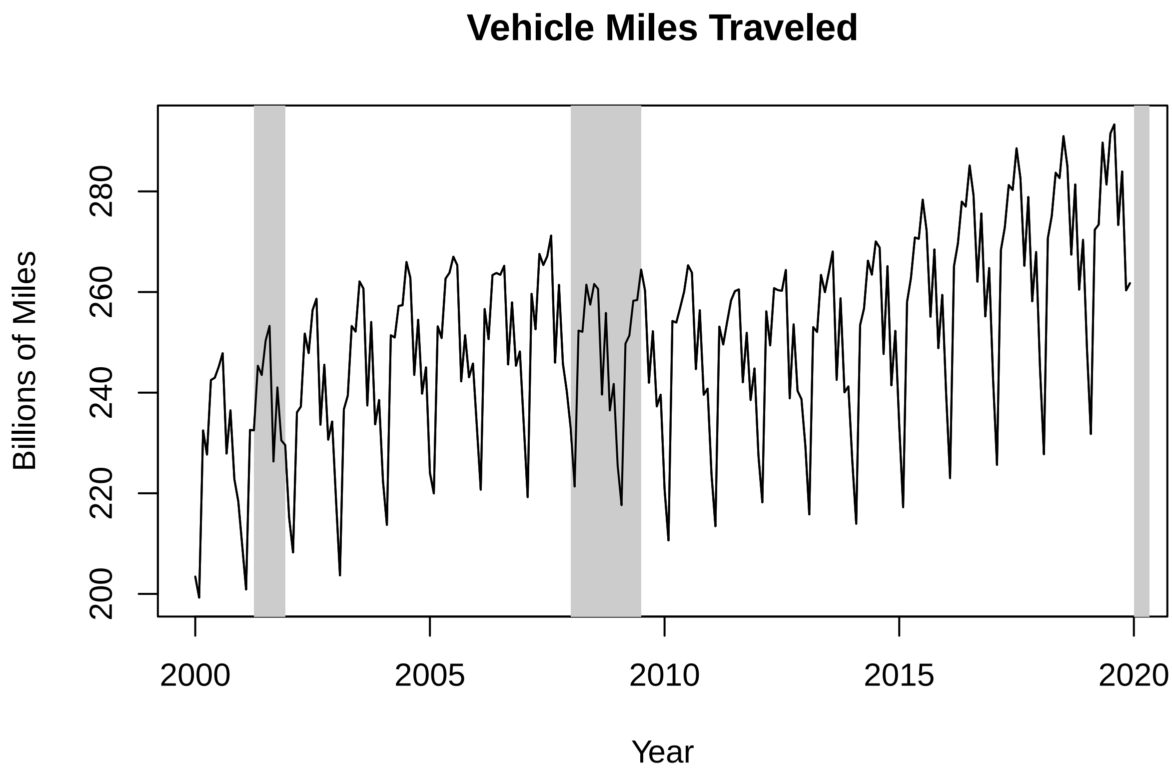
(a)

Produce a time-series plot of your data including the respective ACF and PACF plots.

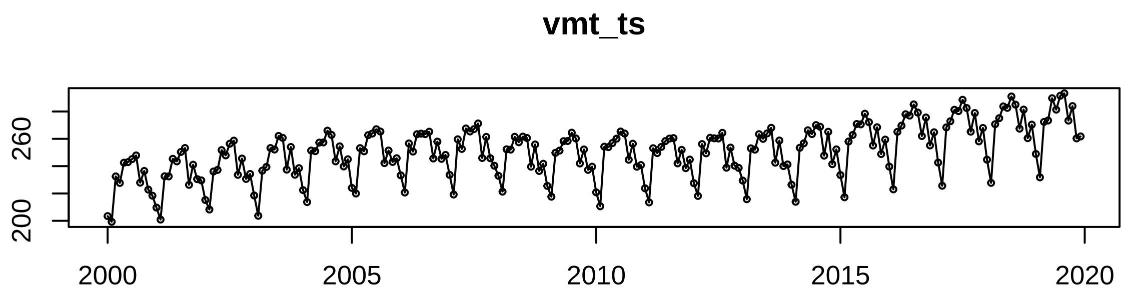
```
# Vehicle Miles Traveled
getSymbols("TRFVOLUSM227NFWA", src = "FRED")
```

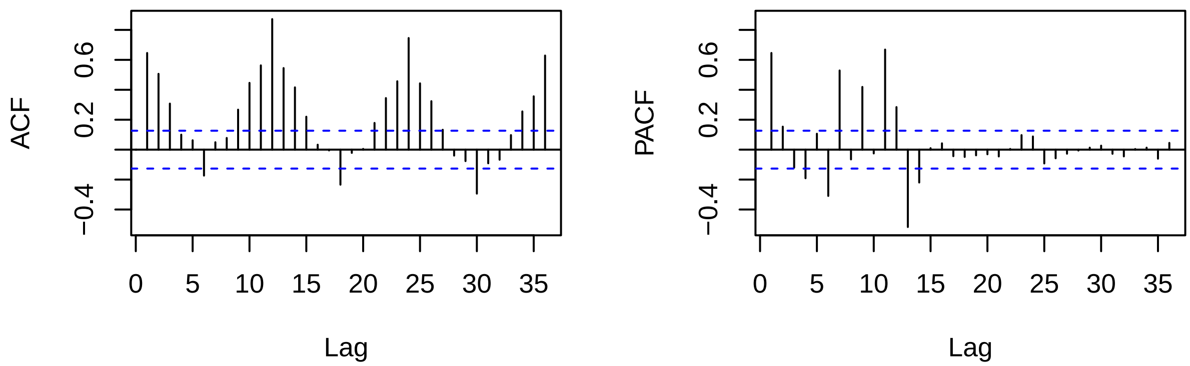
```
## [1] "TRFVOLUSM227NFWA"
```

```
vmt <- as.data.frame(TRFVOLUSM227NFWA[361:(length(TRFVOLUSM227NFWA)-24)])
vmt_ts <- ts(vmt$TRFVOLUSM227NFWA, start = 2000, freq = 12)

# Rescale the data (millions->billions)
vmt_ts <- vmt_ts / 1000

# Plot data
plot(vmt_ts, main = "Vehicle Miles Traveled", xlab = "Year", ylab = "Billions of Miles")
nberShade()
lines(vmt_ts)
```




```r
# View ACF and PACF
tsdisplay(vmt_ts)
```



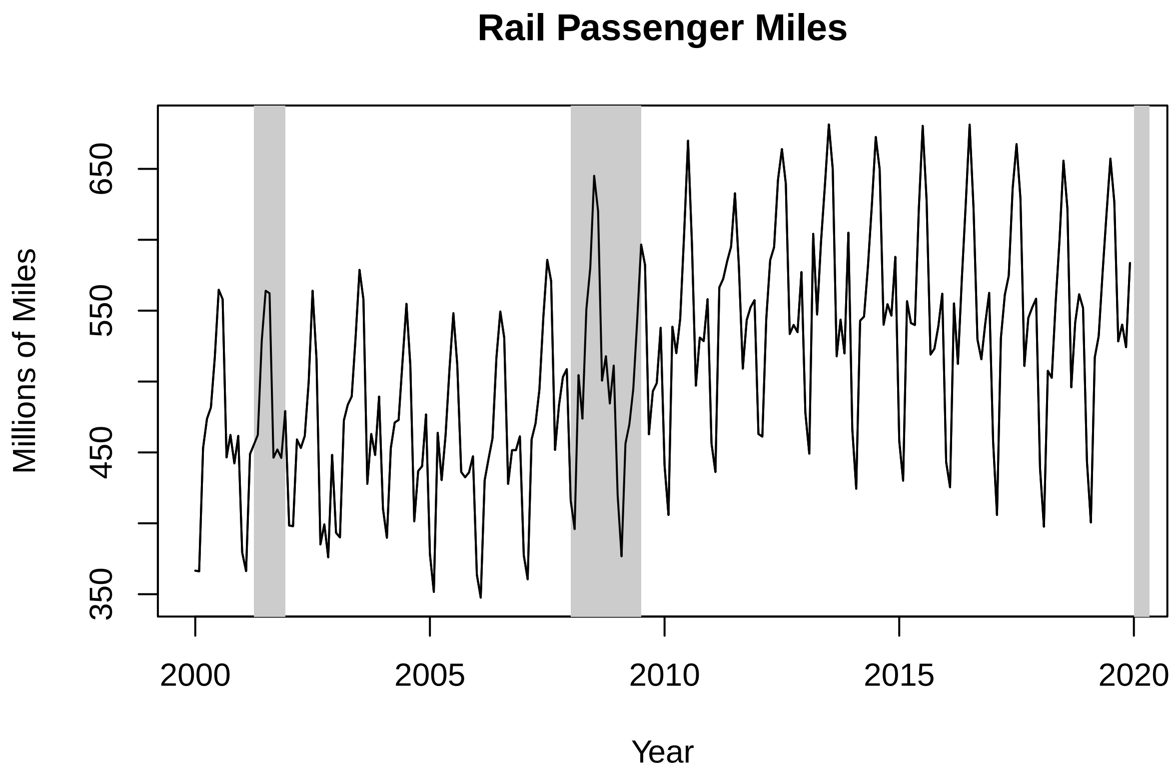


- The Vehicle Miles Traveled data does not appear to be covariance stationary since the mean of each variable is not stationary, but exhibits a clear upward trend.
- The data also shows strong (yearly) seasonality, seen in the regular peaks and troughs.
- The ACF plot decays slowly, indicating a high persistence. The ACF plot also confirms the seasonality of the data, as there are regular cycles that correspond to a yearly (12 period) frequency. There are a few strong spikes in the PACF out to about 13 or 14 months, potentially indicating some long term cycles in the data.
- These observations indicate that the data demonstrates serial correlation from one period to the next.

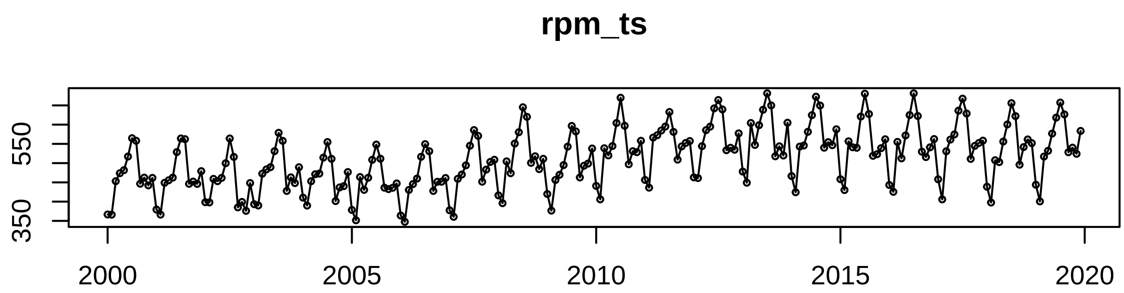
```
# Rail Passenger Miles
getSymbols("RAILPM", src = "FRED")
```

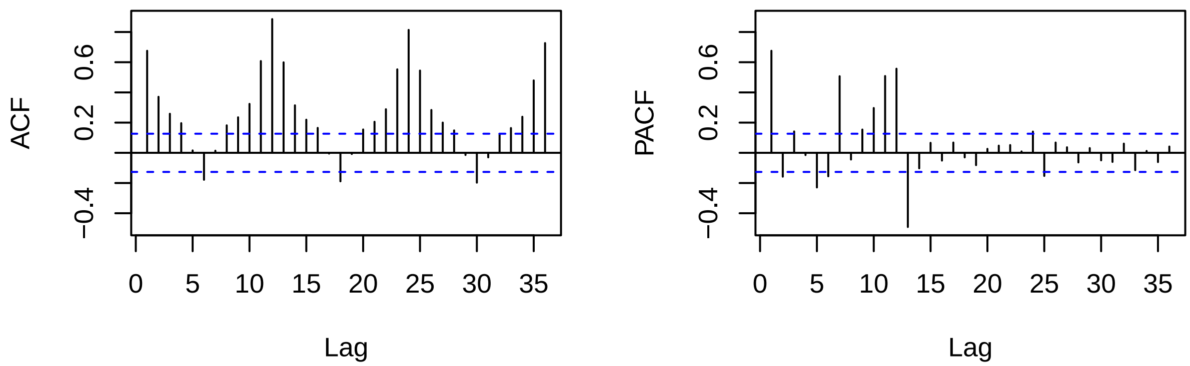
```
## [1] "RAILPM"
```

```
rpm <- as.data.frame(RAILPM[1:(length(vmt_ts))])
rpm_ts <- ts(rpm$RAILPM, start = 2000, freq = 12)

# Rescale the data (miles->millions)
rpm_ts <- rpm_ts / 1000000

# Plot data
plot(rpm_ts, main = "Rail Passenger Miles", xlab = "Year", ylab = "Millions of Miles")
nberShade()
lines(rpm_ts)
```




```
# View ACF and PACF
tsdisplay(rpm_ts)
```



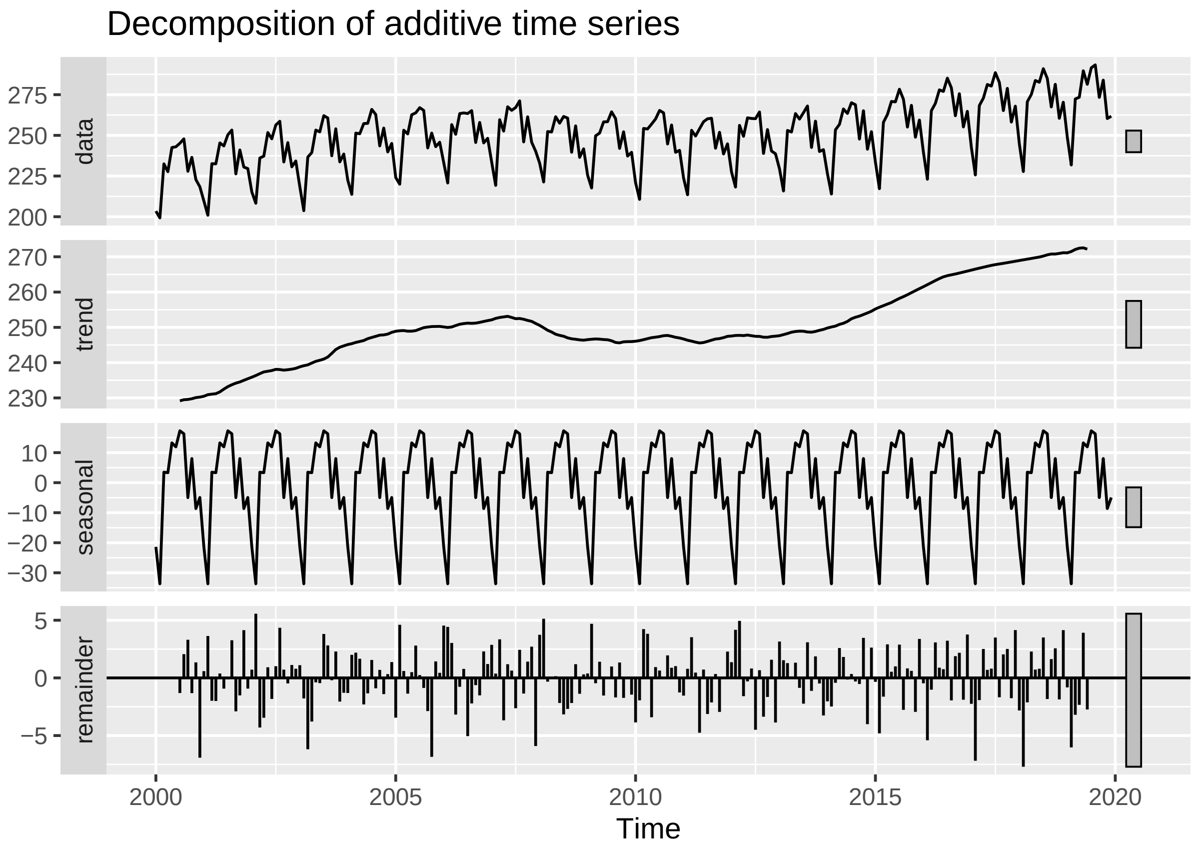


- Rail Passenger Miles data does not appear to be covariance stationary since the mean of each variable is not stationary but displays an upward trend. Additionally, the variance seems to increase slightly with time.
- The data also shows strong (yearly) seasonality, seen in the regular peaks and troughs.
- The ACF plot decays slowly, indicating a high persistence. The ACF plot also confirms the seasonality of the data, as there are regular cycles that correspond to a yearly (12 period) frequency. There are a few strong spikes in the PACF out to about 13 or 14 months, potentially indicating some long term cycles in the data.
- These observations indicate that the data demonstrates serial correlation from one period to the next.

(b)

Plot the stl decomposition plot of your data, and discuss the results.

```
# Vehicle Miles Traveled
vmt_stl <- decompose(vmt_ts, "additive")
autoplot(vmt_stl)
```

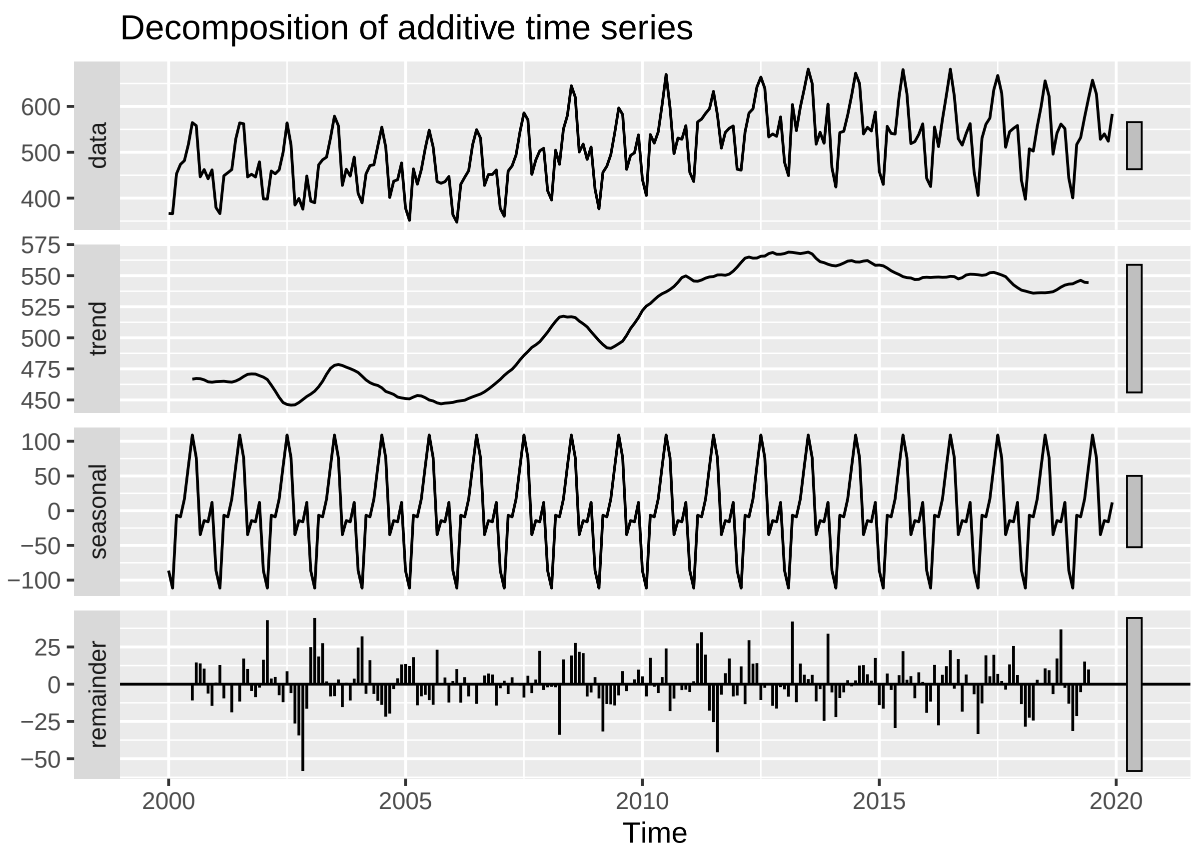


- We used an additive decomposition since the seasonal fluctuations does not appear to vary with time.
- The trend component confirms our previous analysis that there is a strong upward trend. The trend component is very significant as the units on the left axis are large. This means that the trend component explains a good proportion of the variation of the data. We can try fitting a cubic trend:

$$y = b_0 + b_1 t + b_2 t^2 + b_3 t^3$$

  with seasonal dummies and a model for the cycles.
- The seasonal component shows a strong seasonal effect on the data. The seasonal component is also significant compared to the trend component. This means that the seasonal component explains a significant amount of the variation of the data. We can use seasonal dummies to model the seasonal components.
- The remainder component shows that the remainder does contribute some amount to the data, but much less than the trend or season. This means that the remainder component explains little of the variation of the data. We do frequently see negative remainder components when the seasonality is at its lowest point, which means that the additive STL decomposition may not be capturing all of the variation due to seasonal fluctuations.
  - There may also be a cyclical component as the remainder somewhat fluctuates around zero at regular intervals. We would need to plot the ACF and PACF of the residuals to confirm presence of cycles.

```r
# Rail Passenger Miles
rpm_stl <- decompose(rpm_ts, type="additive")
autoplot(rpm_stl)
```



- We used an additive decomposition since the seasonal fluctuations appear to vary very slightly, if at all, with time.

- The trend component confirms our previous analysis that there is an upward trend, although the trend dampens from 2008 onwards. The trend component is very significant as the units on the left axis are large. This means that the trend component explains a good proportion of the variation of the data. We can again try fitting a cubic trend:

$$y = b_0 + b_1 t + b_2 t^2 + b_3 t^3$$

  with seasonal dummies and a model for the cycles, although the trend does not appear as close as with Vehicle Miles Traveled.

- The seasonal component shows a strong seasonal effect on the data. The seasonal component is also significant as the units on the left axis are large. This means that the seasonal component explains a good proportion of the variation of the data. We can use seasonal dummies to model the seasonal components.

- The remainder component shows that most of the time, the remainder does not contributes a significant effect on the data as the units on the left axis are smaller than trend and seasonal factors. However, there are some spikes where the remainder component accounts for a variation of 10% or more in the data. Some of these fluctuations occur where the trend deviates significantly, so the multiplicative STL decomposition may not be capturing the variation we want in the trend component.

- There may also be a cyclical component as the remainder fluctuates around zero at regular intervals.

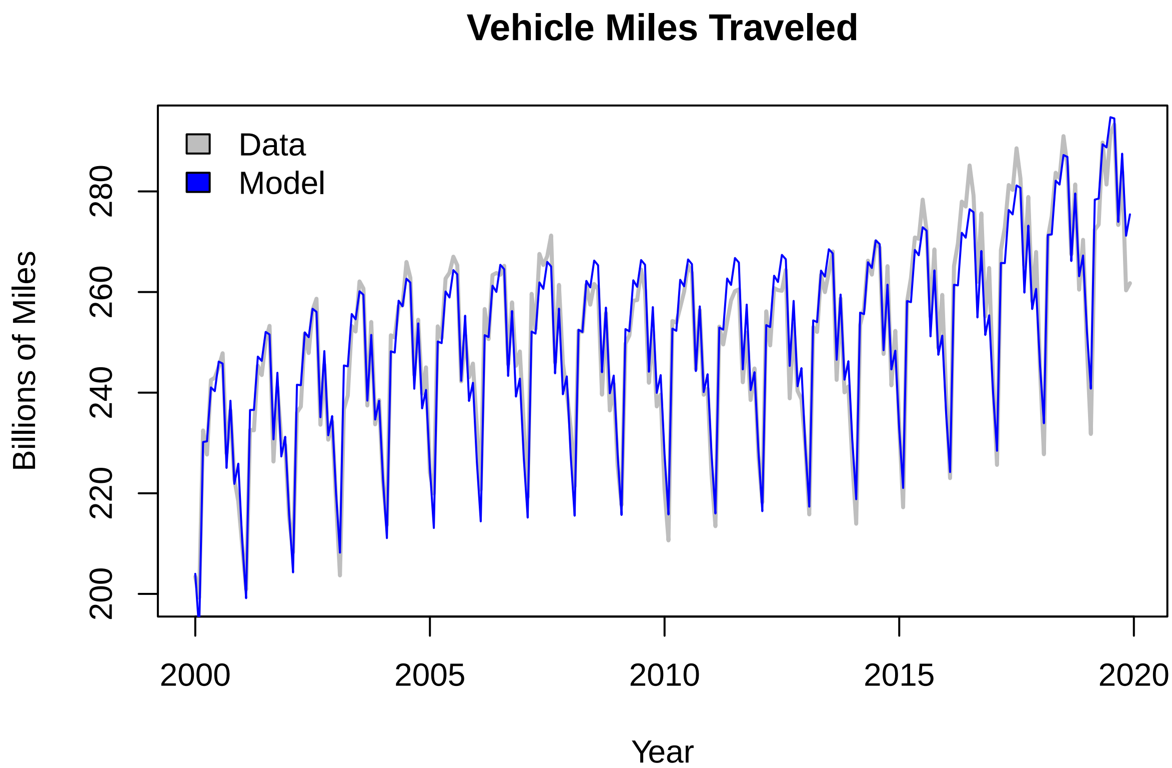
(c)

Fit a model that includes, trend, seasonality and cyclical components. Make sure to discuss your model in detail.

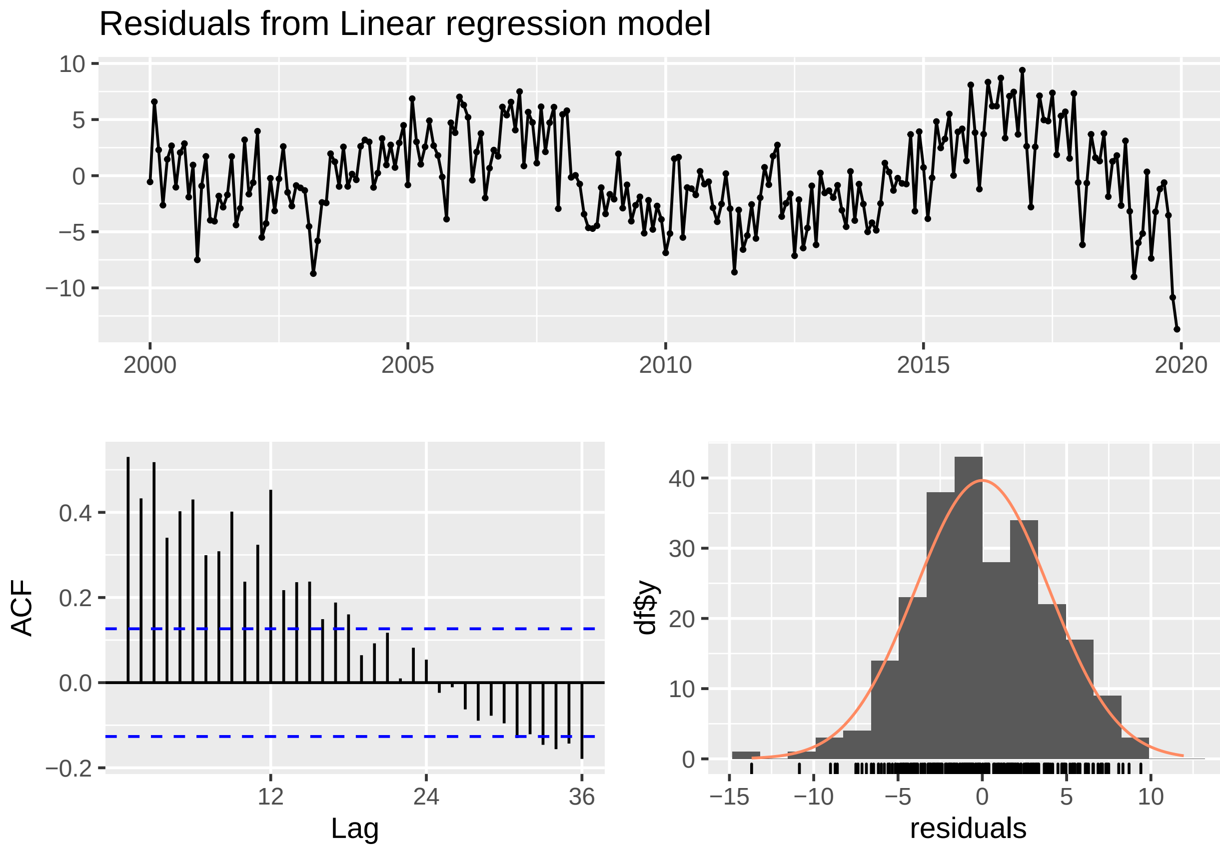
```r
# Vehicle Miles Traveled

# Try a cubic model with seasonal dummies
vmt_model <- tslm(vmt_ts ~ trend + I(trend^2) + I(trend^3) + season)
summary(vmt_model)
```

```
##
## Call:
## tslm(formula = vmt_ts ~ trend + I(trend^2) + I(trend^3) + season)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.6840  -2.7007  -0.3872   2.7370   9.4027
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.034e+02  1.368e+00 148.700   <2e-16 ***
## trend        6.229e-01  3.871e-02  16.091   <2e-16 ***
## I(trend^2)  -5.384e-03  3.728e-04 -14.442   <2e-16 ***
## I(trend^3)   1.569e-05  1.017e-06  15.429   <2e-16 ***
## season2     -1.194e+01  1.298e+00  -9.197   <2e-16 ***
## season3      2.499e+01  1.298e+00  19.257   <2e-16 ***
## season4      2.454e+01  1.298e+00  18.908   <2e-16 ***
## season5      3.468e+01  1.298e+00  26.716   <2e-16 ***
## season6      3.337e+01  1.298e+00  25.702   <2e-16 ***
## season7      3.869e+01  1.298e+00  29.796   <2e-16 ***
## season8      3.775e+01  1.299e+00  29.068   <2e-16 ***
## season9      1.648e+01  1.299e+00  12.685   <2e-16 ***
## season10     2.931e+01  1.299e+00  22.561   <2e-16 ***
## season11     1.226e+01  1.300e+00   9.437   <2e-16 ***
## season12     1.579e+01  1.300e+00  12.142   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.104 on 225 degrees of freedom
## Multiple R-squared:  0.9568, Adjusted R-squared:  0.9541
## F-statistic: 355.6 on 14 and 225 DF,  p-value: < 2.2e-16
```

```r
plot(vmt_ts, main = "Vehicle Miles Traveled", xlab = "Year", ylab = "Billions of Miles",
     col = "grey", lwd = 2)
lines(fitted(vmt_model), col="blue")
legend("topleft", legend = c("Data", "Model"), fill = c("grey","blue"), cex = 1, bty = 'n')
```




```
checkresiduals(vmt_model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 138.39, df = 24, p-value < 2.2e-16
```

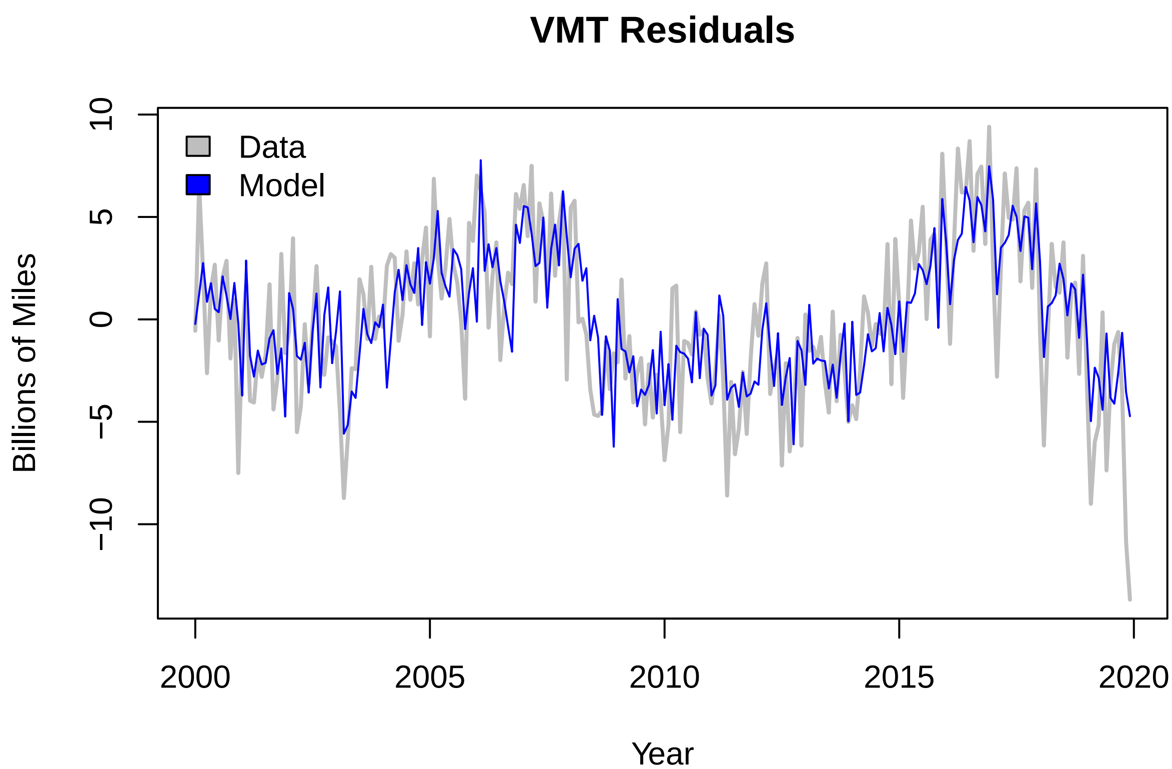
The model fits the data relatively well. We can see that the adjusted R-squared is 0.95, meaning the model explains 95% of the variation in the data. The cubic trend captures most of the trend component, although there are still outstanding residuals often at the peaks and troughs. We can see the ACF plot of the residuals is decaying, indicating that we have not captured all the variation in the data. As a result, we will fit a model to the residuals to capture additional seasonal and cyclical variation.

```
# R suggests ARIMA(3,0,4)(1,0,1)[12]
vmt_res_model <- auto.arima(vmt_model$residuals)
summary(vmt_res_model)
```

```
## Series: vmt_model$residuals
## ARIMA(3,0,4)(1,0,1)[12] with zero mean
##
## Coefficients:
##           ar1     ar2    ar3     ma1     ma2      ma3      ma4    sar1    sma1
##       -0.0850  0.0902  0.794  0.4800  0.2101  -0.4284  -0.1609  0.2579  0.2143
## s.e.   0.1152  0.0857  0.105  0.1305  0.1302   0.1275   0.0897  0.1735  0.1751
##
## sigma^2 = 7.288:  log likelihood = -576.58
```

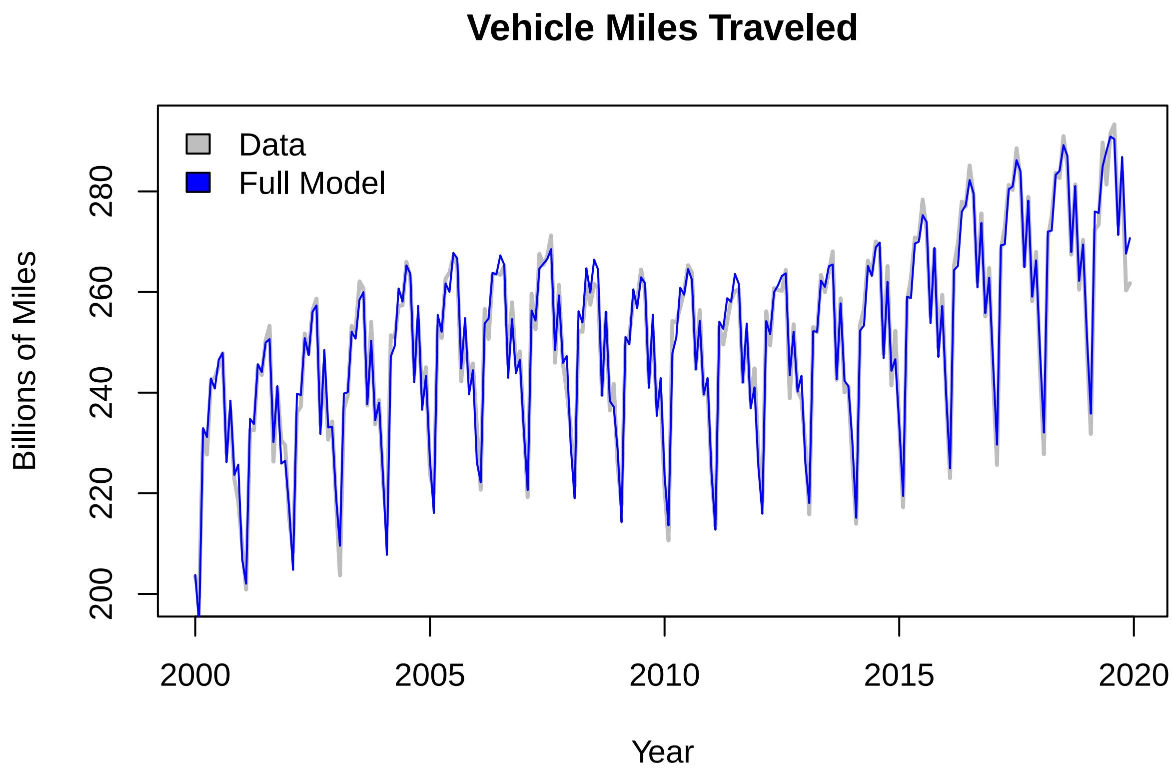
```
## AIC=1173.16   AICc=1174.12   BIC=1207.97
##
## Training set error measures:
##                       ME   RMSE      MAE       MPE     MAPE      MASE
## Training set -0.1014627 2.6485 2.104444 -2.707526 205.1118 0.6887207
##                        ACF1
## Training set -0.0002847343
```

```
# Plot data and our model against residuals
plot(vmt_model$residuals, main = "VMT Residuals", xlab = "Year",
     ylab = "Billions of Miles", col = "grey", lwd = 2)
lines(fitted(vmt_res_model), col="blue")
legend("topleft", legend = c("Data", "Model"), fill = c("grey","blue"), cex = 1, bty = 'n')
```

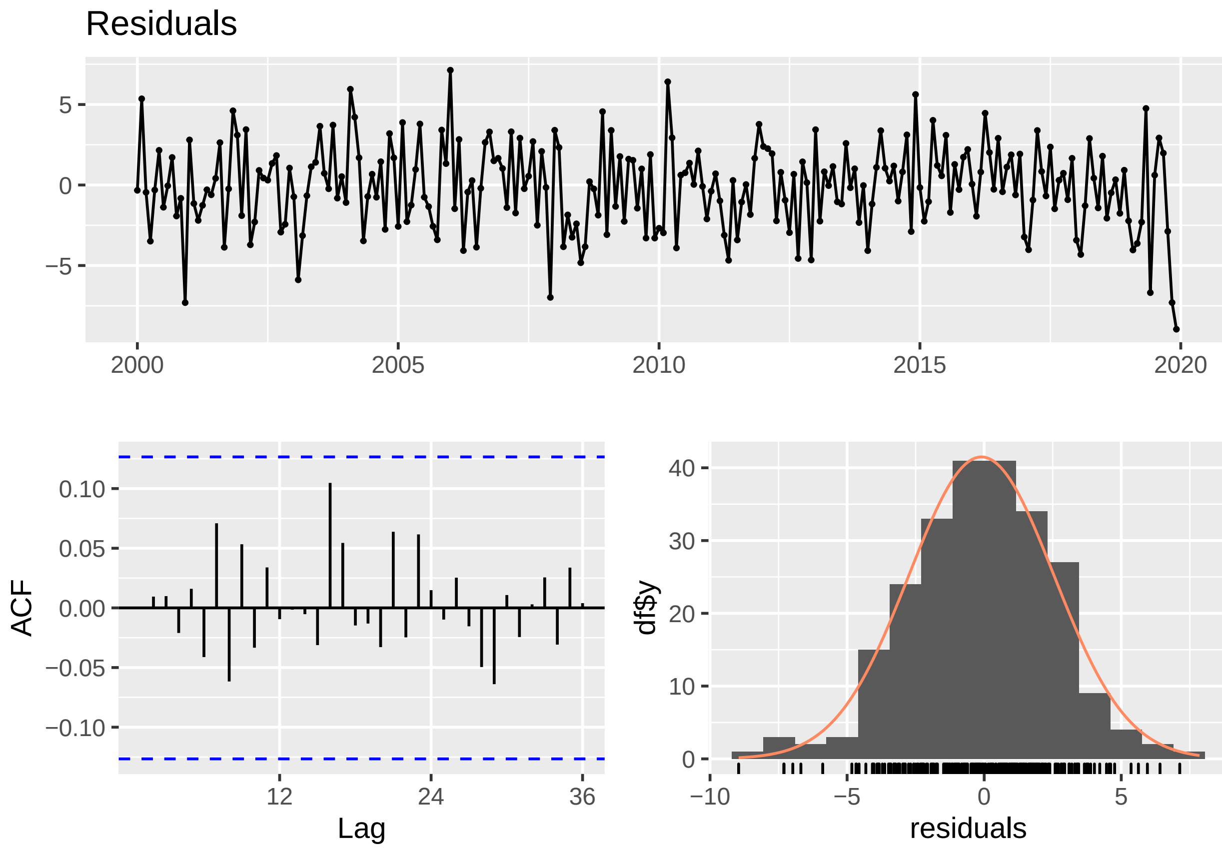


We plot an ARIMA model to capture some of the variation in the residuals. R suggests an S-ARMA(1,1) model, which means that our seasonal dummies did not perfectly capture the data. The ARMA(3,4) component is of a high order, but it does capture some of the remaining variation in the residuals. We can see that the SE of the coefficients is large compared to their estimates, so some of them may not be necessary, but we choose to include them here. Neither the ARMA nor S-ARMA component has an I term, which means that our original model captured the trend adequately.

```
# Plot data against our full model
plot(vmt_ts, main = "Vehicle Miles Traveled", xlab = "Year",
     ylab = "Billions of Miles", col = "grey", lwd = 2)
lines(fitted(vmt_model) + fitted(vmt_res_model), col="blue")
legend("topleft", legend = c("Data", "Full Model"), fill = c("grey","blue"), cex = 1, bty = 'n')
```




```
vmt_fitted = vmt_model$fitted.values + vmt_res_model$fitted
vmt_residuals = vmt_model$residuals - vmt_res_model$fitted
checkresiduals(vmt_residuals)
```



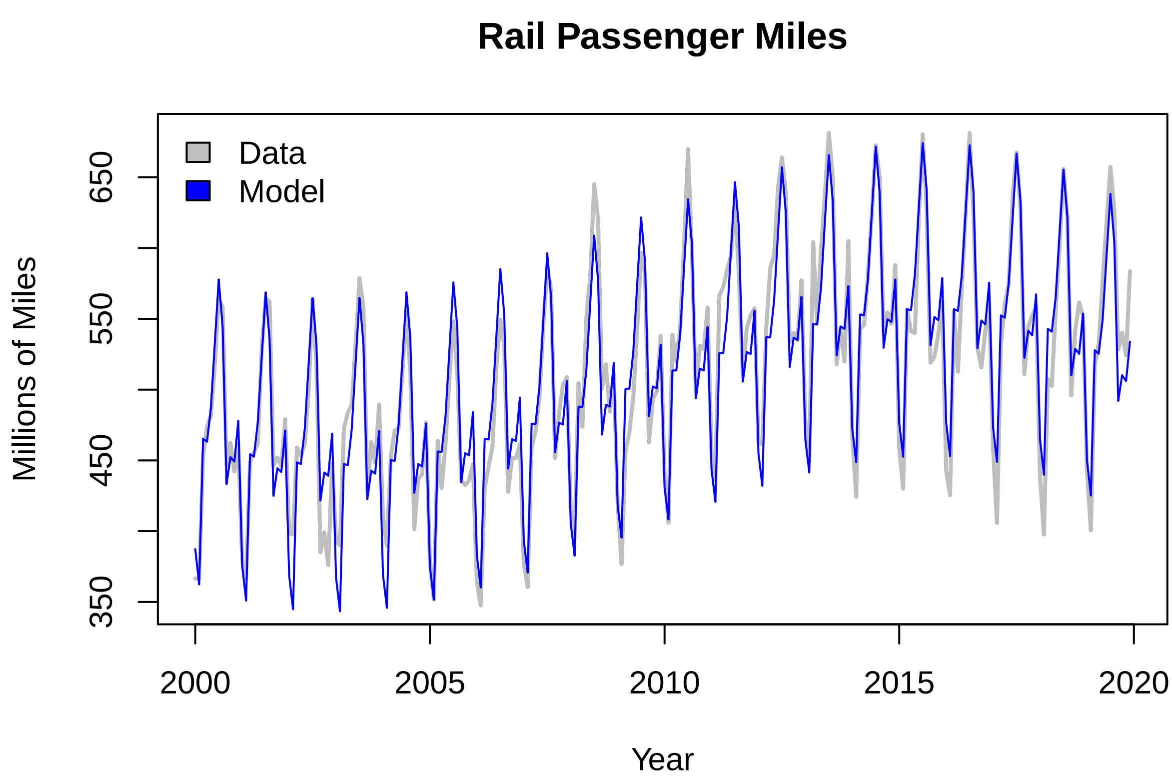
We can see that the full model (cubic trend, seasonal dummies, and ARIMA model for residuals) captures the data very well, better than the cubic trend + seasonal dummies alone. The residuals of our final model demonstrate no serial correlation, and we can see from the histogram of the residuals that they are a very close match for the expected distribution.

```
# Rail Passenger Miles

# Try a cubic model with seasonal dummies
rpm_model <- tslm(rpm_ts ~ trend + I(trend^2) + I(trend^3) + season)
summary(rpm_model)
```

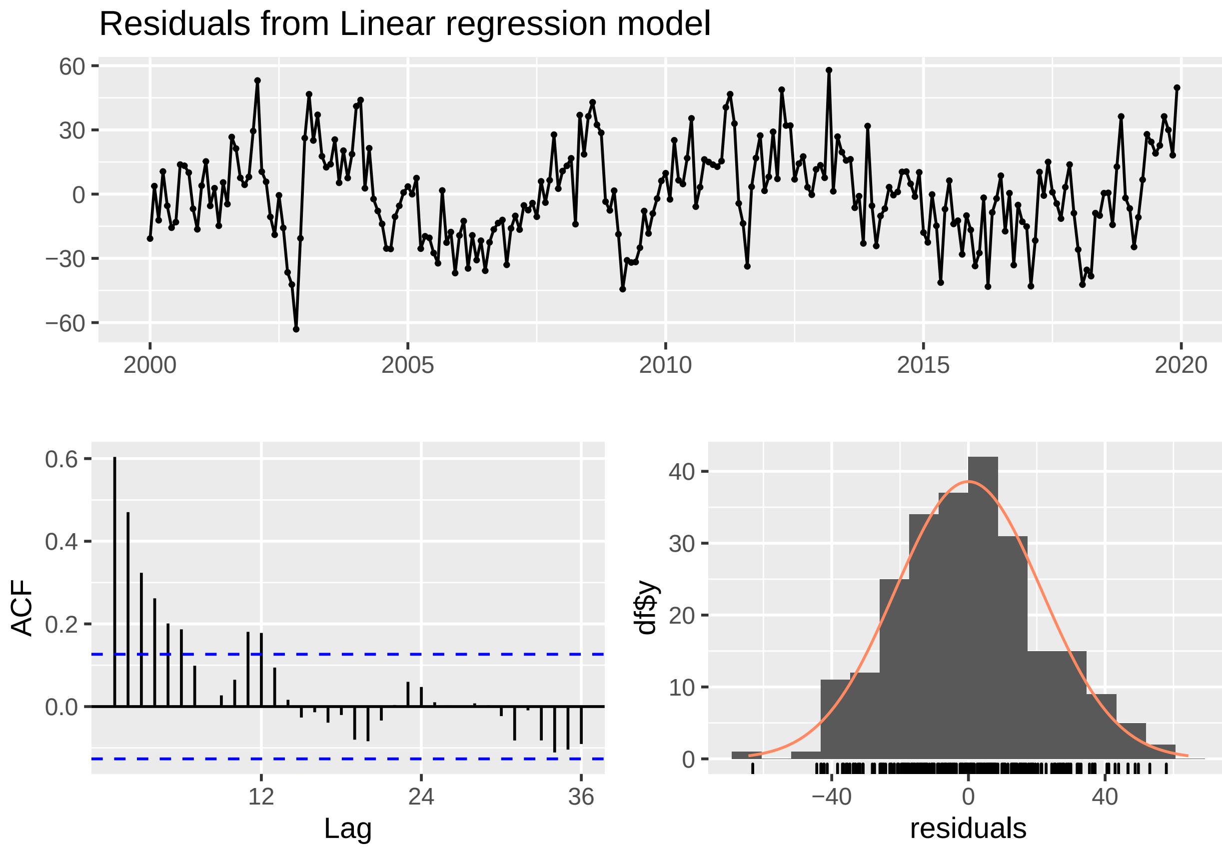
```
##
## Call:
## tslm(formula = rpm_ts ~ trend + I(trend^2) + I(trend^3) + season)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.111 -14.438  -0.104  13.762  57.913
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.886e+02  7.371e+00  52.722  < 2e-16 ***
## trend       -1.268e+00  2.086e-01  -6.080 5.10e-09 ***
## I(trend^2)   2.085e-02  2.009e-03  10.375  < 2e-16 ***
## I(trend^3)  -6.169e-05  5.481e-06 -11.255  < 2e-16 ***
## season2     -2.378e+01  6.994e+00  -3.399 0.000799 ***
```

```
## season3      8.031e+01  6.994e+00  11.482  < 2e-16 ***
## season4      7.930e+01  6.995e+00  11.337  < 2e-16 ***
## season5      1.044e+02  6.996e+00  14.921  < 2e-16 ***
## season6      1.510e+02  6.997e+00  21.588  < 2e-16 ***
## season7      1.971e+02  6.998e+00  28.162  < 2e-16 ***
## season8      1.645e+02  6.999e+00  23.502  < 2e-16 ***
## season9      5.438e+01  7.001e+00   7.769 2.79e-13 ***
## season10     7.433e+01  7.002e+00  10.616  < 2e-16 ***
## season11     7.200e+01  7.004e+00  10.280  < 2e-16 ***
## season12     1.018e+02  7.006e+00  14.529  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.12 on 225 degrees of freedom
## Multiple R-squared:  0.9215, Adjusted R-squared:  0.9166
## F-statistic: 188.6 on 14 and 225 DF,  p-value: < 2.2e-16
```

```
plot(rpm_ts, main = "Rail Passenger Miles", xlab = "Year", ylab = "Millions of Miles",
     col = "grey", lwd = 2)
lines(fitted(rpm_model), col="blue")
legend("topleft", legend = c("Data", "Model"), fill = c("grey","blue"), cex = 1, bty = 'n')
```




```
checkresiduals(rpm_model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 110.09, df = 24, p-value = 5.449e-13
```

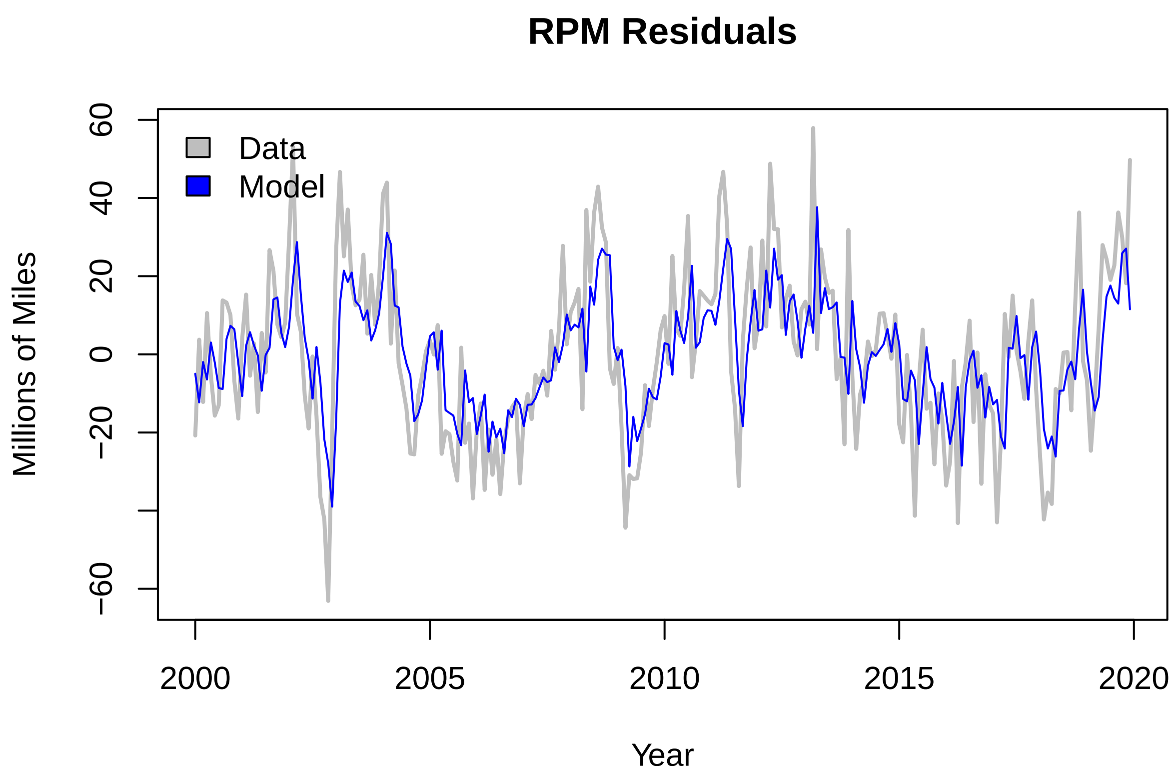
The model fits the data relatively well. We can see that the adjusted R-squared is 0.92, meaning the model explains about 92% of the variation in the data (slightly less than the same model for VMT). The cubic trend captures most of the trend component, although there are still significant outstanding residuals often at the peaks and troughs. We can see the ACF plot of the residuals is decaying and cyclic, indicating that we have not captured all the variation in the data. As a result, we will fit a model to the residuals to capture additional seasonal and cyclical variation.

```
# R suggests ARIMA(1,0,1)(1,0,2)[12]
rpm_res_model <- auto.arima(rpm_model$residuals)
summary(rpm_res_model)
```

```
## Series: rpm_model$residuals
## ARIMA(1,0,1)(1,0,2)[12] with zero mean
##
## Coefficients:
##          ar1      ma1    sar1     sma1     sma2
##       0.7893  -0.2933  0.3014  -0.1374  -0.0021
## s.e.  0.0610   0.0935  0.6310   0.6265   0.1427
##
```

```
## sigma^2 = 277.7:  log likelihood = -1013.64
## AIC=2039.28   AICc=2039.65   BIC=2060.17
##
## Training set error measures:
##                     ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 0.2507227 16.48867 13.05443 86.32423 257.9193 0.5988269
##                     ACF1
## Training set -0.01161357
```

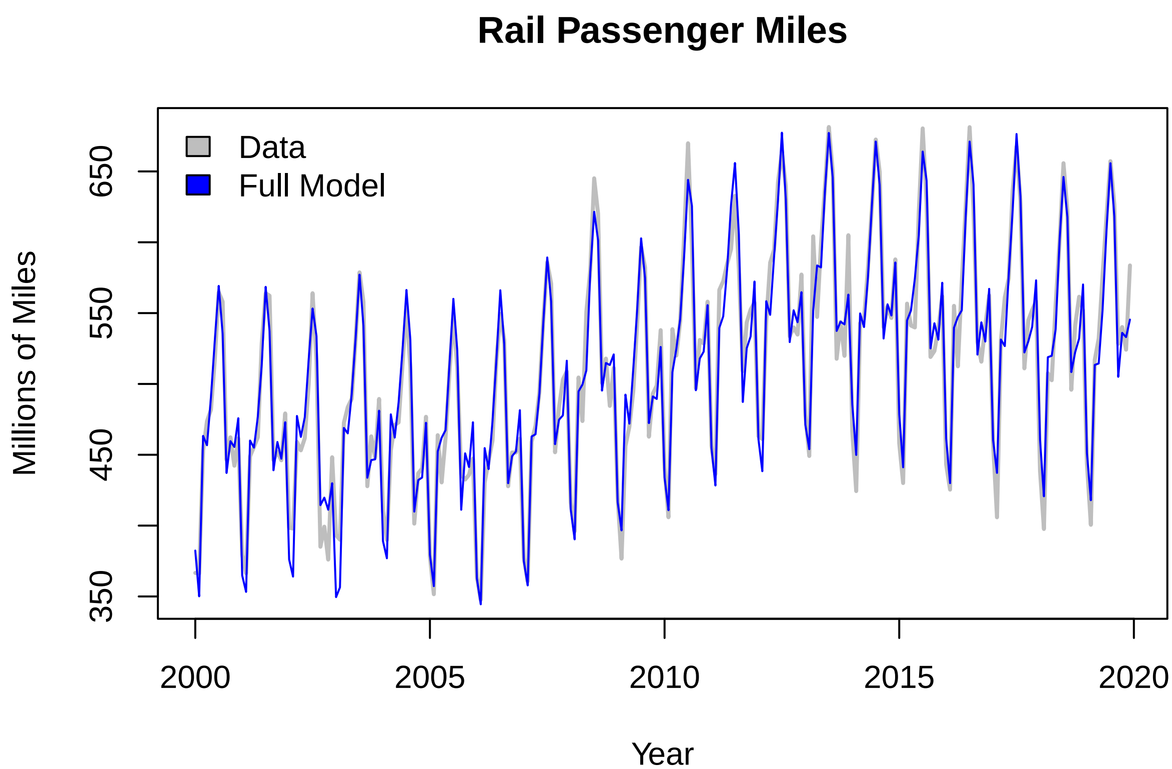
```
# Plot data and our model against residuals
plot(rpm_model$residuals, main = "RPM Residuals", xlab = "Year", ylab = "Millions of Miles",
     col = "grey", lwd = 2)
lines(fitted(rpm_res_model), col="blue")
legend("topleft", legend = c("Data", "Model"), fill = c("grey","blue"), cex = 1, bty = 'n')
```



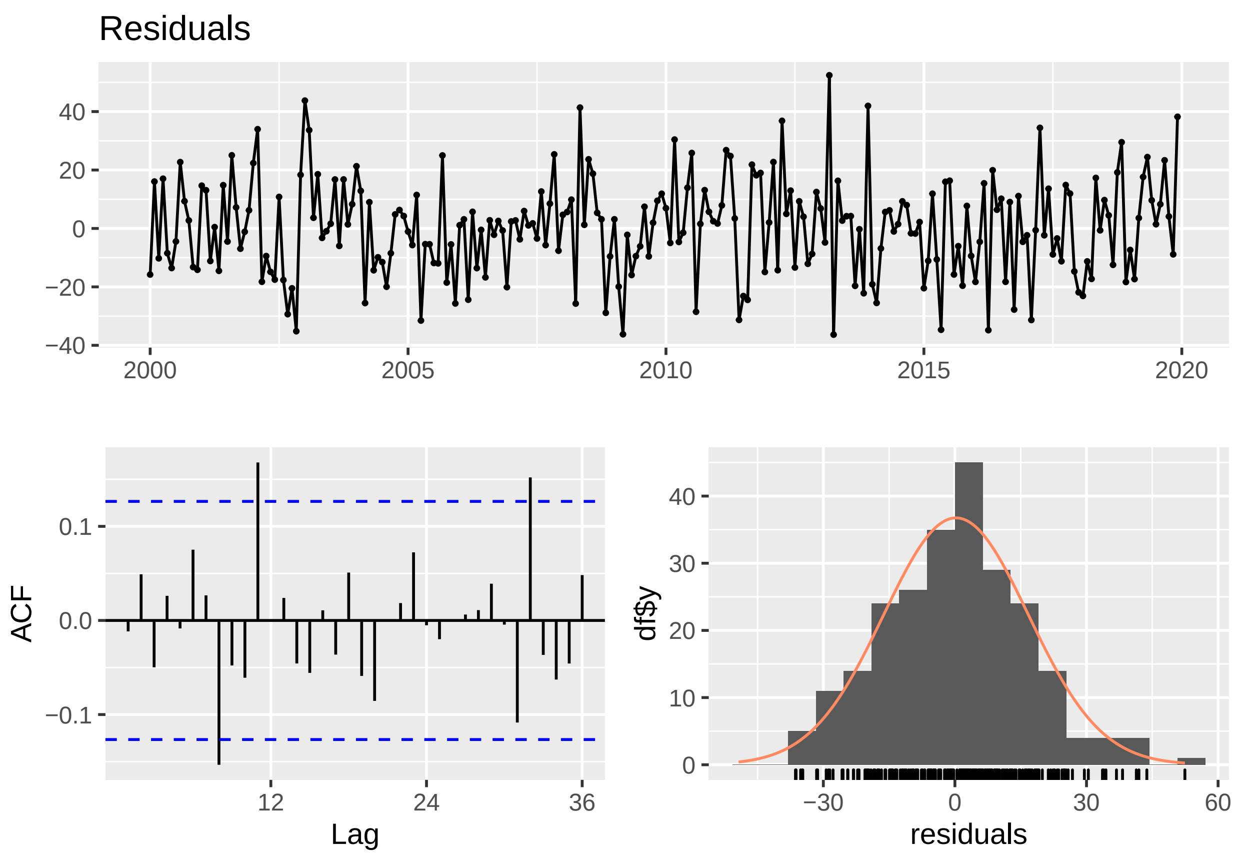
We plot an ARIMA model to capture some of the variation in the residuals. R suggests an S-ARMA(1,2) model, which means that our seasonal dummies did not perfectly capture the seasonal variation in the data. The ARMA(1,1) component is of a high order, but it does capture some of the remaining variation in the residuals. We can see that the SE of the coefficients is large compared to their estimates, so some of them may not be necessary, but we choose to include them here. Neither the ARMA nor S-ARMA component has an I term, which means that our original model captured the trend adequately.

```
# Plot data against our full model
plot(rpm_ts, main = "Rail Passenger Miles", xlab = "Year", ylab = "Millions of Miles",
     col = "grey", lwd = 2)
```

```
lines(fitted(rpm_model) + fitted(rpm_res_model), col="blue")
legend("topleft", legend = c("Data", "Full Model"), fill = c("grey","blue"), cex = 1, bty = 'n')
```




```
rpm_fitted = rpm_model$fitted.values + rpm_res_model$fitted
rpm_residuals = rpm_model$residuals - rpm_res_model$fitted
checkresiduals(rpm_residuals)
```

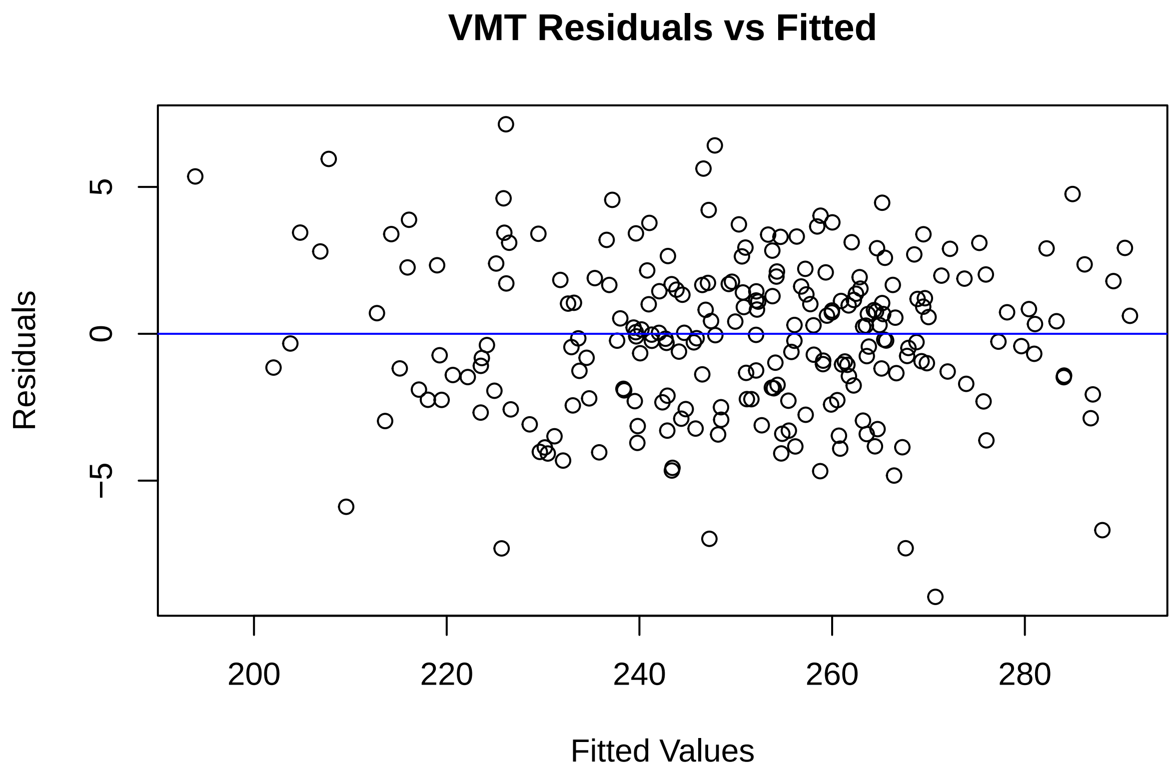


We can see that the full model (cubic trend, seasonal dummies, and ARIMA model for residuals) captures the data very well, better than the cubic trend + seasonal dummies alone. The residuals of our final model demonstrate some serial correlation at unusual lags, though these are unlikely to be system, and we can see from the histogram of the residuals that they are a very close match for the expected distribution.

(e)

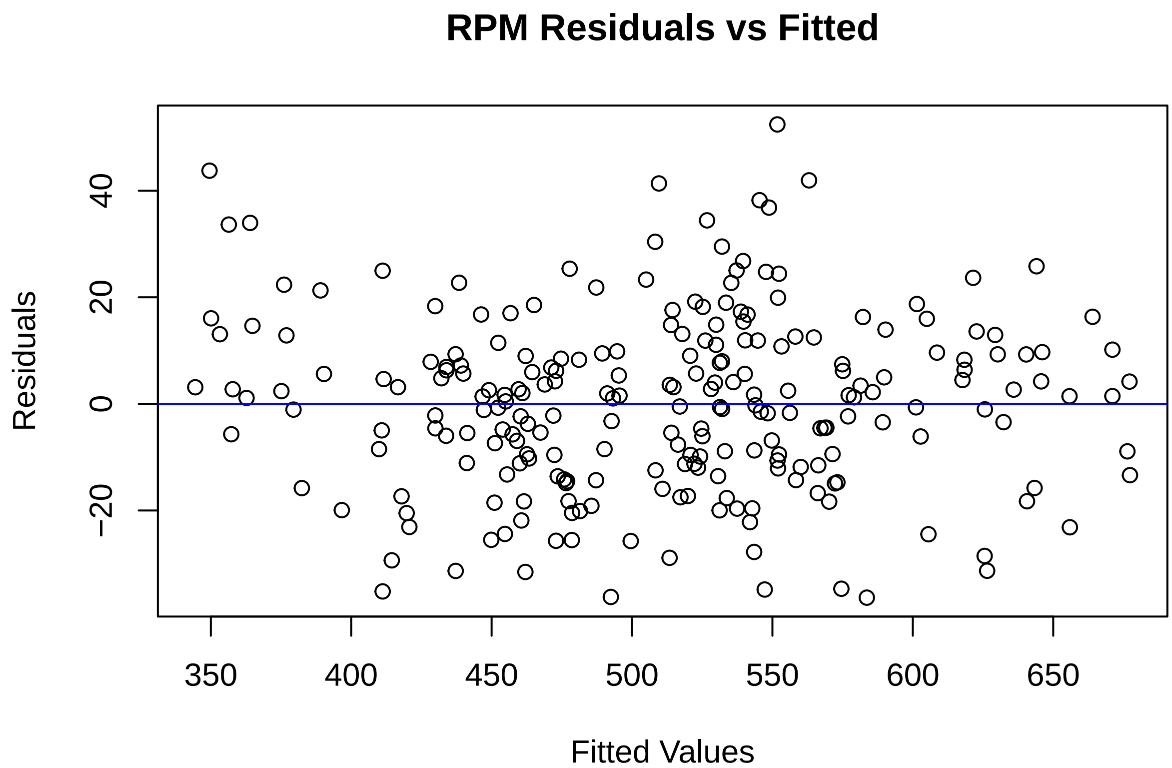
Plot the respective residuals vs. fitted values and discuss your observations.

```
# Vehicle Miles Traveled
plot(x = vmt_fitted, y = vmt_residuals, xlab = "Fitted Values", ylab = "Residuals",
     main = "VMT Residuals vs Fitted")
abline(h = 0, col = "blue")
```



- The residuals are randomly distributed about the y=0 line which suggests that the residuals are normally distributed.
- The residuals form a "horizontal band" around the y=0 line. This suggests that the variance of the residuals is constant.
- Magnitude or spread of residuals is somewhat moderate.
- There are no potential outliers present in the residuals, suggesting no extraordinary events.

```
# Rail Passenger Miles
plot(x = rpm_fitted, y = rpm_residuals, xlab = "Fitted Values", ylab = "Residuals",
     main = "RPM Residuals vs Fitted")
abline(h = 0, col = "blue")
```

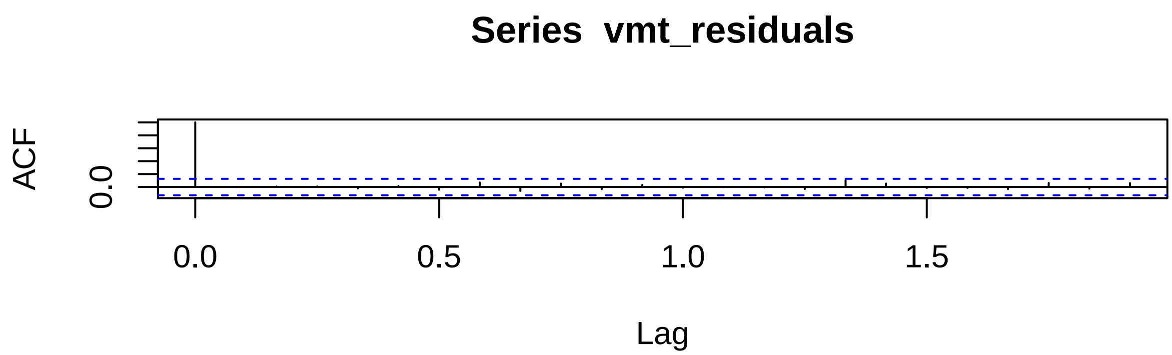


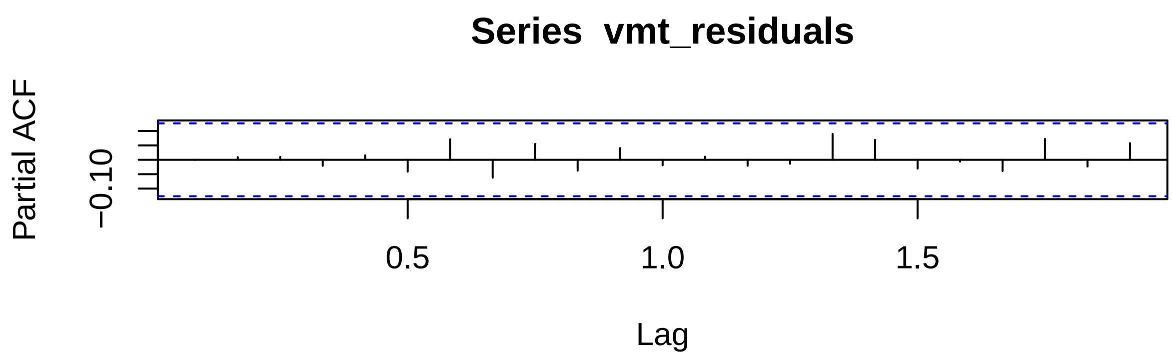
- The residuals are randomly distributed about the y=0 line which suggests that the residuals are normally distributed.
- The residuals form a "horizontal band" around the y=0 line. This suggests that the variance of the residuals is fairly constant.
- Magnitude or spread of residuals is moderate compared to the scale of the data.
- There are no potential outliers present in the residuals, suggesting no extraordinary events.

(f)

Plot the ACF and PACF of the respective residuals and interpret the plots.

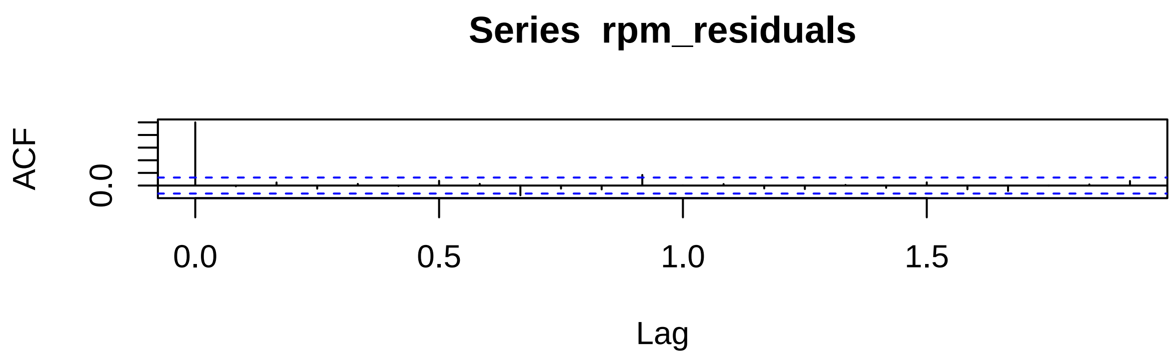
```
# Vehicle Miles Traveled
par(mfrow = c(2,1))
acf(vmt_residuals)
pacf(vmt_residuals)
```

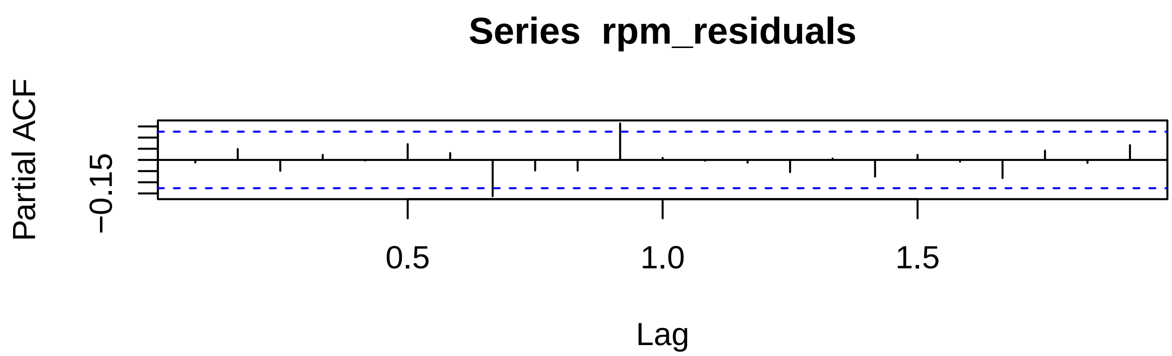




- The ACF and the PACF plot of the residuals contain no significant spikes past k = 0.
- Both these observations indicate that the residuals are a white noise process.

```r
# Rail Passenger Miles
par(mfrow = c(2,1))
acf(rpm_residuals)
pacf(rpm_residuals)
```



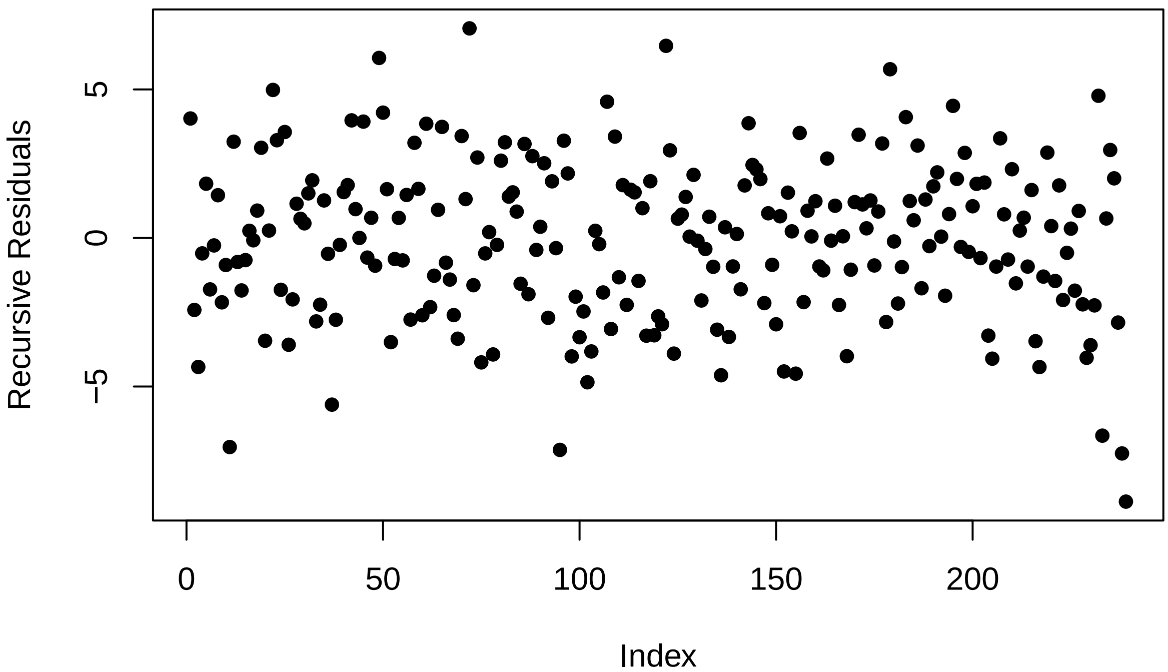


- The ACF plot of the residuals contains some spikes, but they are not at regular seasonal intervals.
- The PACF plot of the residuals also shows some significant spikes at the same irregular lags.
- These observations *may* indicate that the residuals are not a white noise process, but it seems unlikely that there would be a correlation at 8 or 11 months out, so we should not assume the residuals are not a white noise process.

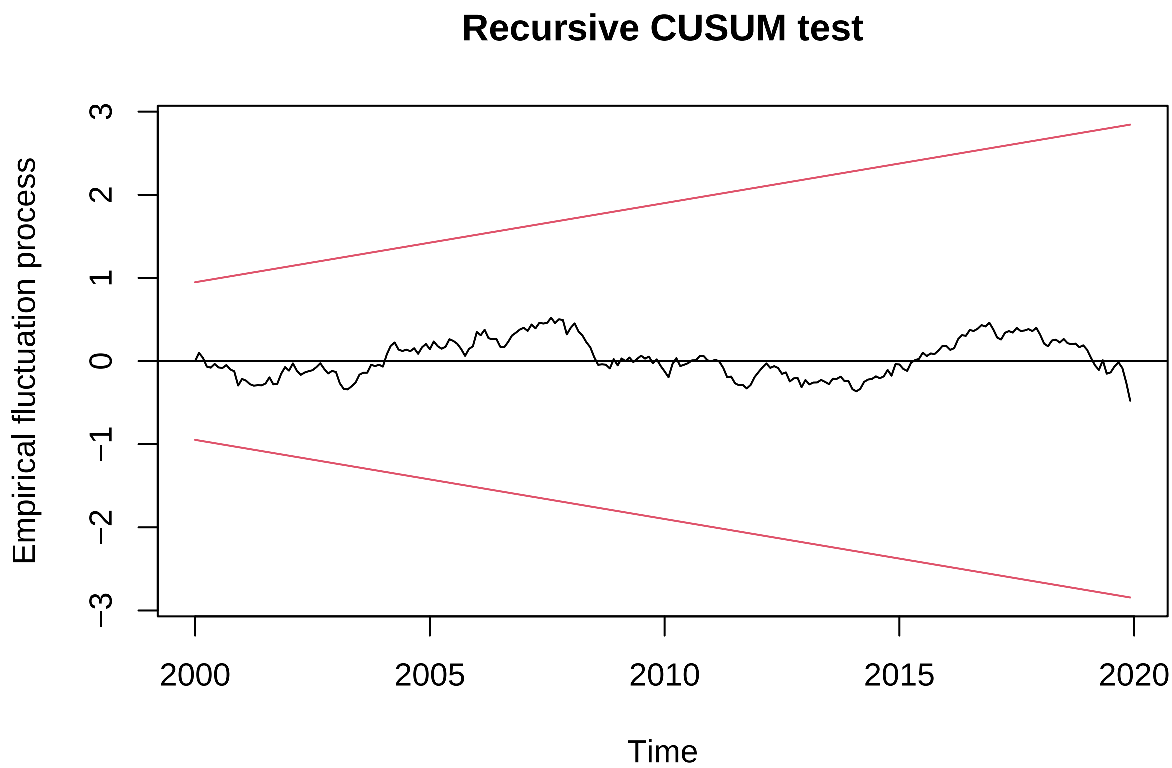
(g)

Plot the respective CUSUM and interpret the plot.

```
# Vehicle Miles Traveled (1976-01-01 to 2020-01-01)
y = recresid(vmt_residuals ~ 1)
plot(y, pch = 16, ylab = "Recursive Residuals")
```

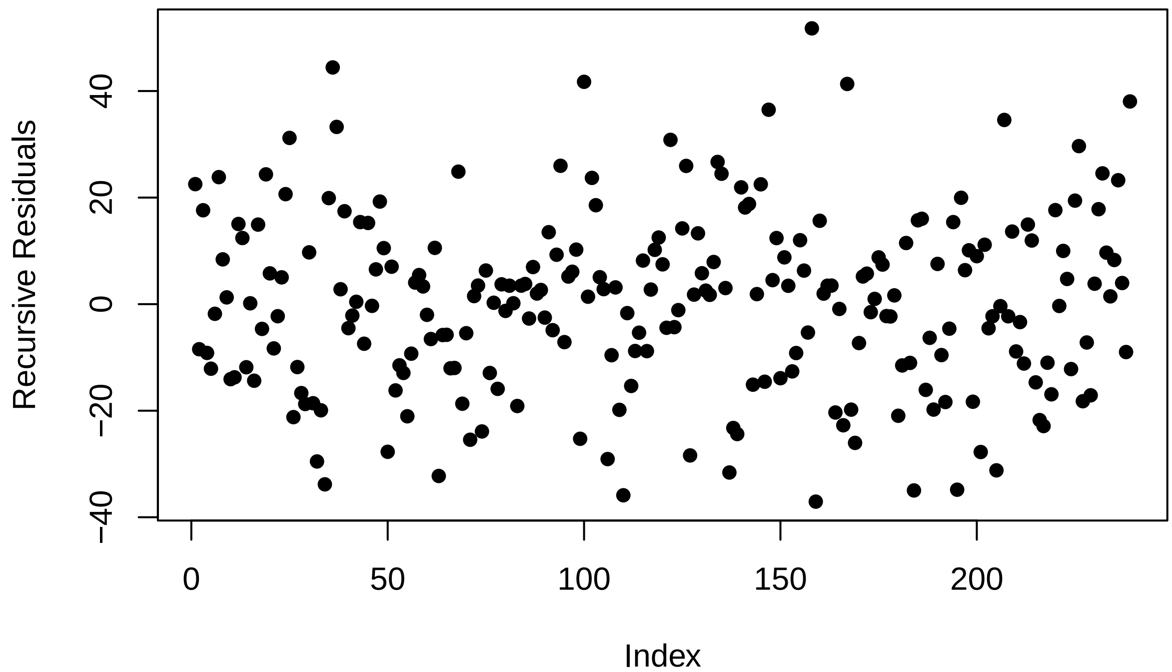



```
plot(efp(vmt_residuals~1, type = "Rec-CUSUM"))
```

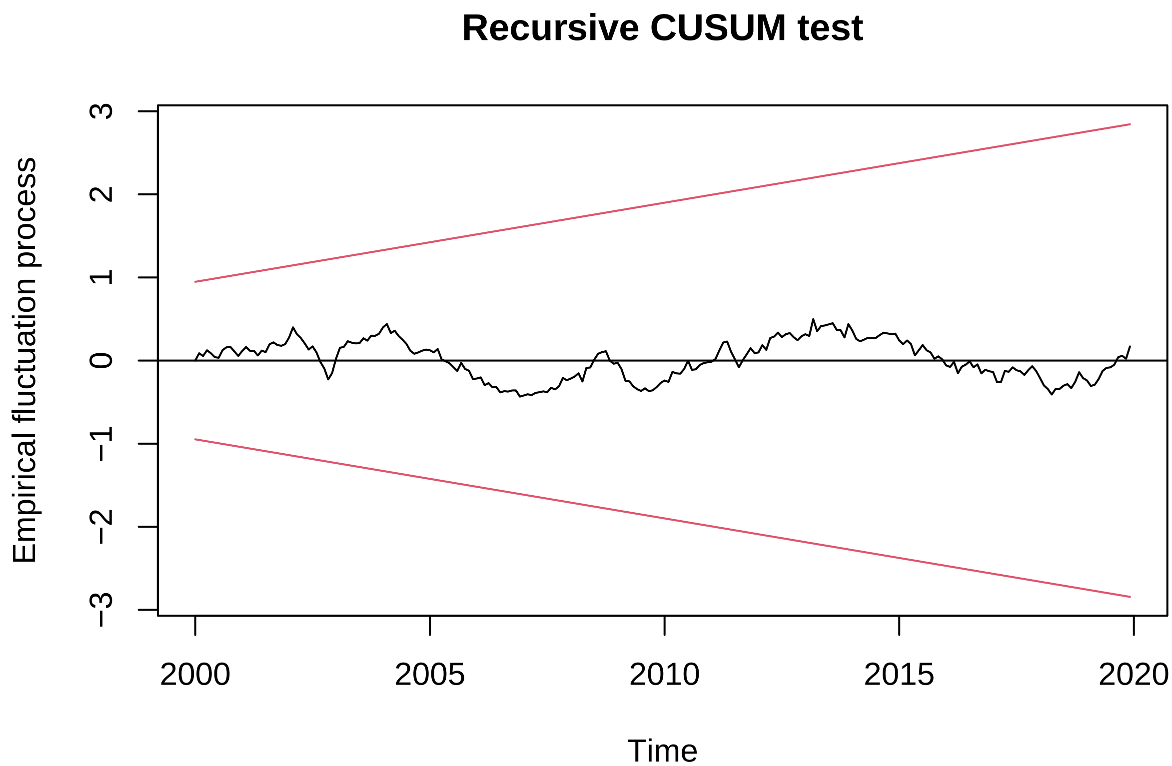


The CUSUM plot of the recursive residuals show that our model parameters are stable since the black line does not cross the red line throughout the series. The cumulative residuals stay relatively close to zero, even crossing a few times, indicating a good model fit over the entire period of the data.

```
# Rail Passenger Miles
y = recresid(rpm_residuals ~ 1)
plot(y, pch = 16, ylab = "Recursive Residuals")
```




```
plot(efp(rpm_residuals~1, type = "Rec-CUSUM"))
```



The CUSUM plot of the recursive residuals show that our model parameters are stable since the black line does not cross the red line throughout the series.The cumulative residuals stay relatively close to zero, even crossing a couple of times, indicating a good model fit over the entire period of the data.

(h)

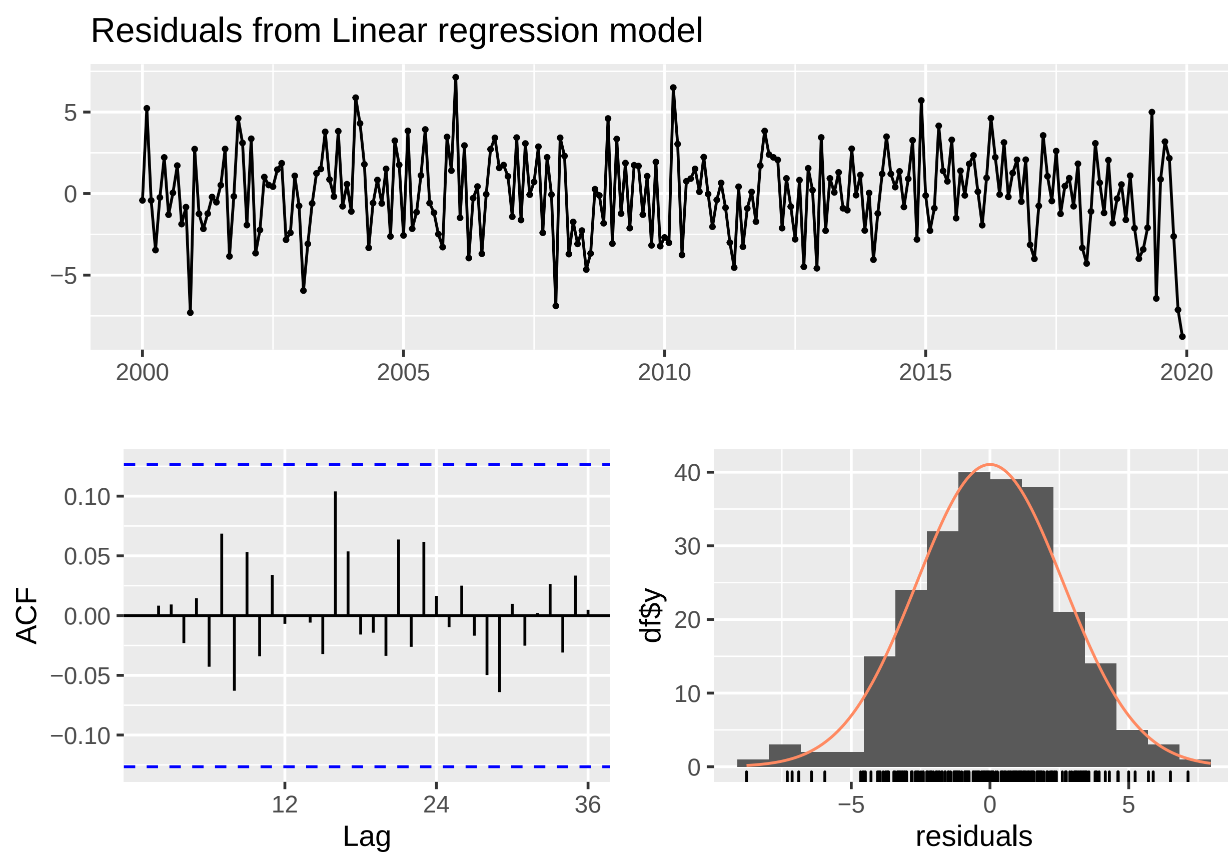
For your model, discuss the associated diagnostic statistics.

```r
# Vehicle Miles Travelled
summary(tslm(vmt_ts ~ vmt_fitted))
```

```
##
## Call:
## tslm(formula = vmt_ts ~ vmt_fitted)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.7742 -1.8339  0.0401  1.7691  7.1376
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.904412   2.267475   0.399     0.69
## vmt_fitted  0.995985   0.009025 110.358   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 2.657 on 238 degrees of freedom
## Multiple R-squared:  0.9808, Adjusted R-squared:  0.9808
## F-statistic: 1.218e+04 on 1 and 238 DF,  p-value: < 2.2e-16
```

```
checkresiduals(tslm(vmt_ts ~ vmt_fitted))
```

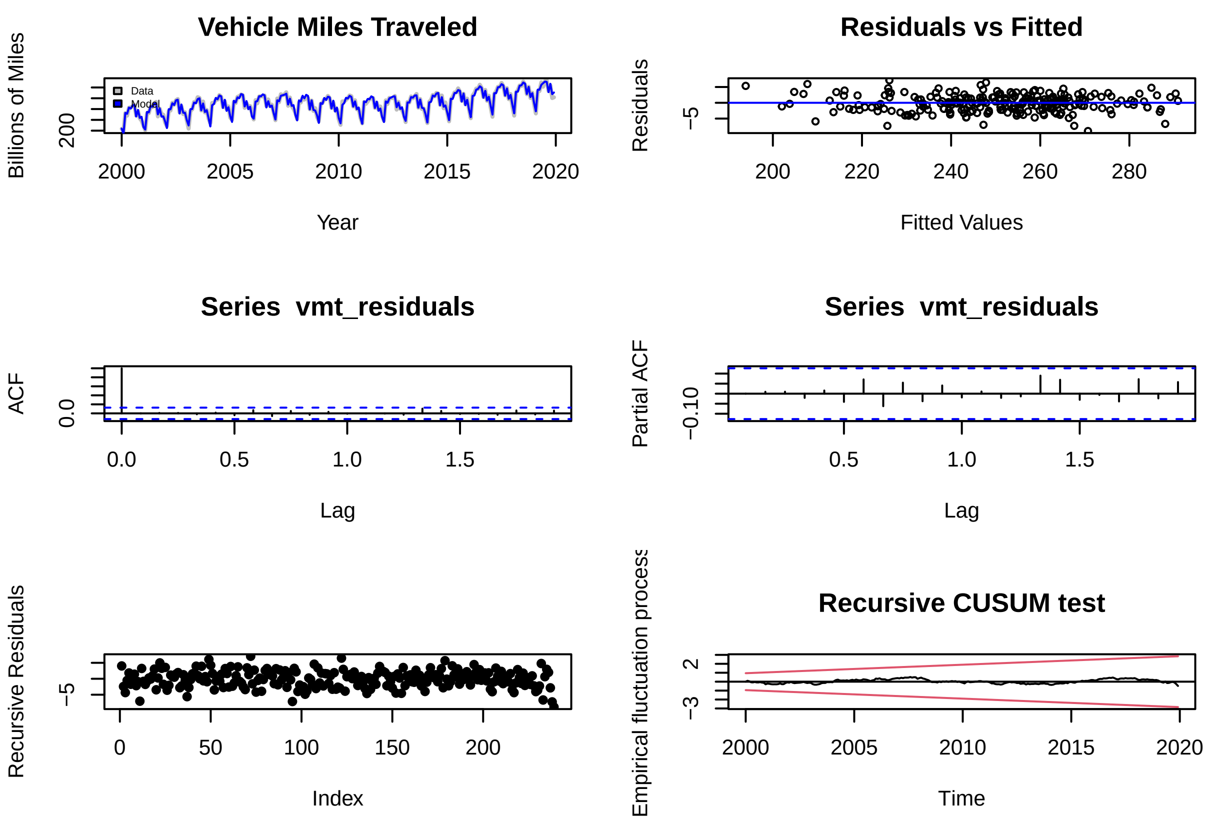



```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 12.641, df = 24, p-value = 0.9717
```

```
vmt_accuracy = cbind(ME(vmt_residuals), MSE(vmt_residuals), RMSE(vmt_residuals),
                     MAE(vmt_residuals), MPE(vmt_residuals, vmt_fitted),
                     MAPE(vmt_residuals, vmt_fitted), MASE(vmt_residuals, vmt_ts, .period=12),
                     RMSSE(vmt_residuals, vmt_ts, .period=12))
colnames(vmt_accuracy) = c("ME", "MSE", "RMSE", "MAE", "MPE", "MAPE", "MASE", "RMSSE")
vmt_accuracy
```

```
##               ME     MSE   RMSE      MAE         MPE      MAPE     MASE
## [1,] -0.1014627 7.01455 2.6485 2.104444 -0.03751458 0.8523134 0.548996
##          RMSSE
## [1,] 0.5748218
```

```
par(mfrow = c(3,2))

# Model Plot
plot(vmt_ts, main = "Vehicle Miles Traveled", xlab = "Year", ylab = "Billions of Miles",
     col = "grey", lwd = 2)
lines(fitted(vmt_model) + fitted(vmt_res_model), col="blue")
legend("topleft", legend = c("Data", "Model"), fill = c("grey","blue"), cex = 0.5, bty = 'n')

# Residuals vs fitted
plot(x = vmt_fitted, y = vmt_residuals, xlab = "Fitted Values", ylab = "Residuals",
     main = "Residuals vs Fitted")
abline(h = 0, col = "blue")

# ACF and PACF
acf(vmt_residuals)
pacf(vmt_residuals)

# Recursive CUSUM
y = recresid(vmt_residuals ~ 1)
plot(y, pch = 16, ylab = "Recursive Residuals")
plot(efp(vmt_residuals~1, type = "Rec-CUSUM"))
```



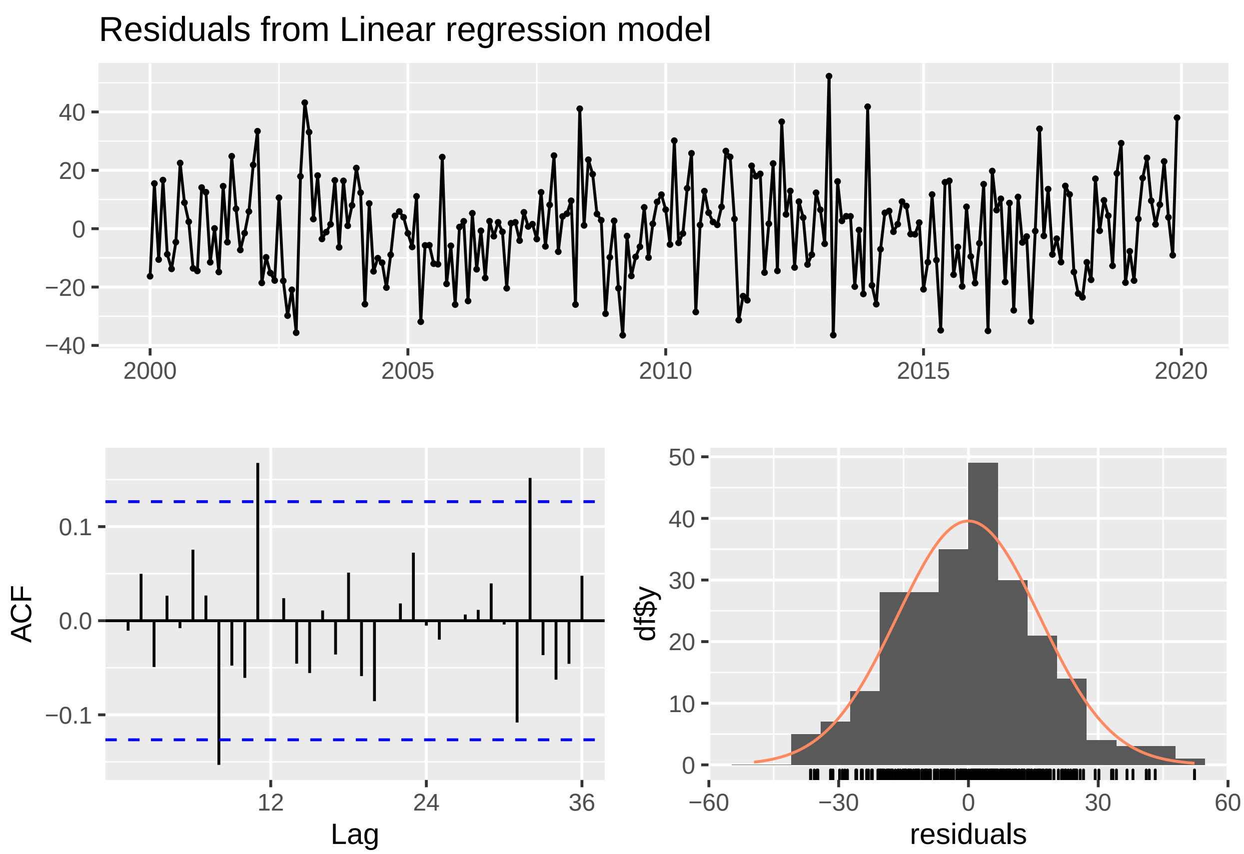
We can see that the training set errors for the full model fit are relatively small, indicating a good fit. The adjusted R-squared for the entire model is 0.98, meaning that our model captures 98% of the variation in the data. We can see from the statistics regarding residuals that there is a low probability of rejecting the hypothesis that the residuals are not white noise.

We conclude that a cubic model + ARIMA(3,0,4)(1,0,1)[12] is a very good fit for Vehicle Miles Traveled.

```
# Rail Passenger Miles
summary(tslm(rpm_ts ~ rpm_fitted))
```

```
##
## Call:
## tslm(formula = rpm_ts ~ rpm_fitted)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.509 -11.504   1.196   9.852  52.261
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.27235    7.38382   0.172    0.863
## rpm_fitted   0.99800    0.01429  69.834   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.56 on 238 degrees of freedom
## Multiple R-squared:  0.9535, Adjusted R-squared:  0.9533
## F-statistic:  4877 on 1 and 238 DF,  p-value: < 2.2e-16
```

```
checkresiduals(tslm(rpm_ts ~ rpm_fitted))
```

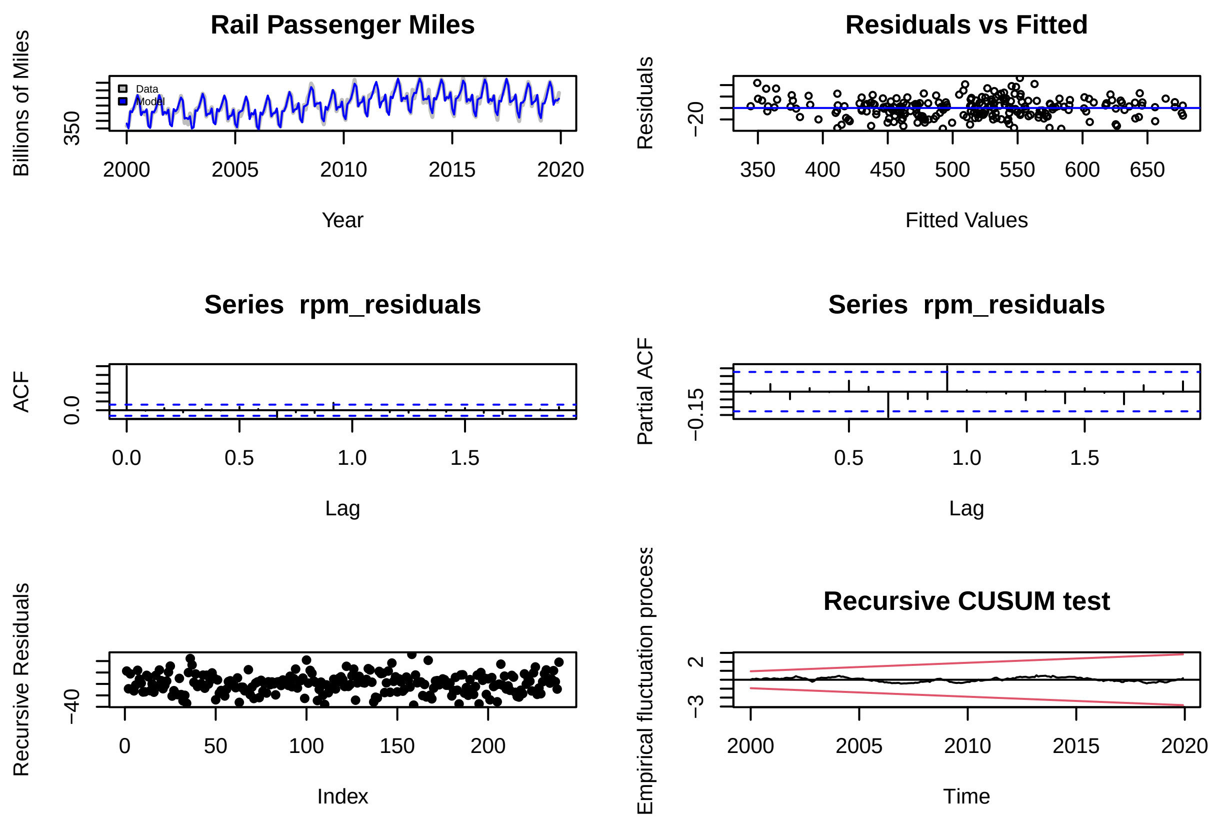


```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 22.207, df = 24, p-value = 0.5669
```

```
rpm_accuracy = cbind(ME(rpm_residuals), MSE(rpm_residuals), RMSE(rpm_residuals),
                     MAE(rpm_residuals), MPE(rpm_residuals, rpm_fitted),
                     MAPE(rpm_residuals, rpm_fitted), MASE(rpm_residuals, rpm_ts, .period=12),
                     RMSSE(rpm_residuals, rpm_ts, .period=12))
colnames(rpm_accuracy) = c("ME", "MSE", "RMSE", "MAE", "MPE", "MAPE", "MASE", "RMSSE")
rpm_accuracy
```

```
##             ME      MSE     RMSE      MAE        MPE     MAPE      MASE
## [1,] 0.2507227 271.8764 16.48867 13.05443 0.06858536 2.619336 0.5915457
##          RMSSE
## [1,] 0.5865631
```

```
par(mfrow = c(3,2))

# Model Plot
plot(rpm_ts, main = "Rail Passenger Miles", xlab = "Year", ylab = "Billions of Miles",
     col = "grey", lwd = 2)
lines(fitted(rpm_model) + fitted(rpm_res_model), col="blue")
legend("topleft", legend = c("Data", "Model"), fill = c("grey","blue"), cex = 0.5, bty = 'n')

# Residuals vs fitted
plot(x = rpm_fitted, y = rpm_residuals, xlab = "Fitted Values", ylab = "Residuals",
     main = "Residuals vs Fitted")
abline(h = 0, col = "blue")

# ACF and PACF
acf(rpm_residuals)
pacf(rpm_residuals)

# Recursive CUSUM
y = recresid(rpm_residuals ~ 1)
plot(y, pch = 16, ylab = "Recursive Residuals")
plot(efp(rpm_residuals~1, type = "Rec-CUSUM"))
```



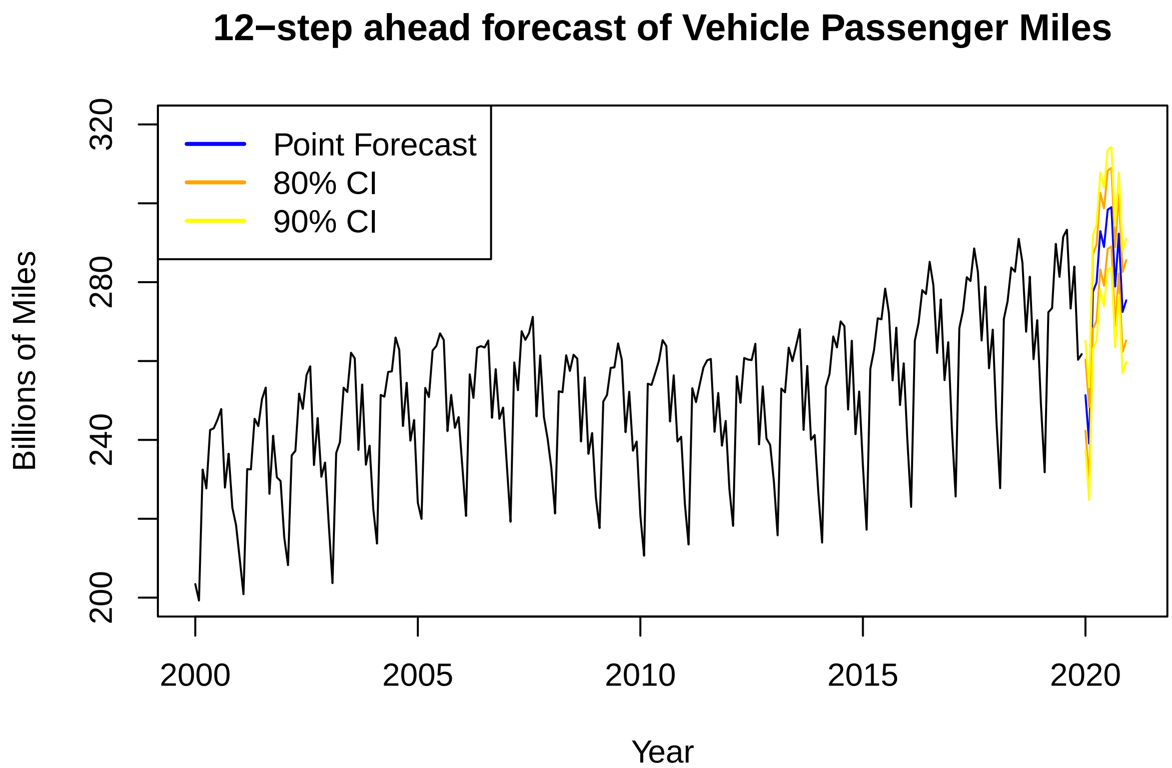
We can see that the training set errors for the full model fit are relatively small (though larger than for VMT), indicating a good fit. The adjusted R-squared for the entire model is 0.95, meaning that our model captures 95% of the variation in the data. We can see from the statistics regarding residuals that there is a low probability of rejecting the hypothesis that the residuals are not white noise.

We conclude that a cubic model + ARIMA(1,0,1)(1,0,2)[12] is a very good fit for Rail Passenger Miles.

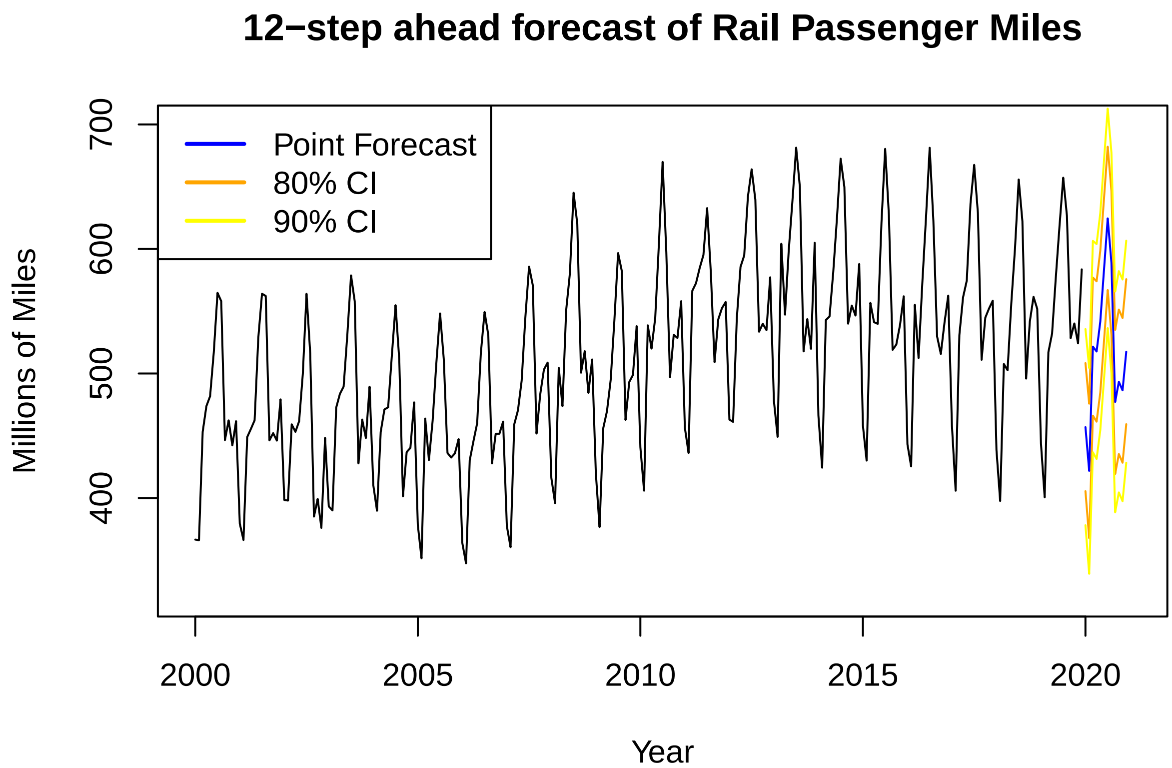
(i)

Use your model to forecast 12-steps ahead. Your forecast should include the respective error bands.

```
# Vehicle Miles Traveled
vmt_forecast = (as.data.frame(forecast(vmt_model, h=12)) +
                  as.data.frame(forecast(vmt_res_model, h=12)))
plot(vmt_ts, xlim=c(2000, 2021), ylim=c(200, 320), xlab = "Year", ylab = "Billions of Miles",
     main="12-step ahead forecast of Vehicle Passenger Miles")
# Point forecast
lines(x=as.yearmon(rownames(vmt_forecast)), vmt_forecast$`Point Forecast`, col="blue")
# 80% CI
lines(x=as.yearmon(rownames(vmt_forecast)), vmt_forecast$`Lo 80`, col="orange")
lines(x=as.yearmon(rownames(vmt_forecast)), vmt_forecast$`Hi 80`, col="orange")
# 90% CI
lines(x=as.yearmon(rownames(vmt_forecast)), vmt_forecast$`Lo 95`, col="yellow")
lines(x=as.yearmon(rownames(vmt_forecast)), vmt_forecast$`Hi 95`, col="yellow")
legend("topleft", legend=c("Point Forecast", "80% CI", "90% CI"),
       col=c("blue", "orange", "yellow"), lwd=2)
```




```
# Rail Passenger Miles
rpm_forecast = (as.data.frame(forecast(rpm_model, h=12)) +
                  as.data.frame(forecast(rpm_res_model, h=12)))
plot(rpm_ts, xlim=c(2000, 2021), ylim=c(320, 700), xlab = "Year", ylab = "Millions of Miles",
     main="12-step ahead forecast of Rail Passenger Miles")
# Point forecast
lines(x=as.yearmon(rownames(rpm_forecast)), rpm_forecast$`Point Forecast`, col="blue")
# 80% CI
lines(x=as.yearmon(rownames(rpm_forecast)), rpm_forecast$`Lo 80`, col="orange")
lines(x=as.yearmon(rownames(rpm_forecast)), rpm_forecast$`Hi 80`, col="orange")
# 90% CI
lines(x=as.yearmon(rownames(rpm_forecast)), rpm_forecast$`Lo 95`, col="yellow")
lines(x=as.yearmon(rownames(rpm_forecast)), rpm_forecast$`Hi 95`, col="yellow")
legend("topleft", legend=c("Point Forecast", "80% CI", "90% CI"),
       col=c("blue", "orange", "yellow"), lwd=2)
```



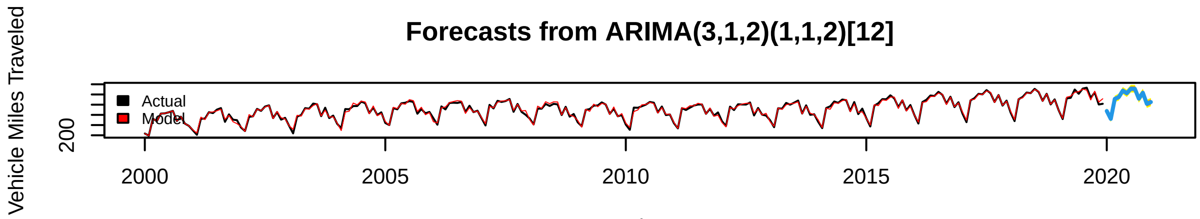
Both forecasts appear to be suitable since they capture the trend and seasonality of the data, and appear as reasonable estimates. The error bands are also relatively small.

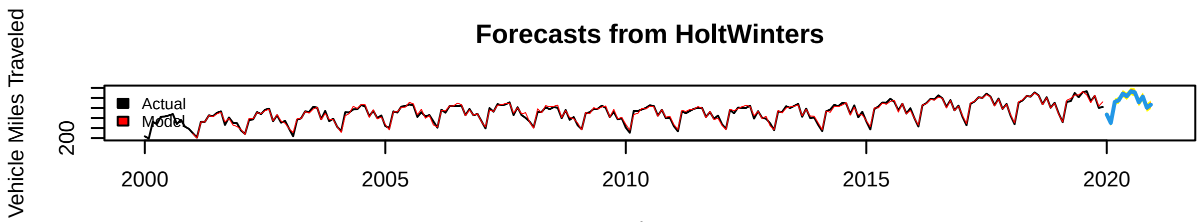
(j)

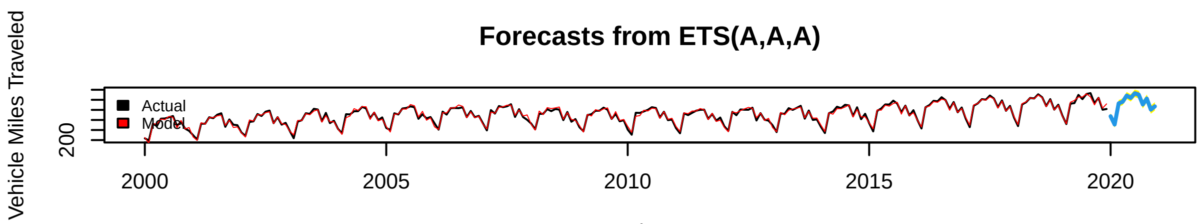
Compare your forecast from (i) to the 12-steps ahead forecasts from ARIMA, Holt-Winters, and ETS models. Which model performs best in terms of MAPE?

```
# Vehicle Miles Traveled
par(mfrow = c(3,1))

# ARIMA
fit_arima <- auto.arima(vmt_ts)
fit_arima <- forecast(fit_arima, h = 12)
plot(fit_arima, shadecols="oldstyle", ylab = "Vehicle Miles Traveled",
     xlab = "Time", xlim = c(2000,2021))
lines(fit_arima$fitted, col = "red", lwd = 0.5)
legend("topleft", legend = c("Actual", "Model"), fill = c("black","red"), cex = 0.7, bty = 'n')

# Holt-Winters
fit_hw <- HoltWinters(vmt_ts)
fit_hw <- forecast(fit_hw, h = 12)
plot(fit_hw, shadecols="oldstyle", ylab = "Vehicle Miles Traveled",
     xlab = "Time", xlim = c(2000,2021))
lines(fit_hw$fitted, col = "red", lwd = 0.5)
```

```r
legend("topleft", legend = c("Actual", "Model"), fill = c("black","red"), cex = 0.7, bty = 'n')

# ETS
fit_ets <- ets(vmt_ts)
fit_ets <- forecast(fit_ets, level = c(50,80,95), h = 12)
plot(fit_ets, shadecols="oldstyle", ylab = "Vehicle Miles Traveled", xlab = "Time")
lines(fit_ets$fitted, col = "red", lwd = 0.5)
legend("topleft", legend = c("Actual", "Model"), fill = c("black","red"), cex = 0.7, bty = 'n')
```








```r
# Summarize RMSE
table <- rbind(accuracy(fit_arima), accuracy(fit_hw), accuracy(fit_ets))
rownames(table) <- c("ARIMA", "Holt-Winters", "ETS")
table
```

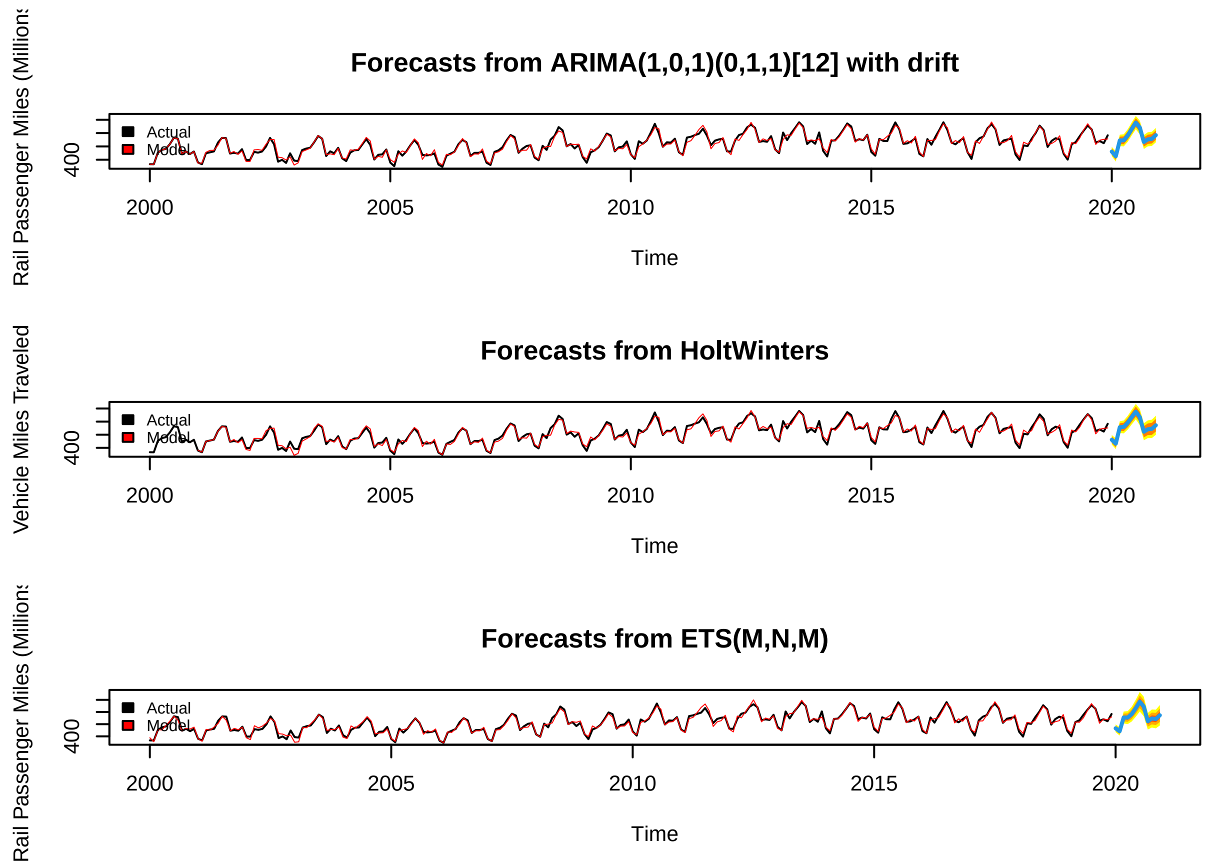
```
##                       ME     RMSE      MAE          MPE      MAPE      MASE
## ARIMA        -0.09216196 2.753480 2.164145 -0.053337305 0.8710721 0.5645706
## Holt-Winters -0.20796591 3.043999 2.395961 -0.100009404 0.9657168 0.6250455
## ETS           0.01469050 3.022237 2.432325 -0.009619984 0.9847353 0.6345319
##                    ACF1
## ARIMA        0.01901436
## Holt-Winters 0.03788958
## ETS          0.04822831
```

```
vmt_accuracy
```

```
##               ME     MSE   RMSE      MAE         MPE      MAPE     MASE
## [1,] -0.1014627 7.01455 2.6485 2.104444 -0.03751458 0.8523134 0.548996
##          RMSSE
## [1,] 0.5748218
```

Our model performs the best in terms of MAPE, with a score of 0.8523134. The ARIMA model performs the second best for the Vehicle Miles Traveled data in terms of MAPE, scoring 0.8724460 while ETS performs the worst, scoring 1.0039406. All three methods have relatively similar forecasts, and all look like reasonable estimates. As a result, the measures of error are close for all three methods, though our method performs best.

```
# Rail Passenger Miles
par(mfrow = c(3,1))

# ARIMA
fit_arima <- auto.arima(rpm_ts)
fit_arima <- forecast(fit_arima, h = 12)
plot(fit_arima, shadecols="oldstyle", ylab = "Rail Passenger Miles (Millions)",
     xlab = "Time", xlim = c(2000,2021))
lines(fit_arima$fitted, col = "red", lwd = 0.5)
legend("topleft", legend = c("Actual", "Model"), fill = c("black","red"), cex = 0.7, bty = 'n')

# Holt-Winters
fit_hw <- HoltWinters(rpm_ts)
fit_hw <- forecast(fit_hw, h = 12)
plot(fit_hw, shadecols="oldstyle", ylab = "Vehicle Miles Traveled",
     xlab = "Time", xlim = c(2000,2021))
lines(fit_hw$fitted, col = "red", lwd = 0.5)
legend("topleft", legend = c("Actual", "Model"), fill = c("black","red"), cex = 0.7, bty = 'n')

# ETS
fit_ets <- ets(rpm_ts)
fit_ets <- forecast(fit_ets, level = c(50,80,95), h = 12)
plot(fit_ets, shadecols="oldstyle", ylab = "Rail Passenger Miles (Millions)", xlab = "Time")
lines(fit_ets$fitted, col = "red", lwd = 0.5)
legend("topleft", legend = c("Actual", "Model"), fill = c("black","red"), cex = 0.7, bty = 'n')
```




```
# Summarize RMSE
table <- rbind(accuracy(fit_arima), accuracy(fit_hw), accuracy(fit_ets))
rownames(table) <- c("ARIMA", "Holt-Winters", "ETS")
table
```

```
##                      ME     RMSE      MAE         MPE     MAPE      MASE
## ARIMA        -0.1952914 17.24018 13.53683 -0.18474500 2.690765 0.6134051
## Holt-Winters  1.1947256 18.55789 14.86262  0.09640864 2.922895 0.6734820
## ETS           0.7419950 16.78456 13.27792  0.05190769 2.639861 0.6016729
##                     ACF1
## ARIMA        0.001428288
## Holt-Winters 0.079918332
## ETS          0.009727773
```

```
rpm_accuracy
```

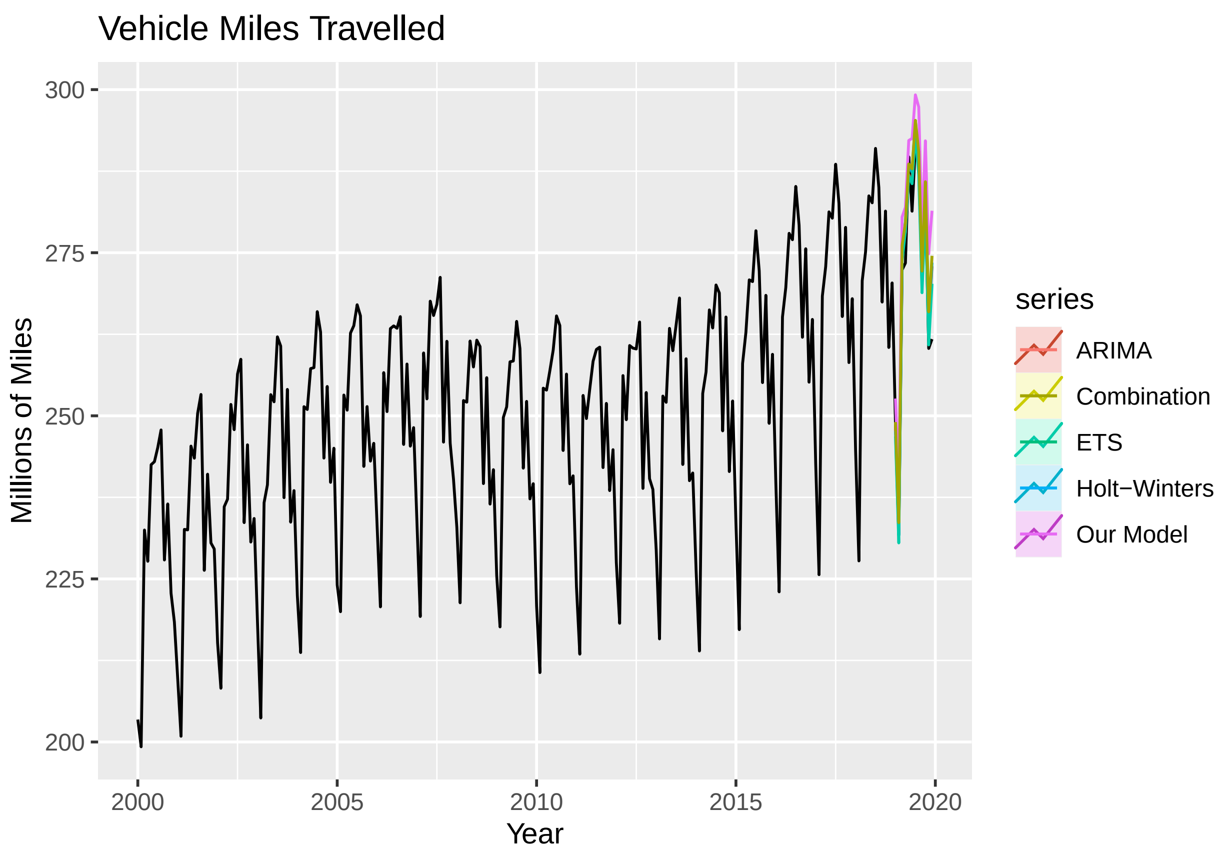
```
##             ME      MSE     RMSE      MAE        MPE     MAPE      MASE
## [1,] 0.2507227 271.8764 16.48867 13.05443 0.06858536 2.619336 0.5915457
##          RMSSE
## [1,] 0.5865631
```

Again, our model performs slightly better in terms of MAPE, with a score of 2.619336. The ETS model performs the second best for the Rail Passenger Miles data in terms of MAPE, scoring 2.658060 while Holt-Winters performs the worst, scoring 2.941808. The ETS model captures some of the irregularity in the data better, resulting in better performance, although all three methods produce similar forecasts.

(k)

Combine the four forecasts and comment on the MAPE from this forecasts vs., the individual ones.

```r
# Vehicle Miles Travelled

# Make the training set, till 12 months before end
train.vmt = window(vmt_ts, end=c(2018,12))

# 5-step ahead forecast for testing
h <- length(vmt_ts) - length(train.vmt)

# Getting the forecasts for different models
our_trend = forecast(tslm(train.vmt ~ trend + I(trend^2) + I(trend^3) + season), h=h)
our_res = forecast(auto.arima(our_trend$residuals), h=h)
fit_our_model = our_trend[["mean"]] + our_res[["mean"]]
fit_arima = forecast(auto.arima(train.vmt), h=h)
fit_hw = forecast(HoltWinters(train.vmt), h=h)
fit_ets = forecast(ets(train.vmt), h=h)

# Get the combination forecast
Combination <- (fit_our_model + fit_arima[["mean"]] + fit_hw[["mean"]] + fit_ets[["mean"]]) / 4

# Plot forecasts from models
autoplot(vmt_ts) +
  autolayer(fit_arima, series="ARIMA", PI=FALSE) +
  autolayer(fit_hw, series="Holt-Winters", PI=FALSE) +
  autolayer(fit_ets, series="ETS", PI=FALSE) +
  autolayer(fit_our_model, series="Our Model") +
  autolayer(Combination, series="Combination") +
  xlab("Year") + ylab("Millions of Miles") +
  ggtitle("Vehicle Miles Travelled")
```



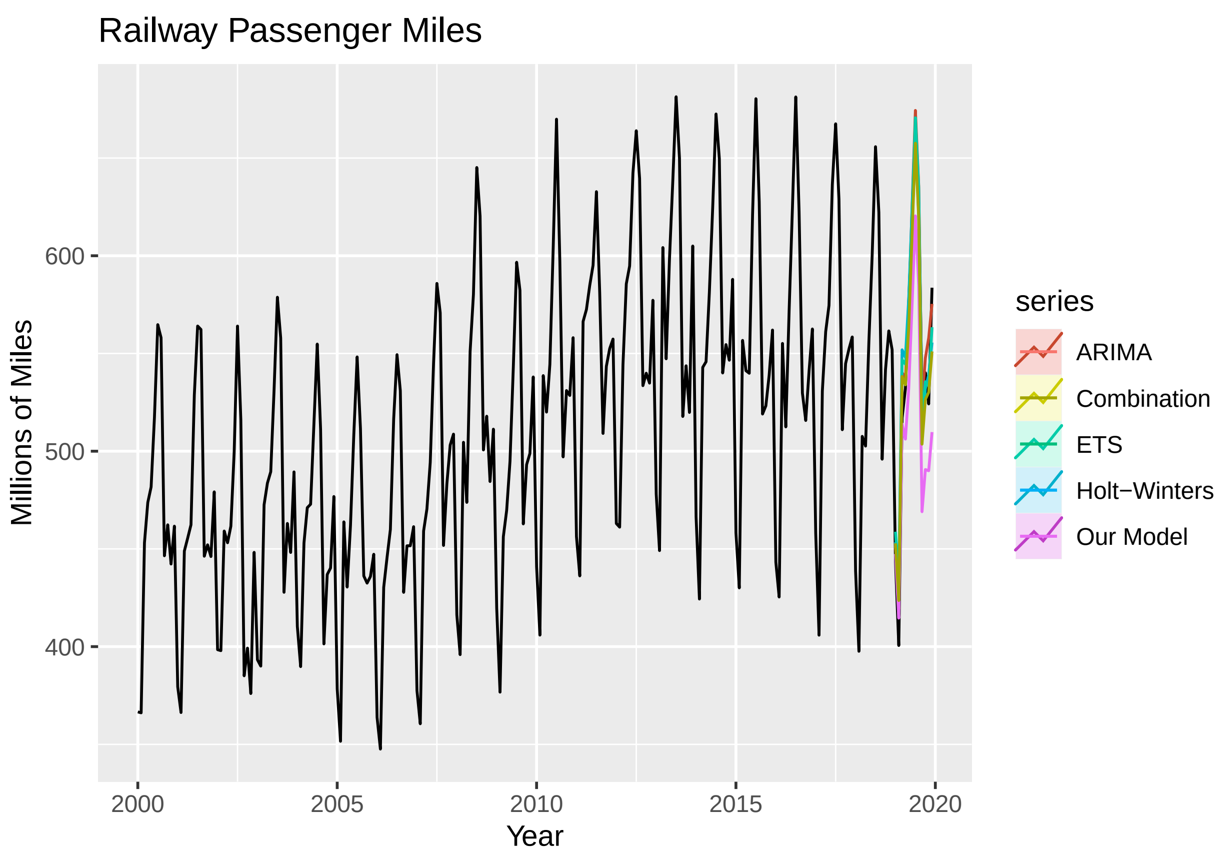

```
# Get the MAPE for forecast
c(ETS = accuracy(fit_ets, vmt_ts)["Test set","MAPE"],
  ARIMA = accuracy(fit_arima, vmt_ts)["Test set","MAPE"],
  HoltWinters = accuracy(fit_hw, vmt_ts)["Test set","MAPE"],
  Ours = accuracy(fit_our_model, vmt_ts)["Test set", "MAPE"],
  Combination = accuracy(Combination, vmt_ts)["Test set","MAPE"])
```

```
##         ETS       ARIMA HoltWinters        Ours Combination
##    1.225350    1.252163    1.336817    3.118145    1.458393
```

Combination forecast has the second lowest MAPE. ETS gives the best model and ARIMA gives the second-best model. Our model performs much worse than all three, but combining the models together gives a much better estimate than ours.

```
# Railway Passenger Miles

# Make the training set, till 12 months before end
train.rpm = window(rpm_ts, end=c(2018,12))

# 5-step ahead forecast for testing
h <- length(rpm_ts) - length(train.rpm)

# Getting the forecasts for different models
our_trend = forecast(tslm(train.rpm ~ trend + I(trend^2) + I(trend^3) + season), h=h)
our_res = forecast(auto.arima(our_trend$residuals), h=h)
```

```
fit_our_model = our_trend[["mean"]] + our_res[["mean"]]
fit_arima = forecast(auto.arima(train.rpm), h=h)
fit_hw = forecast(HoltWinters(train.rpm), h=h)
fit_ets = forecast(ets(train.rpm), h=h)

# Get the combination forecast
Combination <- (fit_our_model + fit_arima[["mean"]] + fit_hw[["mean"]] + fit_ets[["mean"]]) / 4

# Plot forecasts from models
autoplot(rpm_ts) +
  autolayer(fit_arima, series="ARIMA", PI=FALSE) +
  autolayer(fit_hw, series="Holt-Winters", PI=FALSE) +
  autolayer(fit_ets, series="ETS", PI=FALSE) +
  autolayer(fit_our_model, series="Our Model") +
  autolayer(Combination, series="Combination") +
  xlab("Year") + ylab("Millions of Miles") +
  ggtitle("Railway Passenger Miles")
```




```
# Get the MAPE for forecast
c(ETS = accuracy(fit_ets, rpm_ts)["Test set","MAPE"],
  ARIMA = accuracy(fit_arima, rpm_ts)["Test set","MAPE"],
  HoltWinters = accuracy(fit_hw, rpm_ts)["Test set","MAPE"],
  Ours = accuracy(fit_our_model, rpm_ts)["Test set", "MAPE"],
  Combination = accuracy(Combination, rpm_ts)["Test set","MAPE"])
```

```
##         ETS       ARIMA HoltWinters        Ours Combination
##    2.803617    2.334502    2.896450    6.361022    2.579000
```

Our model has a much worse MAPE than the other three methods, because it predicts that the downward trend at the end of the training period will continue into the forecast period when it does not. The other models all have the capability to adjust once the trend changes (because they are based on autoregression), while ours does not. ARIMA performs the best of all three methods, and surprisingly, despite the inclusion of our model, the Combination model performs the second best. This means that all three models with worse MAPE (ETS, HW, and Ours) combined with ARIMA produced a better approximation than any of them individually.

(l)

Fit an appropriate VAR model using your two variables. Make sure to show the relevant plots and discuss your results from the fit.

```r
# Combine the variables
y = cbind(vmt_ts, rpm_ts)
y_tot = data.frame(y)

# Select VAR model
vars::VARselect(y_tot, lag.max = 30)
```

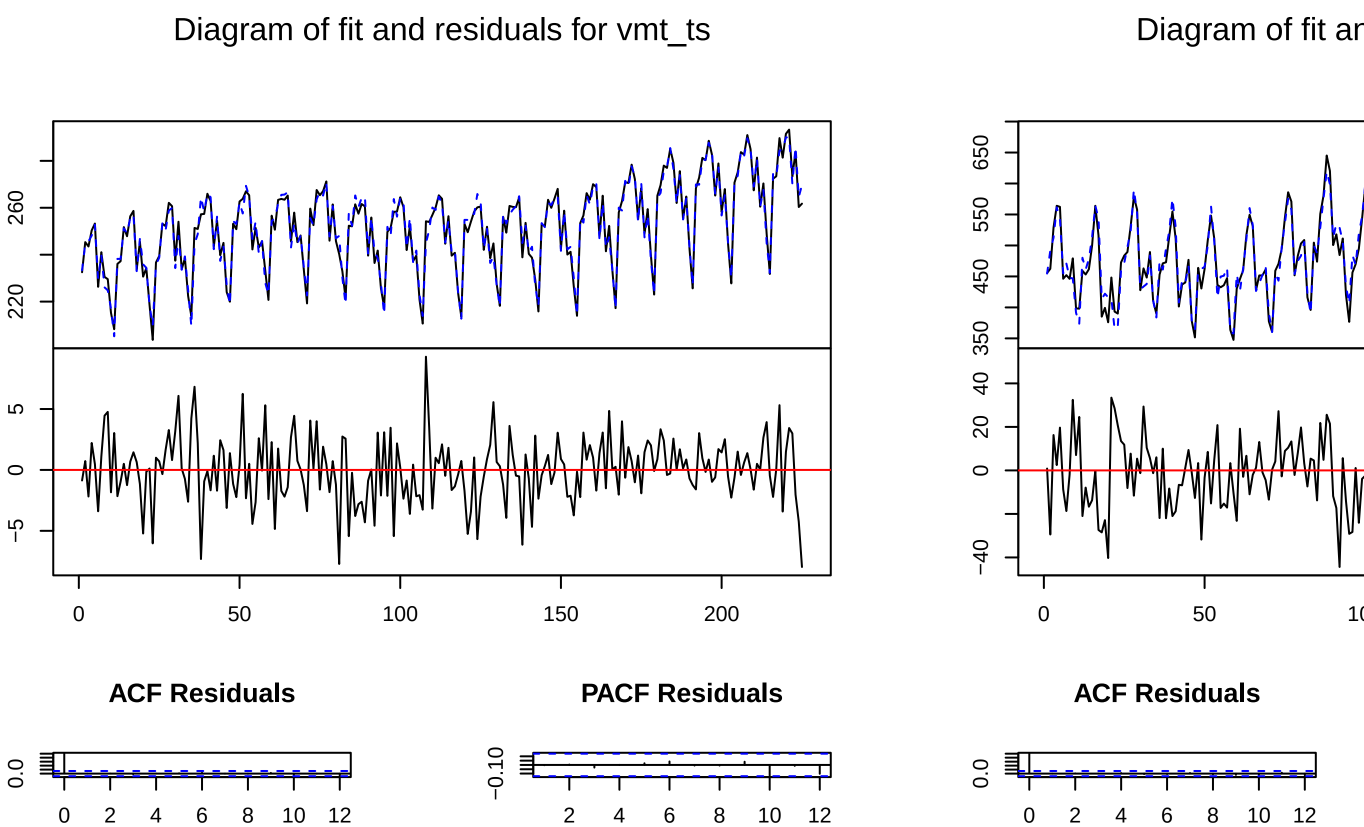
```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     15     14     14     15
##
## $criteria
##                    1            2            3            4           5
## AIC(n)      12.51850     12.37283     12.08408     11.94000    11.41079
## HQ(n)       12.55716     12.43727     12.17428     12.05598    11.55254
## SC(n)       12.61414     12.53222     12.30722     12.22689    11.76144
## FPE(n) 273349.83765 236298.12828 177039.08008 153292.62204 90307.80396
##                   6           7           8            9           10
## AIC(n)     11.30621    10.73467    10.44844     9.507178     9.414523
## HQ(n)      11.47374    10.92797    10.66752     9.752027     9.685146
## SC(n)      11.72062    11.21283    10.99036    10.112845    10.083945
## FPE(n) 81351.06373 45943.08681 34515.19727 13469.331044 12281.690253
##                  11          12          13          14          15          16
## AIC(n)     9.117398    8.472740    8.332248    8.267532    8.250067    8.280993
## HQ(n)      9.413794    8.794910    8.680191    8.641249    8.649557    8.706256
## SC(n)      9.850574    9.269671    9.192932    9.191971    9.238261    9.332941
## FPE(n) 9128.549750 4793.467471 4167.659362 3909.185559 3844.555376 3968.901509
##                  17          18          19          20          21          22
## AIC(n)     8.298433    8.276720    8.294596    8.302744    8.292531    8.270961
## HQ(n)      8.749471    8.753530    8.797181    8.831101    8.846662    8.850866
## SC(n)      9.414136    9.456177    9.537808    9.609709    9.663251    9.705436
## FPE(n) 4042.874252 3960.601922 4037.245348 4076.115444 4041.109457 3961.803781
##                  23          24          25          26          27          28
## AIC(n)     8.298761    8.318031    8.346589    8.331387    8.353065    8.297086
## HQ(n)      8.904440    8.949483    9.003814    9.014386    9.061838    9.031632
## SC(n)      9.796990    9.880014    9.972327   10.020879   10.106312   10.114087
```

```
## FPE(n) 4081.316987 4169.471877 4300.095900 4245.773677 4350.540922 4125.707090
##                  29          30
## AIC(n)     8.320611    8.318571
## HQ(n)      9.080930    9.104664
## SC(n)     10.201366   10.263081
## FPE(n) 4237.226064 4242.937386
```

```r
# Fit a VAR model
y_model = vars::VAR(y_tot, p = 15)
```

We will use a VAR(15) model based on both AIC and FPE.

```r
# Note: I've spent an hour trying to get these plots to display separately and they won't.
# It keeps telling me the margins are too large, no matter what I change them to.
# This is the best I can do to see both of them.
plot(y_model)
```



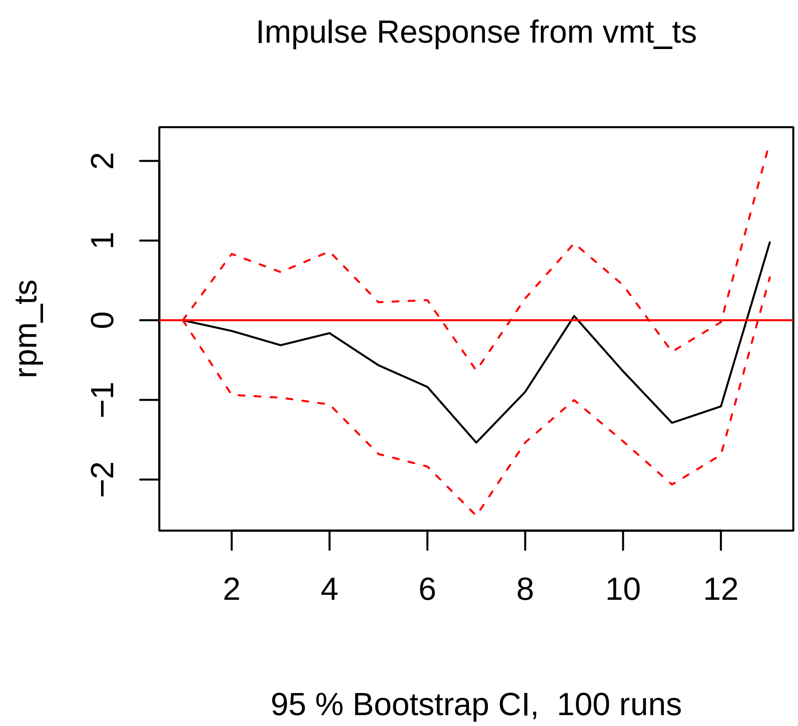
- The first plot shows VAR(15) for Vehicle Miles Traveled. The VAR(15) model seem to fit the Vehicle Miles Traveled data well as it closely follows the variations in the data. The residuals are stationary about 0, although their variance does not seem to be constant (especially the last observations in 2020). There are no spikes in the ACF and PACF plots, indicating no statistically significant time dependence in the residuals.

- The second plot shows VAR(15) for Rail Passenger Miles. The VAR(15) model seem to fit the Rail Passenger Miles data well as it closely follows the variations in the data. The residuals are stationary about 0, although their variance does not seem to be constant. There are no spikes in the ACF and PACF plots, indicating no statistically significant time dependence in the residuals.

(m)

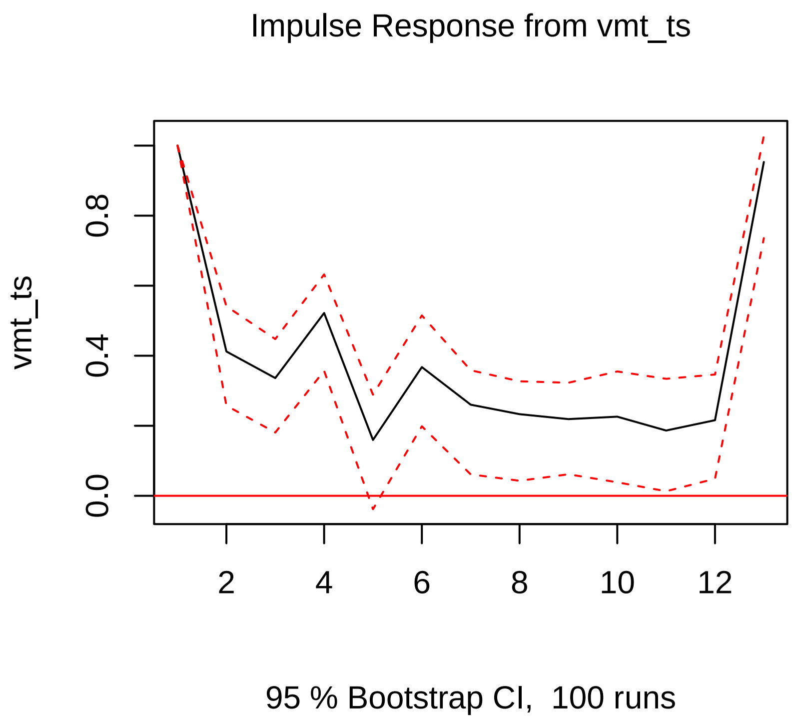
Compute, plot, and interpret the respective impulse response functions.

```
plot(vars::irf(y_model, impulse = "vmt_ts", response = "rpm_ts", n.ahead=12, ortho = FALSE))
```



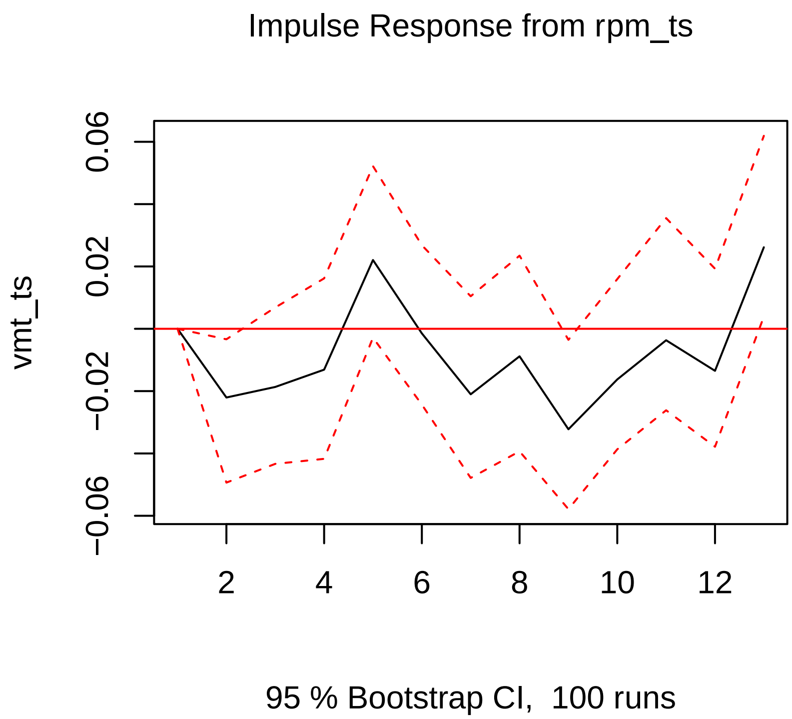
Effect of vmt_ts shock on rpm_ts: initially little response, then a decrease around lag 7 before increasing sharply after lag 12.

```
plot(vars::irf(y_model, impulse = "vmt_ts", response = "vmt_ts", n.ahead=12, ortho = FALSE))
```



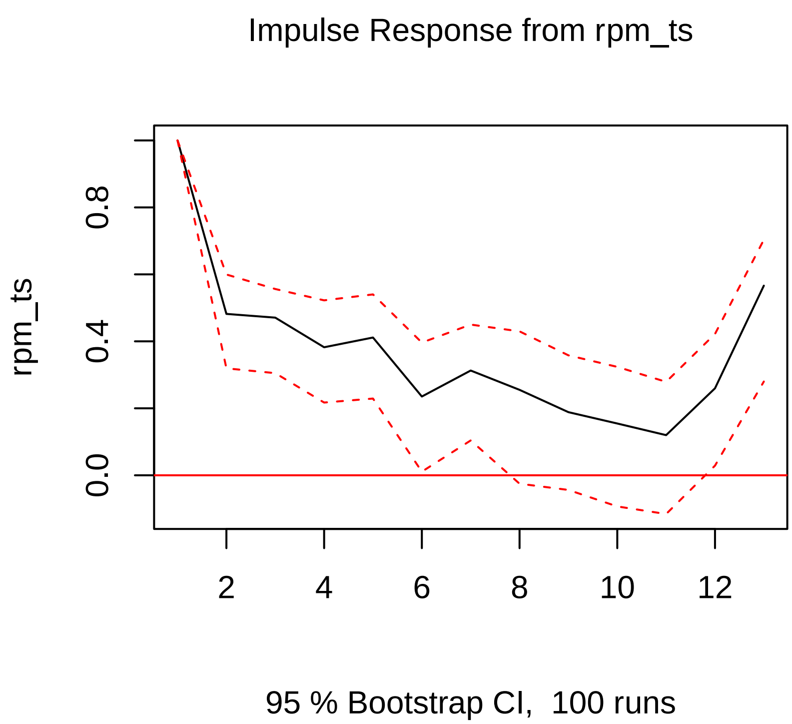
Effect of vmt_ts shock on vmt_ts: initially a large effect lessens around lag 5 and stays relatively low.

```
plot(vars::irf(y_model, impulse = "rpm_ts", response = "vmt_ts", n.ahead=12, ortho = FALSE))
```



Effect of rpm_ts shock on vmt_ts: very little effect throughout, somewhat negative.

```
plot(vars::irf(y_model, impulse = "rpm_ts", response = "rpm_ts", n.ahead=12, ortho = FALSE))
```



effect of rpm_ts shock on rpm_ts: initially strong effect which gradually decreases until lag 11, remaining positive.

(n)

Perform a Granger-Causality test on your variables and discuss your results from the test.

```
grangertest(vmt_ts ~ rpm_ts, order = 15)
```

```
## Granger causality test
##
## Model 1: vmt_ts ~ Lags(vmt_ts, 1:15) + Lags(rpm_ts, 1:15)
## Model 2: vmt_ts ~ Lags(vmt_ts, 1:15)
##   Res.Df  Df      F    Pr(>F)
## 1    194
## 2    209 -15 2.9929 0.0002497 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
grangertest(rpm_ts ~ vmt_ts, order = 15)
```

```
## Granger causality test
##
## Model 1: rpm_ts ~ Lags(rpm_ts, 1:15) + Lags(vmt_ts, 1:15)
```

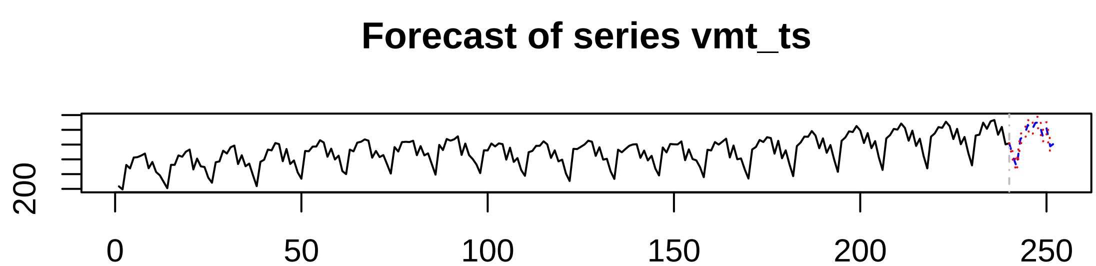
```
## Model 2: rpm_ts ~ Lags(rpm_ts, 1:15)
##   Res.Df  Df      F    Pr(>F)
## 1    194
## 2    209 -15 5.8897 5.323e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

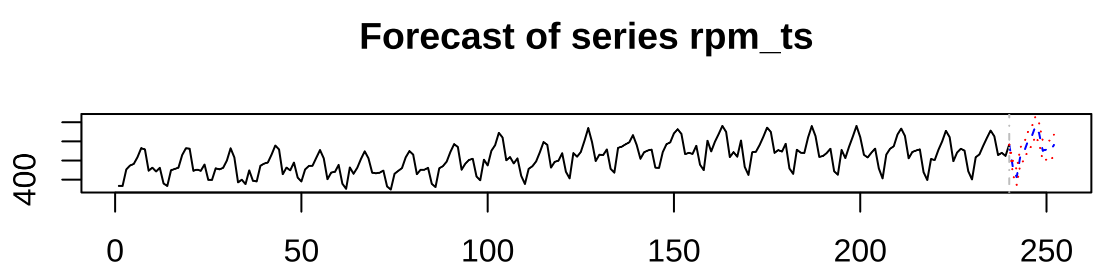
We use a Granger Test to determine causality. It seems as though both relationships are statistically significant as the p-values as very small and quite significant. We could conclude that both data sets "Granger-cause" each other and contain significant information that helps predict the other series. This is not unsurprising, as the choice to travel in general may influence both data sets, and specifically the choice of one mode of transportation over another could likely impact both data sets as well.

(o)

Use your VAR model to forecast 12-steps ahead. Your forecast should include the respective error bands. Comment on the differences between the VAR forecast and the other ones obtained using the different methods.

```r
var.predict = predict(object=y_model, n.ahead=12)
plot(var.predict)
```





produces a similar forecast compared to other methods and captures the seasonal fluctuations in the data. However, while the single-series models predicted increased peaks in the following year, the VAR model seem to predict a decreasing trend compared to other methods, expecting both series trends to decrease or remain the same in the next year.

III. Conclusions and Future Work.

We conclude that ARIMA models provide a good tool for forecasting time series data. Further, we find that forecast combinations can be a useful tool to provide more accurate forecasts than just a single method. We also conclude that VAR methods are useful for comparing similar data sets and capturing information contained in each other.

Future work comparing vehicle miles and rail passenger miles may take into account confounding factors that could impact both data sets, like general economic conditions (including recessions) that affect the transportation of goods, as that would likely impact both data sets, as well as the impact of transportation costs, including petroleum products, on miles traveled. We believe these could provide significant information that would aid in forecasting future transportation demand.

IV. References

1. U.S. Federal Highway Administration, Vehicle Miles Traveled [TRFVOLUSM227NFWA], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/TRFVOLUSM227NFWA, February 21, 2022.
2. U.S. Bureau of Transportation Statistics, Rail Passenger Miles [RAILPM], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/RAILPM, February 21, 2022.

V. R Source Code

Our R source code is included in the document throughout, with comments.