ECON 144: Project 3

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

03/10/22

Contents


```
rm(list=ls(all=TRUE))
library(stats)
library(xts)
library(tis)
library(strucchange)
library(quantmod)
library(forecast)
library(feasts)
library(fable)
library(fabletools)
library(fable.prophet)
library(ggplot2)
```

I. Introduction

Dataset chosen for this project:

Retail Sales: Electronic Shopping and Mail-Order Houses (Monthly, Millions of Dollars, Not Seasonally Adjusted)

> Electronic Shopping and Mail-Order Houses comprises establishments primarily engaged in retailing all types of merchandise using nonstore means, such as catalogs, toll free telephone numbers, or electronic media, such as interactive television or the Internet.

With the rise of internet shopping in the last decade (especially since the start of 2020), and as with any retail data set in general, we should expect to see interesting dynamics in the data that should provide us with a good opportunity to explore different models.

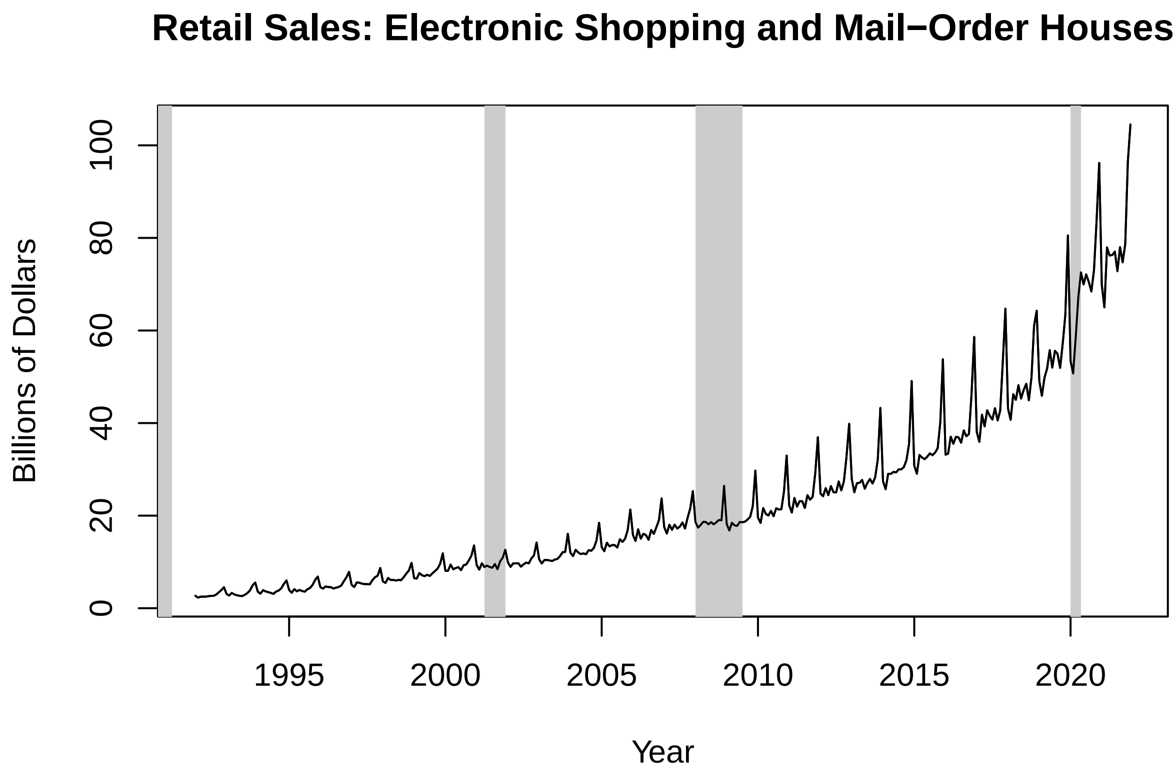
II. Results

Inspect the data

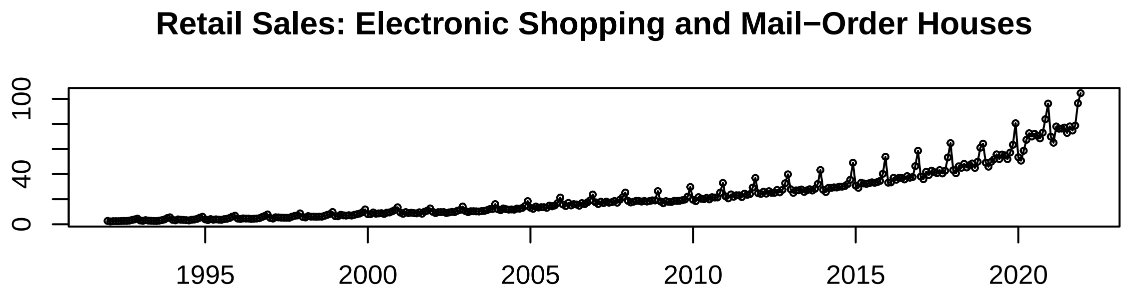
```
# Time-series plot of data & the respective ACF and PACF plots
getSymbols("MRTSSM4541USN", src = "FRED")
```

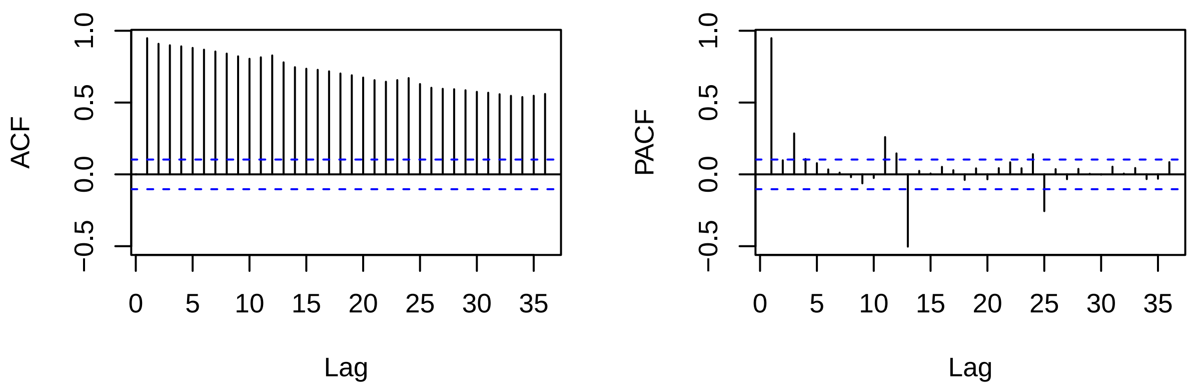
```
## [1] "MRTSSM4541USN"
```

```
rs_ts <- ts(MRTSSM4541USN, start = 1992, freq = 12)

# Rescale the data (Millions -> Billions)
rs_ts <- rs_ts / 1000

# Plot data with recession periods
plot(rs_ts, main = "Retail Sales: Electronic Shopping and Mail-Order Houses",
     xlab = "Year", ylab = "Billions of Dollars")
nberShade()
lines(rs_ts)
```




```r
# View ACF and PACF
tsdisplay(rs_ts, main = "Retail Sales: Electronic Shopping and Mail-Order Houses")
```



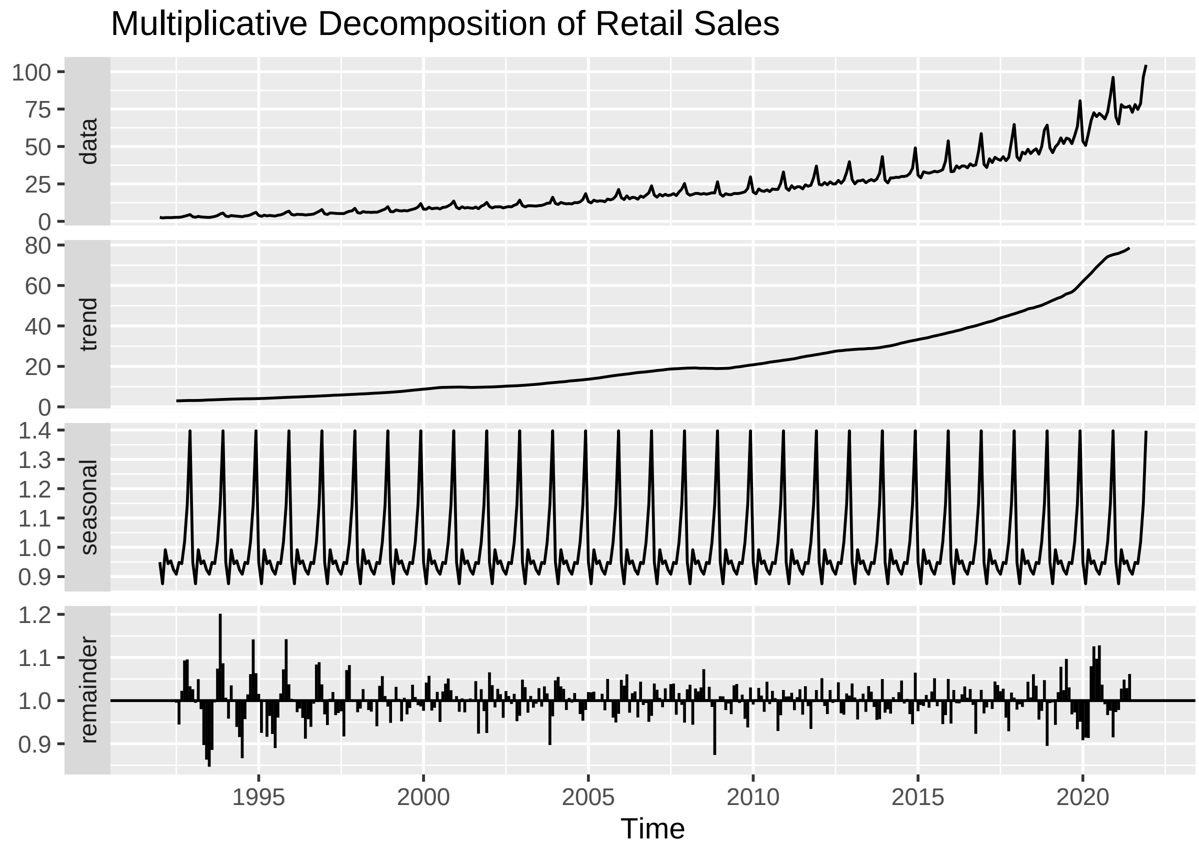


The series shows a strong upward quadratic trend with clear seasonality. The seasonality appears to be multiplicative, since the seasonal fluctuations vary with time. The ACF slowly decays to 0 while the PACF shows several strong spikes at regular intervals (e.g. k = 1, 13, 25, . . .).

We can also observe some cyclical behavior in the ACF plot, which contains small regular spikes at yearly lags (12, 24, etc.). Whichever models we choose will have to account for a (possibly non-linear) trend, yearly seasonality, and any other cycles in the data.

We will examine a multiplicative decomposition to see if it confirms our intial observations:

```
# Decomposition
dcmp <- decompose(rs_ts, type = "multiplicative")
autoplot(dcmp, main = "Multiplicative Decomposition of Retail Sales")
```

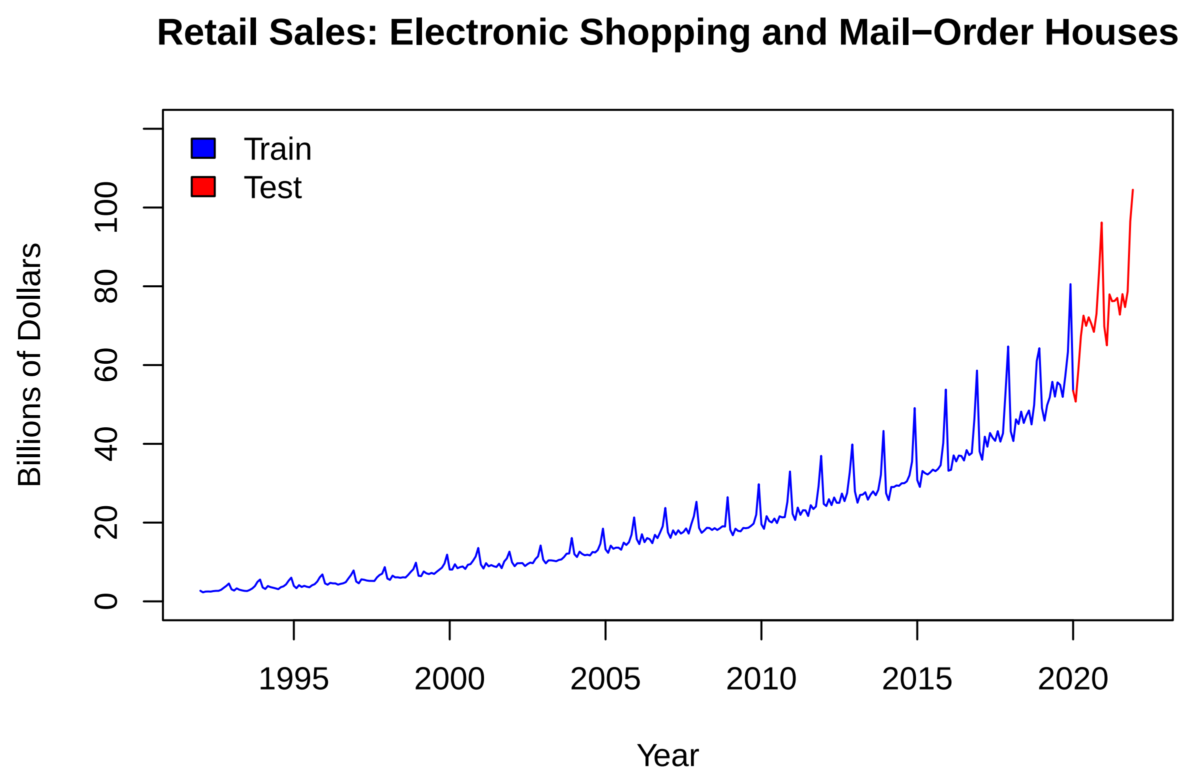


The multiplicative decomposition confirms our initial analysis that:

1. There is a strong quadratic upward trend. The trend explains most of the variation in the data, as evident by the y-axis scale compared to the original data.
2. There is strong seasonality in the data. The seasonal component explains a good amount of the variation in the data, as evident by the relatively large size of the multiplicative amplitude (up to 1.4).
3. There may be presence of cycles in the data, as the remainder component appears to vary cyclically. The remainder component is especially high near the beginning and end of the series. This may be due to the restriction on the decomposition of a static seasonal component.

For model testing and fitting, we can split the data into testing and training sets:

```r
# Training and testing data
rs_train <- window(rs_ts, end = c(2020, 1))
rs_test <- window(rs_ts, start = c(2020, 1))

# Check if data have been split correctly
plot(rs_train, xlab="Year", ylab="Billions of Dollars", xlim=c(1992, 2022), ylim=c(0, 120),
     col="blue", main="Retail Sales: Electronic Shopping and Mail-Order Houses")
lines(rs_test, col = "red")
legend("topleft", legend = c("Train", "Test"), fill = c("blue","red"), cex = 1, bty = 'n')
```



The sudden jump at the beginning of the test set may pose a problem for some of the models. We will have to see how they handle it.

For our first model, we can try fitting a quadratic trend: $y = b_0 + b_1 t + b_2 t^2$ with seasonal dummies and a model for the cycles.

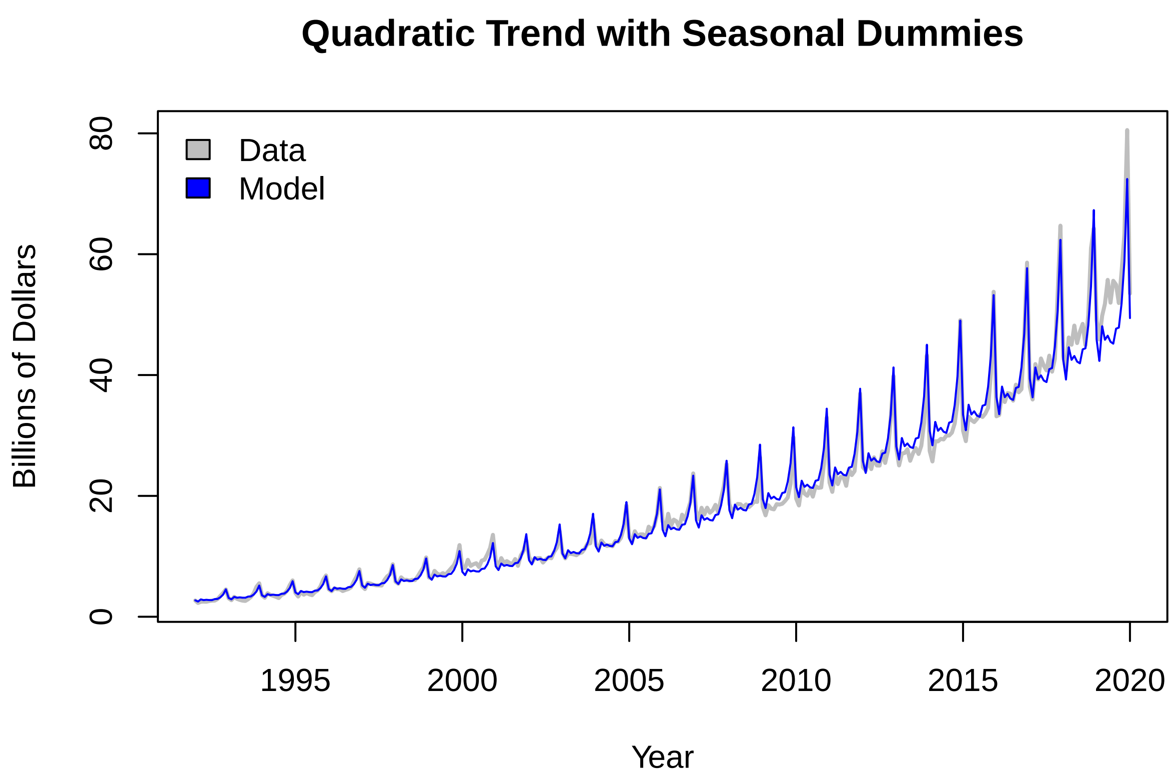
Custom Model: Quadratic Trend + Seasonal Dummies + Cycles

```
# Try a quadratic model with seasonal dummies
rs_model <- tslm(rs_train ~ trend + I(trend^2) + season, lambda = 0)
summary(rs_model)
```

```
##
## Call:
## tslm(formula = rs_train ~ trend + I(trend^2) + season, lambda = 0)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.194896 -0.051038 -0.003403  0.051093  0.207348
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.890e-01  1.765e-02  56.037  < 2e-16 ***
## trend        1.124e-02  1.652e-04  68.041  < 2e-16 ***
```

```
## I(trend^2)  -7.724e-06  4.734e-07 -16.316  < 2e-16 ***
## season2     -8.756e-02  1.949e-02  -4.493 9.79e-06 ***
## season3      3.360e-02  1.949e-02   1.724 0.085632 .
## season4     -2.046e-02  1.949e-02  -1.050 0.294592
## season5     -1.176e-02  1.949e-02  -0.604 0.546590
## season6     -3.990e-02  1.949e-02  -2.048 0.041407 *
## season7     -5.305e-02  1.949e-02  -2.722 0.006836 **
## season8     -5.672e-03  1.949e-02  -0.291 0.771188
## season9     -8.033e-03  1.949e-02  -0.412 0.680457
## season10     6.727e-02  1.949e-02   3.452 0.000631 ***
## season11     1.863e-01  1.949e-02   9.561  < 2e-16 ***
## season12     3.885e-01  1.949e-02  19.935  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07354 on 323 degrees of freedom
## Multiple R-squared:  0.9929, Adjusted R-squared:  0.9927
## F-statistic:  3497 on 13 and 323 DF,  p-value: < 2.2e-16
```

```
# Plot fitted values against actual data
plot(rs_train, main = "Quadratic Trend with Seasonal Dummies",
     xlab = "Year", ylab = "Billions of Dollars", col = "grey", lwd = 2)
lines(fitted(rs_model), col="blue")
legend("topleft", legend = c("Data", "Model"), fill = c("grey","blue"), cex = 1, bty = 'n')
```

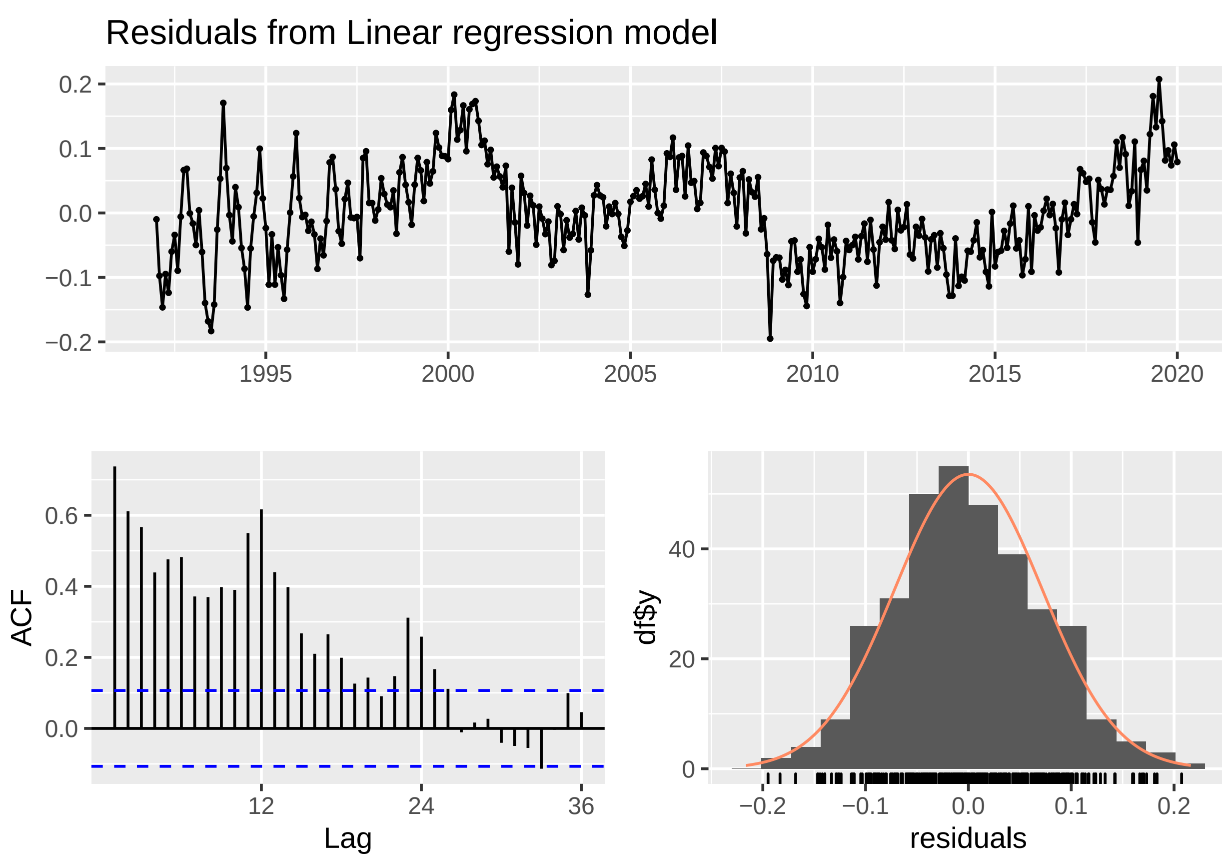


Our quadratic model with seasonal dummies seem to fit the data well.

- We included the lambda = 0 argument in the tslm model to allow our linear model to approximate the changing seasonality in the data, which it does quite well.
- The coefficients for our quadratic trend are statistically significant.
- While many of the coefficients for the seasonal dummies are not statistically significant, we still include them in our model to take into account monthly variations in the data from year to year.
- The Adjusted R-squared for this model is very high at 0.9927 This means that our model is able to explain 99% of the variation in the data.
- The plot shows that our model fits the seasonal variation well, but we can see some deviations from the model.

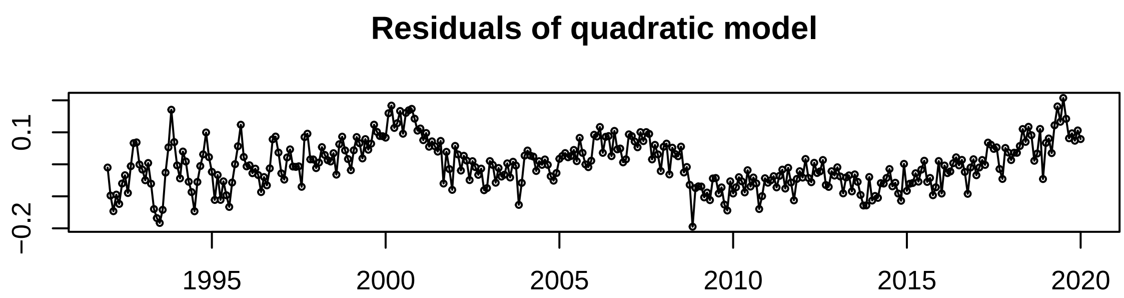
Next, we try to fit a model to the cycles in the data:

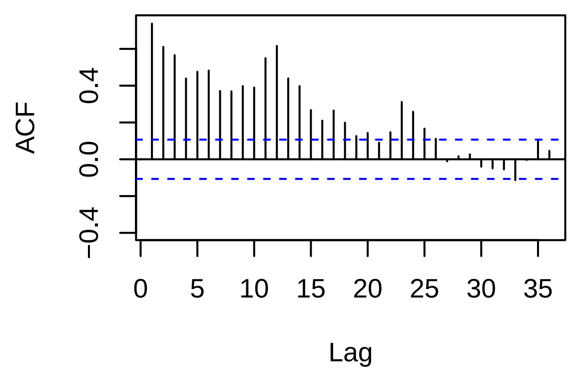
```
checkresiduals(rs_model)
```

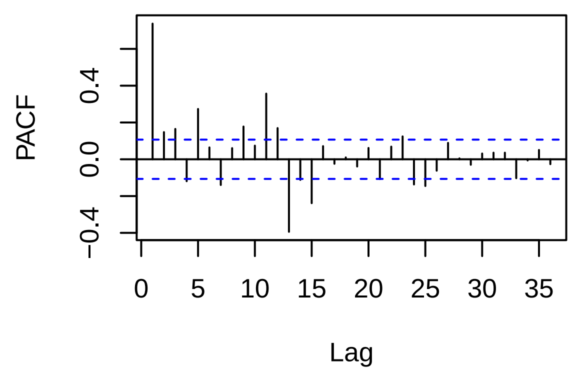



```
##
##  Breusch-Godfrey test for serial correlation of order up to 24
##
## data:  Residuals from Linear regression model
## LM test = 263.59, df = 24, p-value < 2.2e-16
```

```
tsdisplay(rs_model$residuals, main = "Residuals of quadratic model")
```





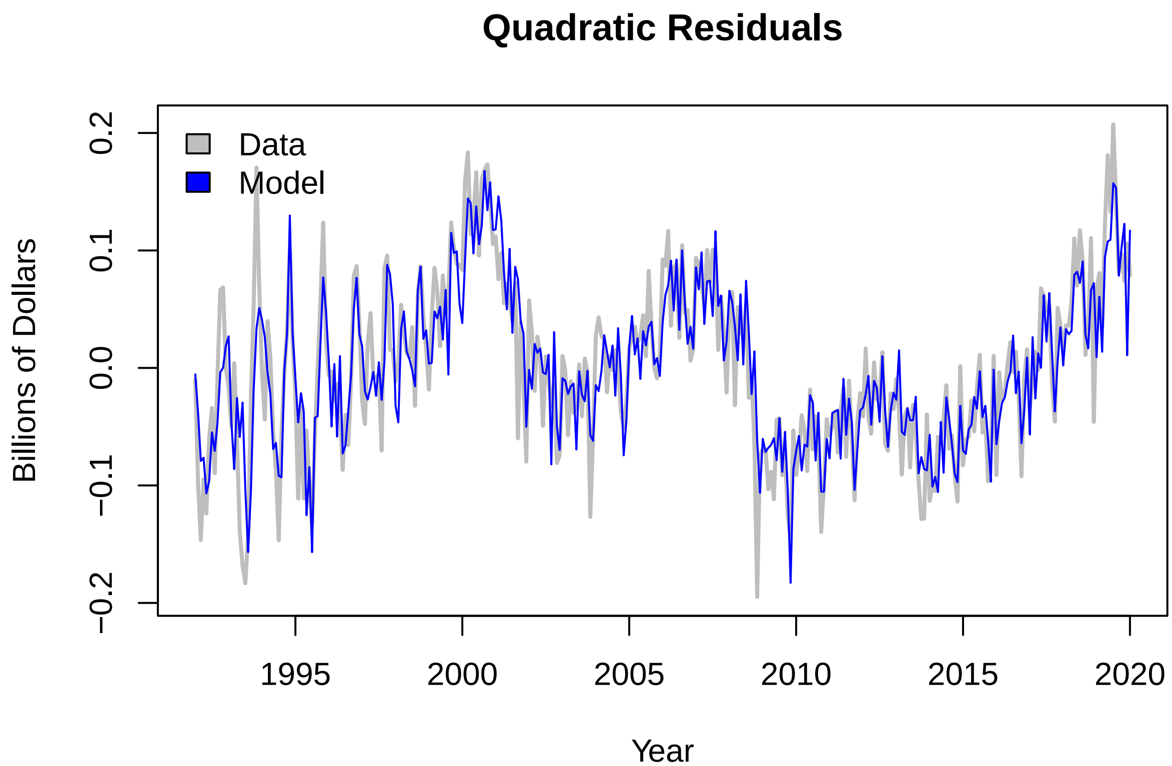


The residuals strongly appear to have some kind of persistence, since the plot does not mean revert quickly. The ACF slowly decays to 0 while the PACF has several strong spikes up to k = 15. We can try fitting an ARMA(p,q) model to the residuals.

```
rs_res_model <- auto.arima(rs_model$residuals)
summary(rs_res_model)
```

```
## Series: rs_model$residuals
## ARIMA(3,0,1)(2,0,1)[12] with zero mean
##
## Coefficients:
##           ar1     ar2     ar3     ma1    sar1     sar2     sma1
##       -0.1066  0.3457  0.4767  0.5358  1.1777  -0.3174  -0.5658
## s.e.   0.0933  0.0656  0.0526  0.1014  0.2629   0.1816   0.2524
##
## sigma^2 = 0.00139:  log likelihood = 629.86
## AIC=-1243.73   AICc=-1243.29   BIC=-1213.16
##
## Training set error measures:
##                       ME       RMSE        MAE      MPE     MAPE      MASE
## Training set 0.001001213 0.03688844 0.02841132 89.72724 158.0818 0.6371927
##                    ACF1
## Training set 0.02040428
```

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

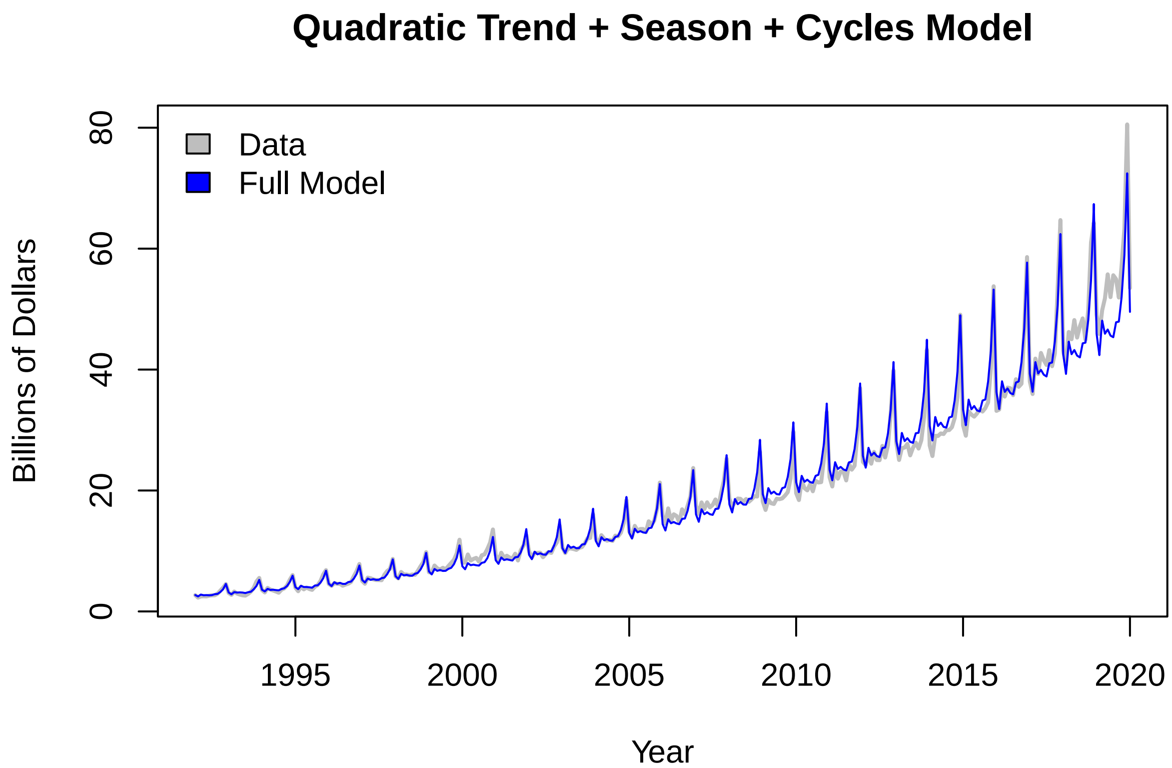


R suggests ARIMA(3,0,1)(2,0,1)[12] for the cycles in our data. Since R suggests I(0), our model successfully takes into account the trend of the data and the residuals are largely stationary. The ARMA(3,1) component takes into account the cycles in our residuals while the S-ARMA(2,1) component takes into account the seasonality in our residuals. This indicates that our cubic trend + seasonal dummies model did not fully take into account all seasonality in the data.

From the plot, we observe that our ARIMA(3,0,1)(2,0,1)[12] residual model seems to track the residuals closely and is a good fit.

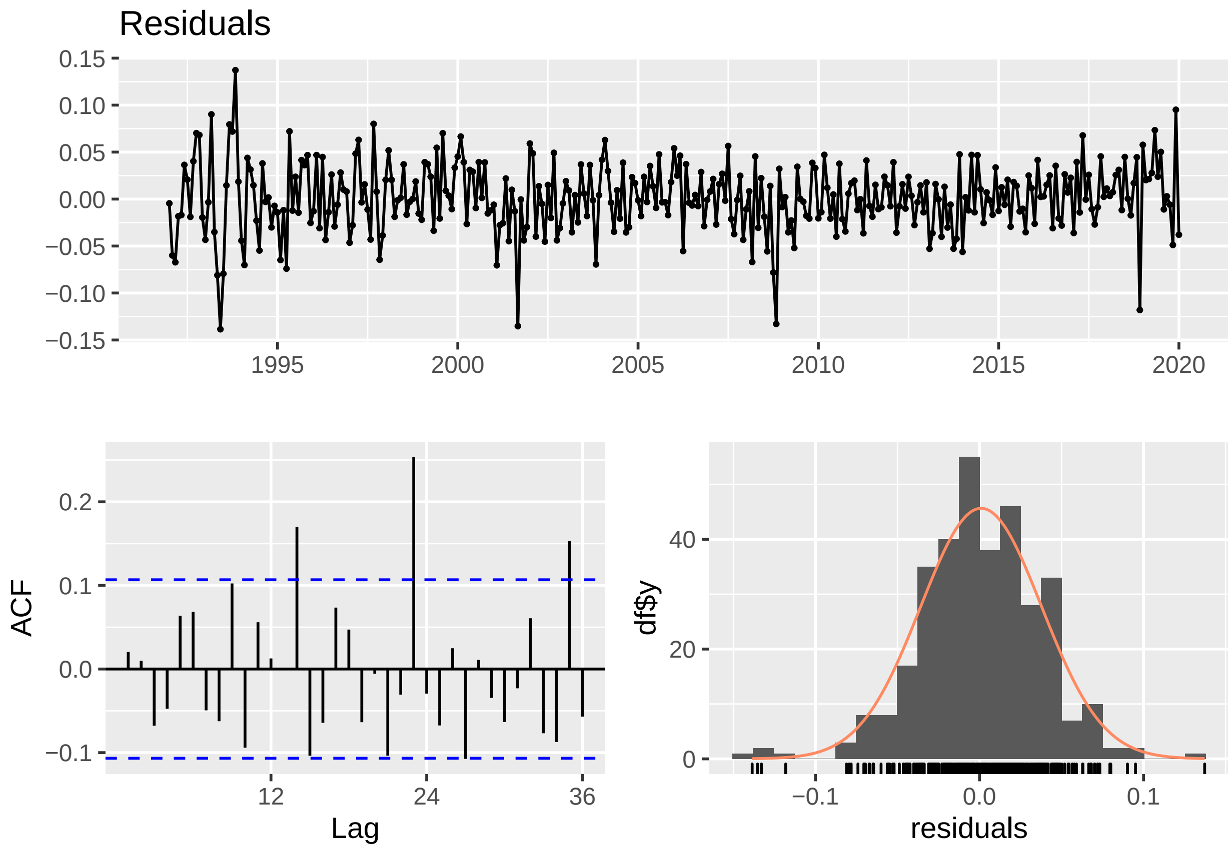
We can combine the quadratic trend + seasonal dummies model with our residual model to obtain a comprehensive model for the original data.

```r
# Plot data against our full model
plot(rs_train, main = "Quadratic Trend + Season + Cycles Model",
     xlab = "Year", ylab = "Billions of Dollars", col = "grey", lwd = 2)
lines(fitted(rs_model) + fitted(rs_res_model), col="blue")
legend("topleft", legend = c("Data", "Full Model"), fill = c("grey","blue"),
       cex = 1, bty = 'n')
```

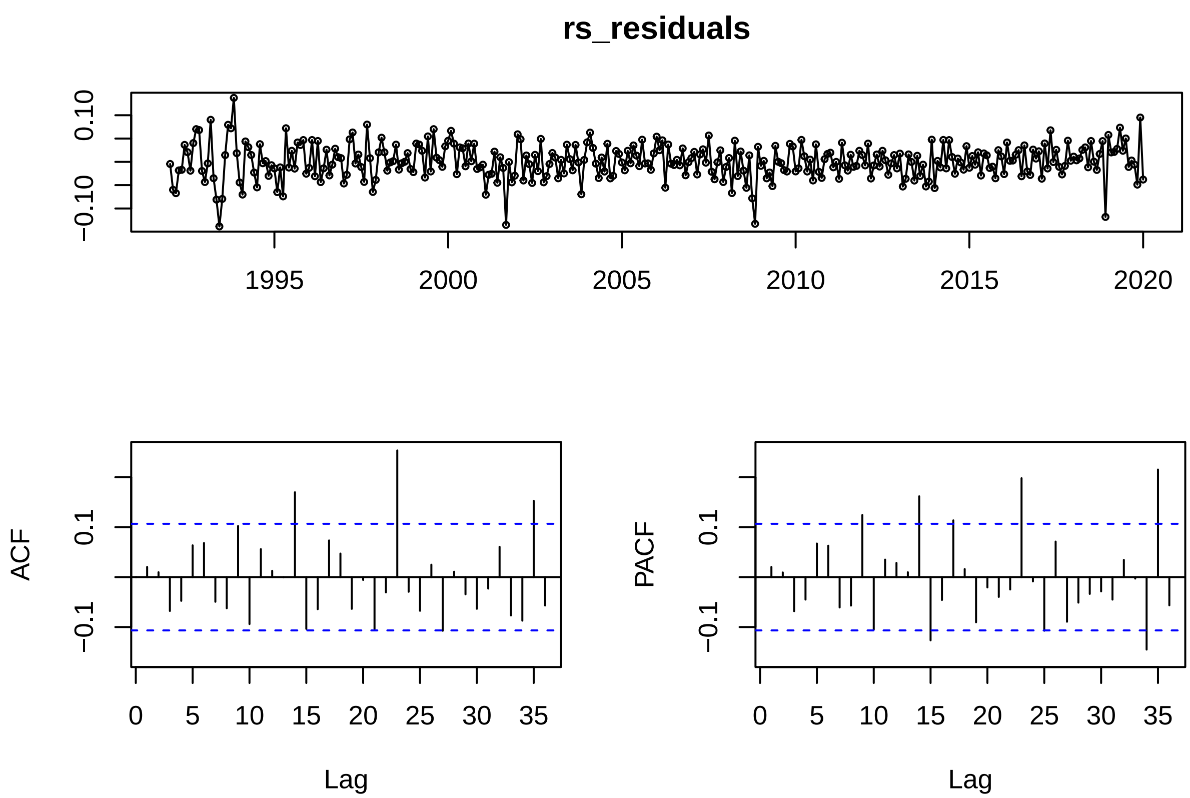


Our comprehensive model seem to capture the variations in the data well since it now takes into account the trend, seasonality, and cycles of the data. However, we note that the model does not seem to be a very good fit for the final few period of the data since it does not follow the data well, especially after 2018.

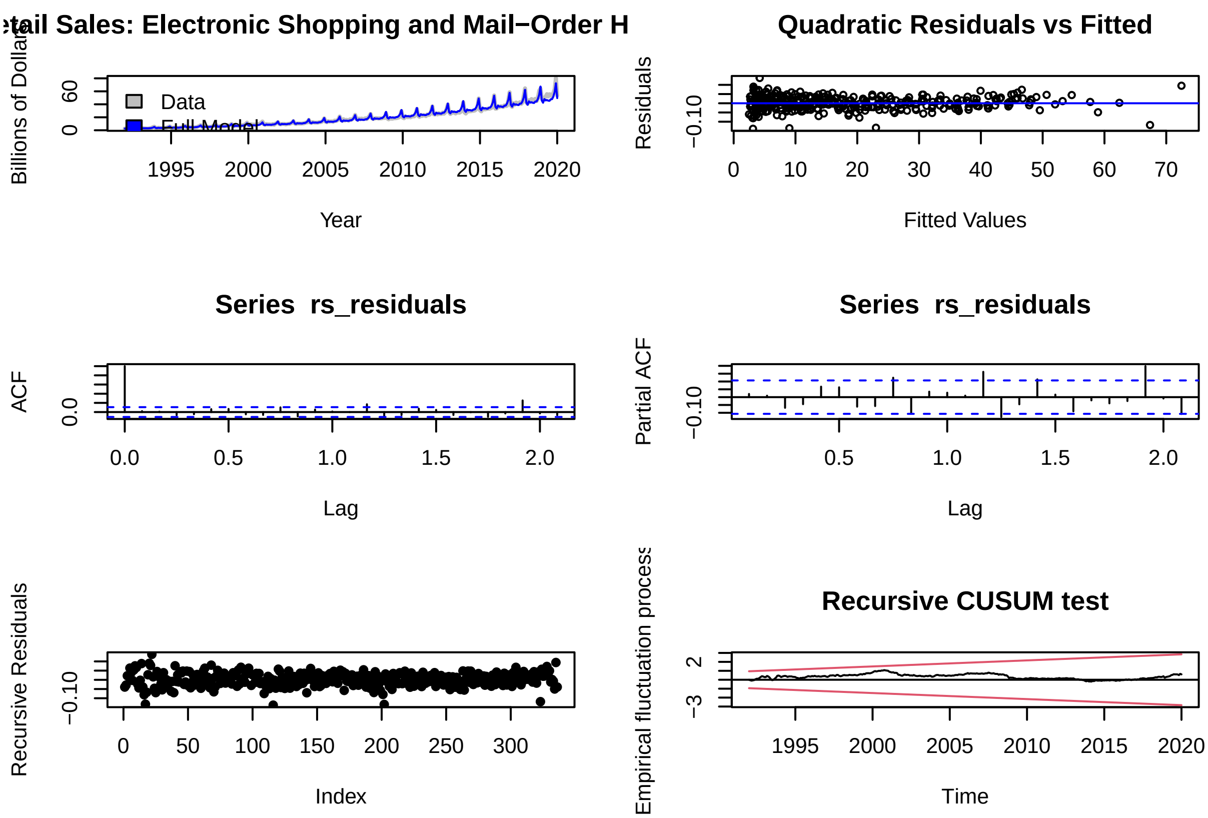
```
# Residual analysis

# Check residuals
rs_fitted <- rs_model$fitted.values + rs_res_model$fitted
rs_residuals <- rs_model$residuals - rs_res_model$fitted
checkresiduals(rs_residuals)
```




```
tsdisplay(rs_residuals)
```




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

# Plot 1: Plot data against our full model
plot(rs_train, main = "Retail Sales: Electronic Shopping and Mail-Order Houses",
     xlab = "Year", ylab = "Billions of Dollars", col = "grey", lwd = 2)
lines(fitted(rs_model) + fitted(rs_res_model), col="blue")
legend("topleft", legend = c("Data", "Full Model"), fill = c("grey","blue"),
       cex = 1, bty = 'n')

# Plot 2: Residuals vs fitted
plot(x = rs_fitted, y = rs_residuals, xlab = "Fitted Values", ylab = "Residuals",
     main = "Quadratic Residuals vs Fitted")
abline(h = 0, col = "blue")

# Plot 3 & 4: ACF and PACF
acf(rs_residuals)
pacf(rs_residuals)

# Plot 5 & 6: Recursive CUSUM
y = recresid(rs_residuals ~ 1)
plot(y, pch = 16, ylab = "Recursive Residuals")
plot(efp(rs_residuals~1, type = "Rec-CUSUM"))
```



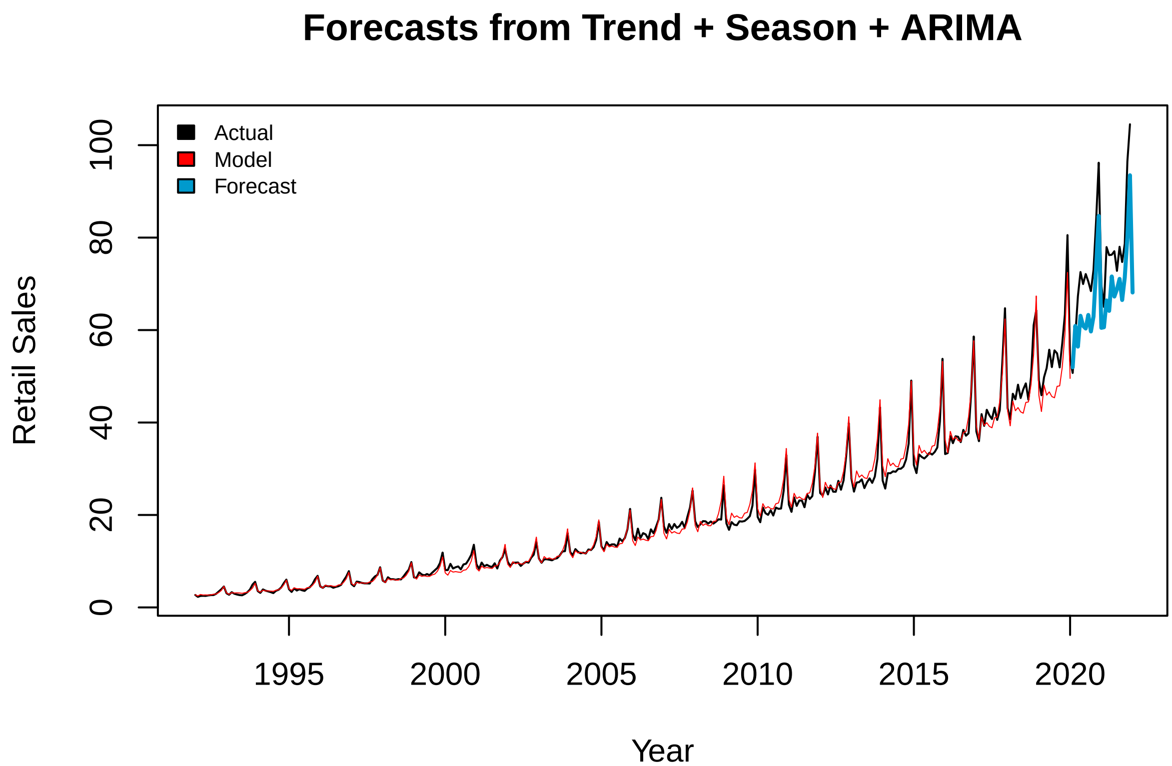
The ACF and PACF plots contains strong spikes at high lags. Furthermore, the spikes in the ACF and PACF are quite regular, indicating that our full model may not fit the data very well as there is still some dynamics that is not captured by our trend, seasonal, and cycles model.

We also note that the residuals are close to white noise.

The residuals vs fitted plot show that the residuals do not form a "horizontal band" around the y = 0 line. This suggests that the variance of the residuals is not constant, which we can confirm by looking at the plot of the residuals. There are no potential outliers present in the residuals, suggesting no extraordinary events.

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.

```r
# Full model
custom_trend <- forecast(tslm(rs_train ~ trend + I(trend^2) + season), h = 24)
custom_res <- forecast(auto.arima(custom_trend$residuals), h = 24)
fit_custom <- custom_trend[["mean"]] + custom_res[["mean"]]

# Plot
plot(rs_ts, ylab = "Retail Sales", xlab = "Year", lwd = 1,
     main = "Forecasts from Trend + Season + ARIMA")
lines(rs_fitted, col = "red", lwd = 0.5)
lines(fit_custom, col = "deepskyblue3", lwd = 2)
legend("topleft", legend = c("Actual", "Model", "Forecast"),
       fill=c("black","red", "deepskyblue3"), cex = 0.7, bty = 'n')
```




```r
# Test set error statistics
accuracy(fit_custom, rs_test)
```

```
##                 ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
## Test set 8.312838 9.280831 8.616991 10.65562 11.20274 0.2013456  1.109726
```

We can see that the forecast for the model struggles to predict the data after 2020, which is expected.

We can see reasonable values for the test set accuracy values. The MAPE is 11.20274. However, we cannot come to any conclusions, since we do not have any other models to compare this to. We can fit several other models and compare their error statistics, to hopefully find a better model.

ARIMA

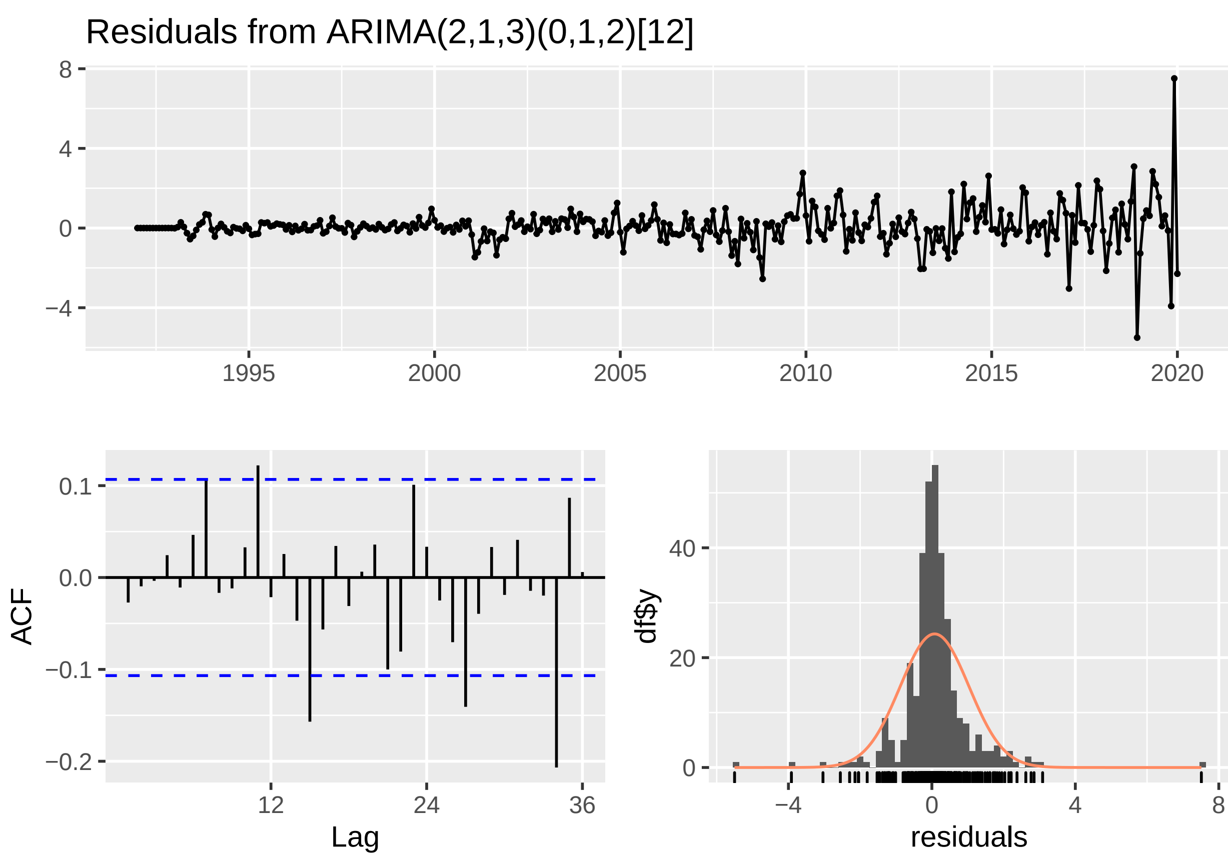
```r
arima_model <- auto.arima(rs_train)
summary(arima_model)
```

```
## Series: rs_train
## ARIMA(2,1,3)(0,1,2)[12]
##
## Coefficients:
##           ar1      ar2     ma1      ma2      ma3     sma1    sma2
##       -1.1137  -0.8237  0.1346  -0.0394  -0.2656  -0.2469  0.2253
```

```
## s.e.    0.0776   0.0906  0.1182   0.0823   0.1255   0.0710  0.0638
##
## sigma^2 = 0.9819:  log likelihood = -455.01
## AIC=926.02   AICc=926.48   BIC=956.27
##
## Training set error measures:
##                      ME      RMSE       MAE        MPE     MAPE      MASE
## Training set 0.07577676 0.9610671 0.5658549 0.09417335 3.124916 0.2794125
##                     ACF1
## Training set -0.02720459
```

R suggests ARIMA(2,1,3)(0,1,2)[12] as the best model for the Retail Sales: Electronic Shopping and Mail-Order Houses data. The I(1) component takes care of the trend since it is integrated with order 1. We can see a substantial ARMA model on the differenced data, with order ARMA(2,3). The seasonal component is captured by an S-ARMA(0,2) model on the seasonally-differenced data. The seasonal difference is likely due to the increasing amplitude in the seasonal variation that we identified in our decomposition earlier.

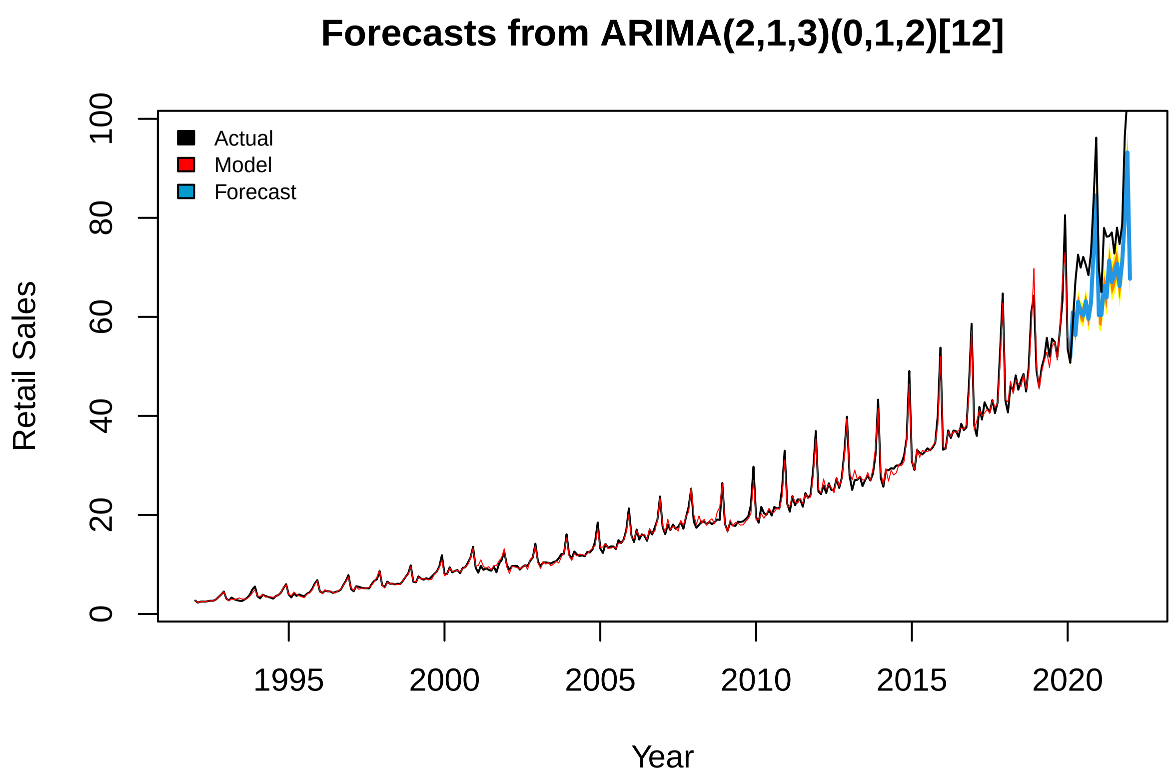
```
checkresiduals(arima_model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,3)(0,1,2)[12]
## Q* = 33.345, df = 17, p-value = 0.01019
##
## Model df: 7.   Total lags used: 24
```

The residuals show an interesting plot, with variance that increases with time. They do not appear to be normally distributed, as the histogram and Ljung-Box test indicate. However, the ACF does show relatively weak correlations, especially at low lags.

```
fit_arima <- forecast(arima_model, h = 24)
# Plot
plot(fit_arima, shadecols="oldstyle", ylab="Retail Sales", xlab="Year", xlim=c(1992, 2022))
lines(rs_test, col="black")
lines(fit_arima$fitted, col = "red", lwd = 0.5)
legend("topleft", legend = c("Actual", "Model", "Forecast"),
       fill=c("black","red", "deepskyblue3"), cex = 0.7, bty = 'n')
```




```
# Test set error statistics
accuracy(fit_arima, rs_test)
```

```
##                      ME      RMSE       MAE         MPE      MAPE      MASE
## Training set 0.07577676 0.9610671 0.5658549  0.09417335  3.124916 0.2794125
## Test set     8.47339176 9.4405412 8.7743122 10.86118017 11.401427 4.3326513
##                     ACF1 Theil's U
## Training set -0.02720459        NA
## Test set      0.20631122  1.127652
```

The ARIMA model suffers from much the same issue as the quadratic trend. Despite the sudden jump in 2020, the ARIMA model still tracks the variation in the data, albeit at a constant underestimate. Its accuracy metrics are quite similar to the quadratic model, and it performs roughly equivalent.

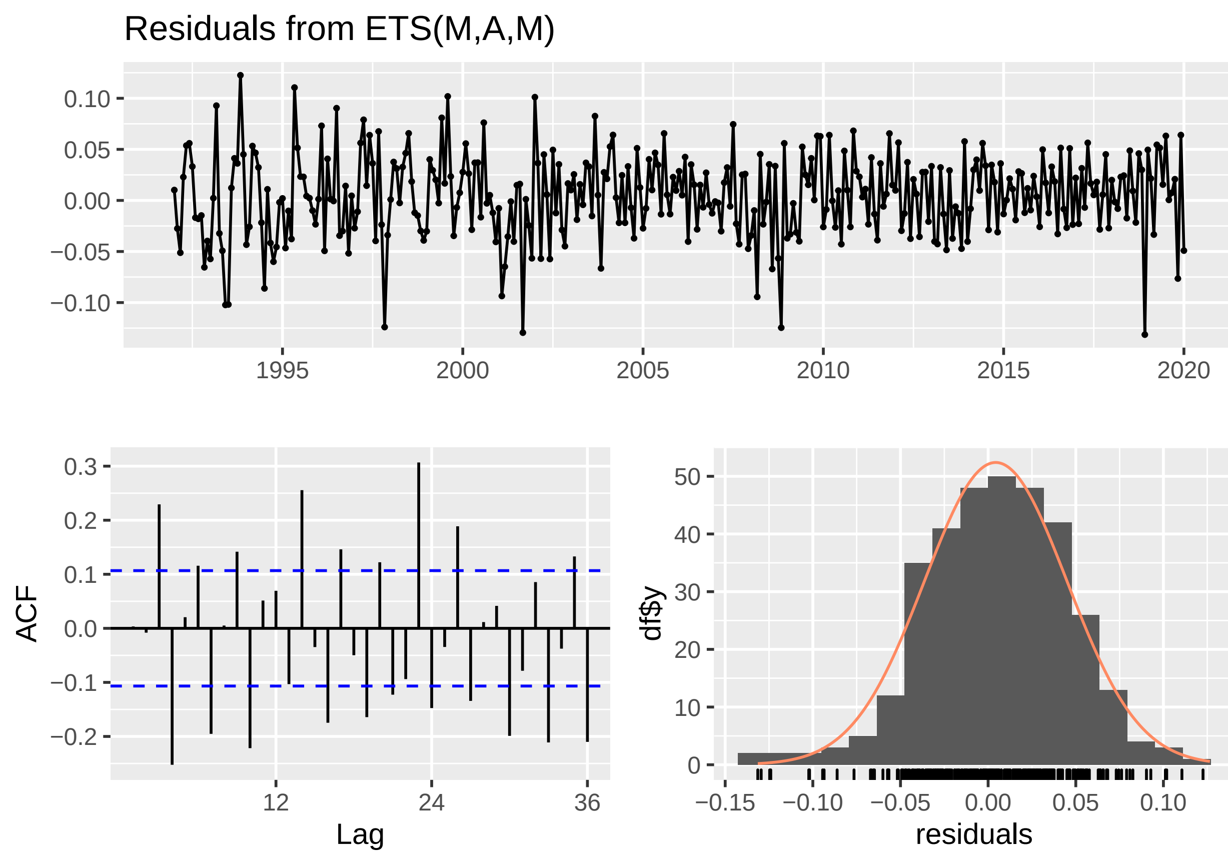
ETS

```
ets_model <- ets(rs_train)
summary(ets_model)
```

```
## ETS(M,A,M)
##
## Call:
##  ets(y = rs_train)
##
##   Smoothing parameters:
##     alpha = 0.2932
##     beta  = 0.0103
##     gamma = 0.5531
##
##   Initial states:
##     l = 2.595
##     b = 0.0509
##     s = 1.4787 1.3109 1.1026 0.955 0.8803 0.8583
##            0.8483 0.8387 0.8937 0.9577 0.8686 1.0072
##
##   sigma:  0.0419
##
##      AIC     AICc      BIC
## 1612.811 1614.730 1677.753
##
## Training set error measures:
##                      ME     RMSE       MAE       MPE     MAPE      MASE
## Training set 0.08844617 1.025324 0.6024697 0.2634997 3.247284 0.2974924
##                    ACF1
## Training set -0.3221904
```

As expected, R selected an ETS model with a multiplicative seasonality (and error) component, and a non-damped trend. The training set metrics are relatively similar to those from the ARIMA model, but slightly worse.

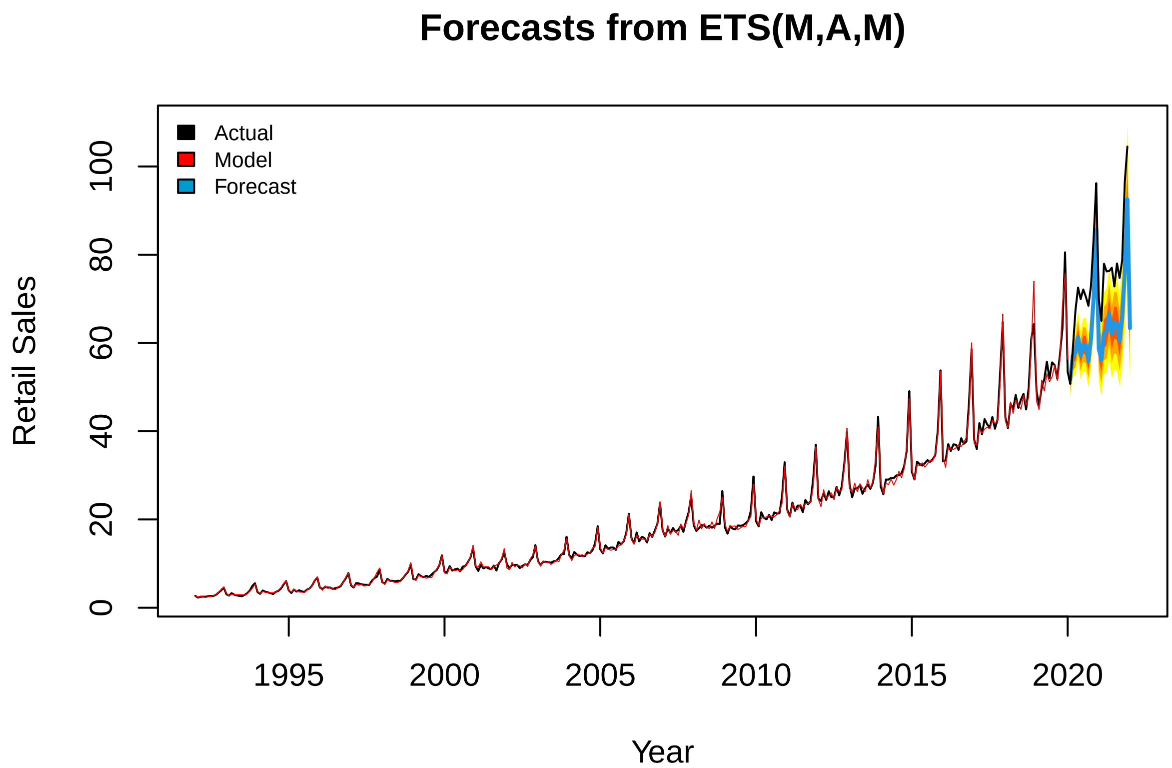
```
checkresiduals(ets_model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,A,M)
## Q* = 197.16, df = 8, p-value < 2.2e-16
##
## Model df: 16.   Total lags used: 24
```

While the plot of residuals from the ETS model looks good, and the histogram looks close to the theoretical distribution, the ACF reveals very strong serial correlations at many, many lags. There is clearly something that the ETS model is unable to capture in the data. The Ljung-Box test strongly rejects the null hypothesis, confirming that the residuals are not normally distributed.

```
fit_ets <- forecast(ets_model, level = c(50,80,95), h = 24)

# Plot
plot(fit_ets, shadecols="oldstyle", ylab = "Retail Sales", xlab = "Year", xlim = c(1992, 2022))
lines(fit_ets$fitted, col = "red", lwd = 0.5)
lines(rs_test, col="black")
legend("topleft", legend = c("Actual", "Model", "Forecast"),
       fill=c("black","red", "deepskyblue3"), cex = 0.7, bty = 'n')
```




```
# Test set error statistics
accuracy(fit_ets, rs_test)
```

```
##                       ME      RMSE        MAE        MPE      MAPE      MASE
## Training set  0.08844617  1.025324  0.6024697  0.2634997  3.247284 0.2974924
## Test set     11.37335722 12.154886 11.4599681 14.7777984 14.948575 5.6587964
##                    ACF1 Theil's U
## Training set -0.3221904        NA
## Test set      0.3291942  1.449226
```

The ETS model has much wider confidence intervals than the ARIMA model did. It appears to deeply underestimate the continued trend, resulting in worse test set accuracy statistics than our previous two models.

Holt-Winters

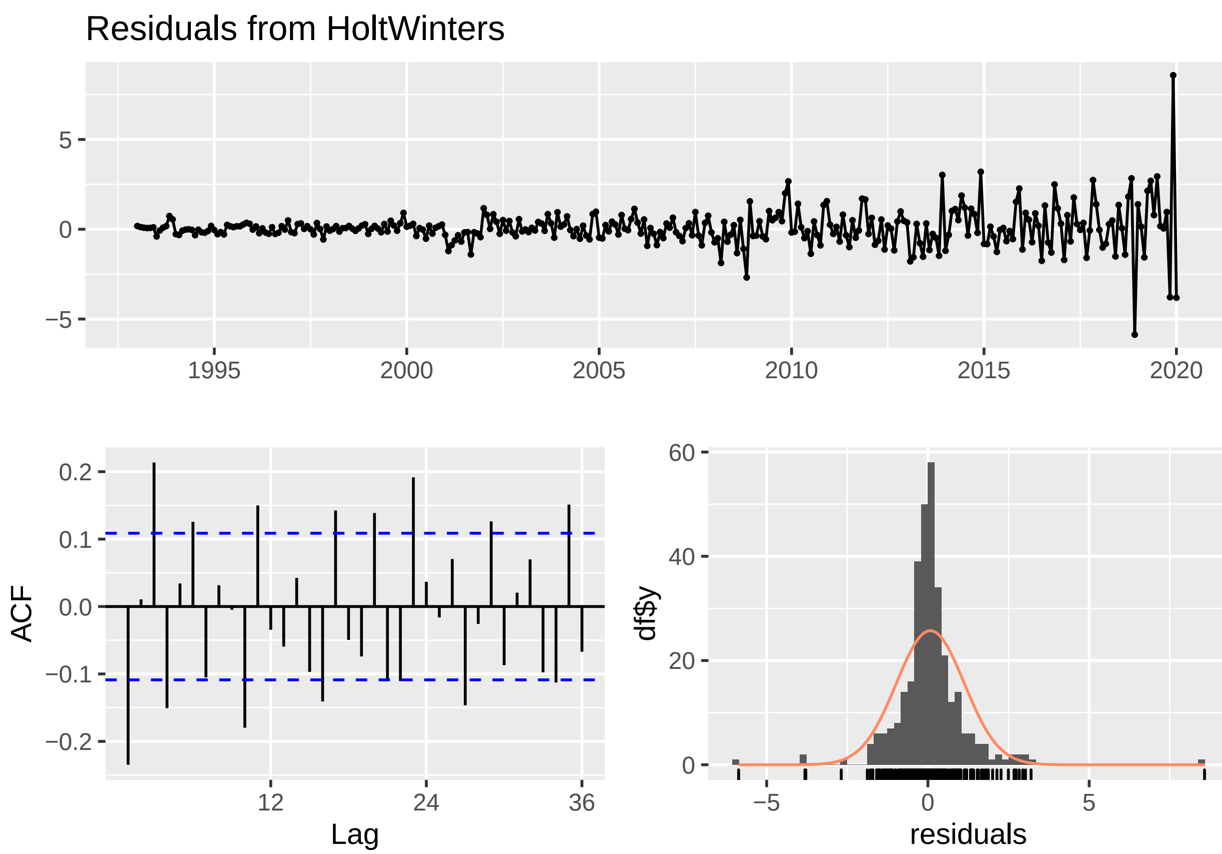
```
hw_model <- HoltWinters(rs_train, seasonal="additive")
hw_model
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
```

```
## HoltWinters(x = rs_train, seasonal = "additive")
##
## Smoothing parameters:
##  alpha: 0.1645026
##  beta : 0.1542361
##  gamma: 0.8289628
##
## Coefficients:
##            [,1]
## a   58.34968804
## b    0.64001822
## s1  -5.13428536
## s2  -1.13648105
## s3  -0.39825280
## s4   2.62362824
## s5  -1.49482041
## s6   0.82572814
## s7  -0.03401756
## s8  -3.66299580
## s9   0.74142189
## s10  7.57099172
## s11 21.07697905
## s12 -4.31968568
```

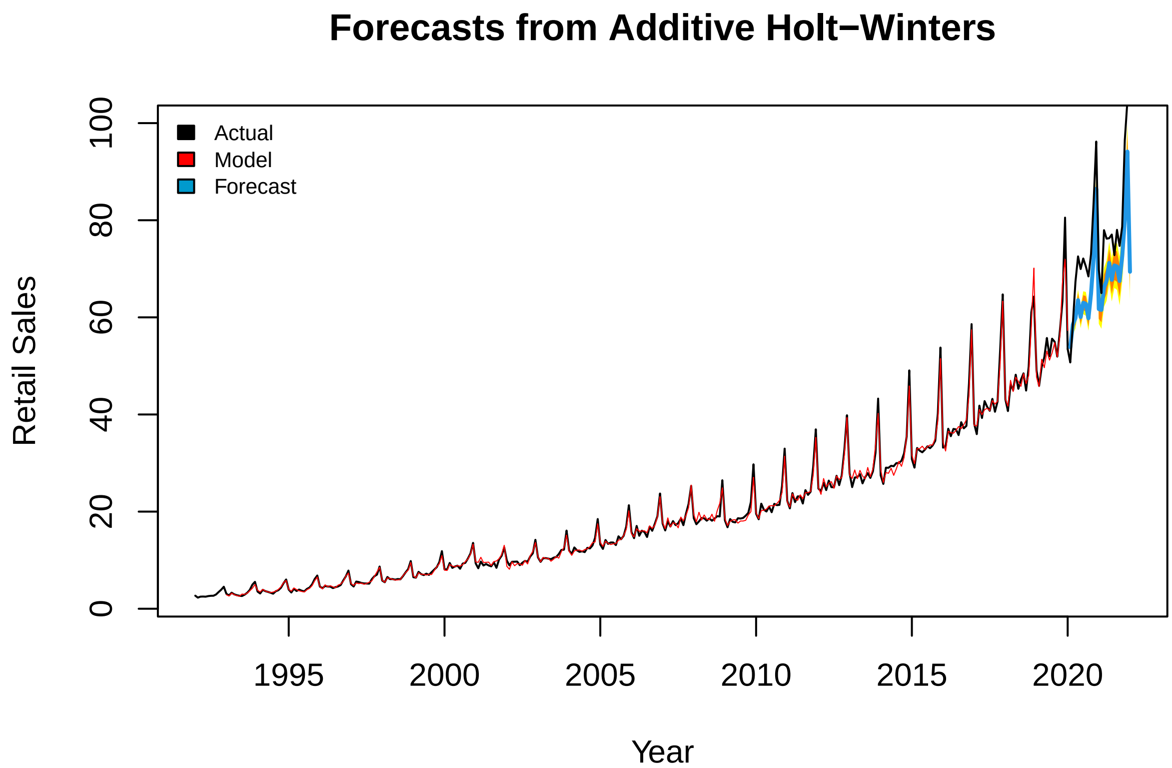
There is not very much to comment on for the Holt-Winters model, other than to say that we chose an additive seasonality because the multiplicative Holt-Winters method is almost equivalent to an ETS(M,A,M) model, which we already explored. As a result, we should expect the additive seasonality model to perform poorly, due to the factors we noticed during decomposition.

```
checkresiduals(hw_model)
```



We can see increasing variance of the residuals with time, likely due to the additive seasonality. We can also clearly see from the ACF plot and the histogram of residuals that it does not appear that the residuals are normally distributed.

```
fit_hw <- forecast(hw_model, h = 24)

# Plot
plot(fit_hw, shadecols="oldstyle", ylab="Retail Sales", xlab="Year", xlim=c(1992, 2022),
     main = "Forecasts from Additive Holt-Winters")
lines(fit_hw$fitted, col = "red", lwd = 0.5)
lines(rs_test, col="black")
legend("topleft", legend = c("Actual", "Model", "Forecast"),
       fill=c("black","red", "deepskyblue3"), cex = 0.7, bty = 'n')
```




```
# Test set error statistics
accuracy(fit_hw, rs_test)
```

```
##                       ME     RMSE       MAE        MPE      MAPE      MASE
## Training set 0.07400564 1.056381 0.6370406 0.07671928  3.351061 0.3145631
## Test set     7.58647997 8.571232 7.8594731 9.65643255 10.194711 3.8809146
##                     ACF1 Theil's U
## Training set -0.2346330        NA
## Test set      0.2223199  1.015902
```

Surprisingly, the Holt-Winters additive method performs well on the test set, with better accuracy measures than all of the previous models. The exponential smoothing method captures the variation very well, and the trend is closest to the actual data.

NNETAR

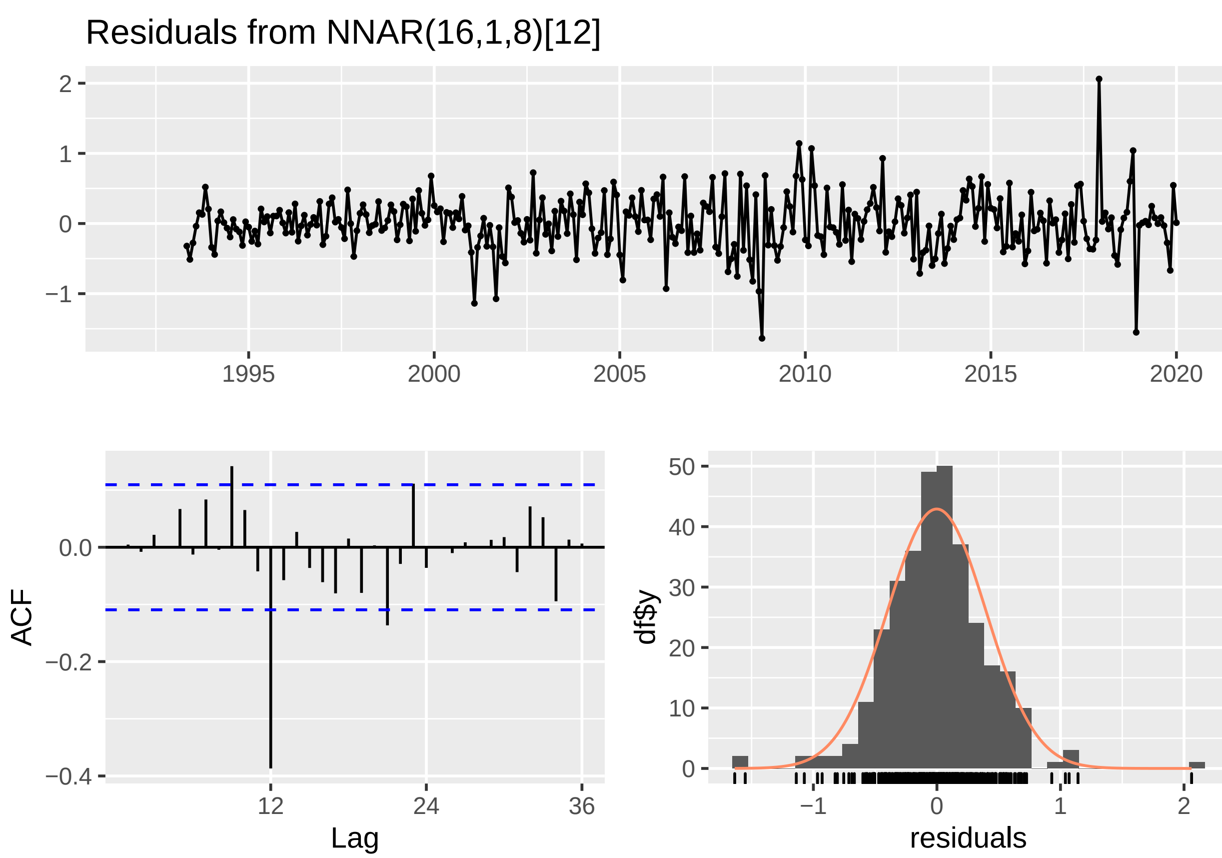
```
nnet_model <- nnetar(rs_train)
nnet_model
```

```
## Series: rs_train
## Model:  NNAR(16,1,8)[12]
## Call:   nnetar(y = rs_train)
```

```
##
## Average of 20 networks, each of which is
## a 16-8-1 network with 145 weights
## options were - linear output units
##
## sigma^2 estimated as 0.1591
```

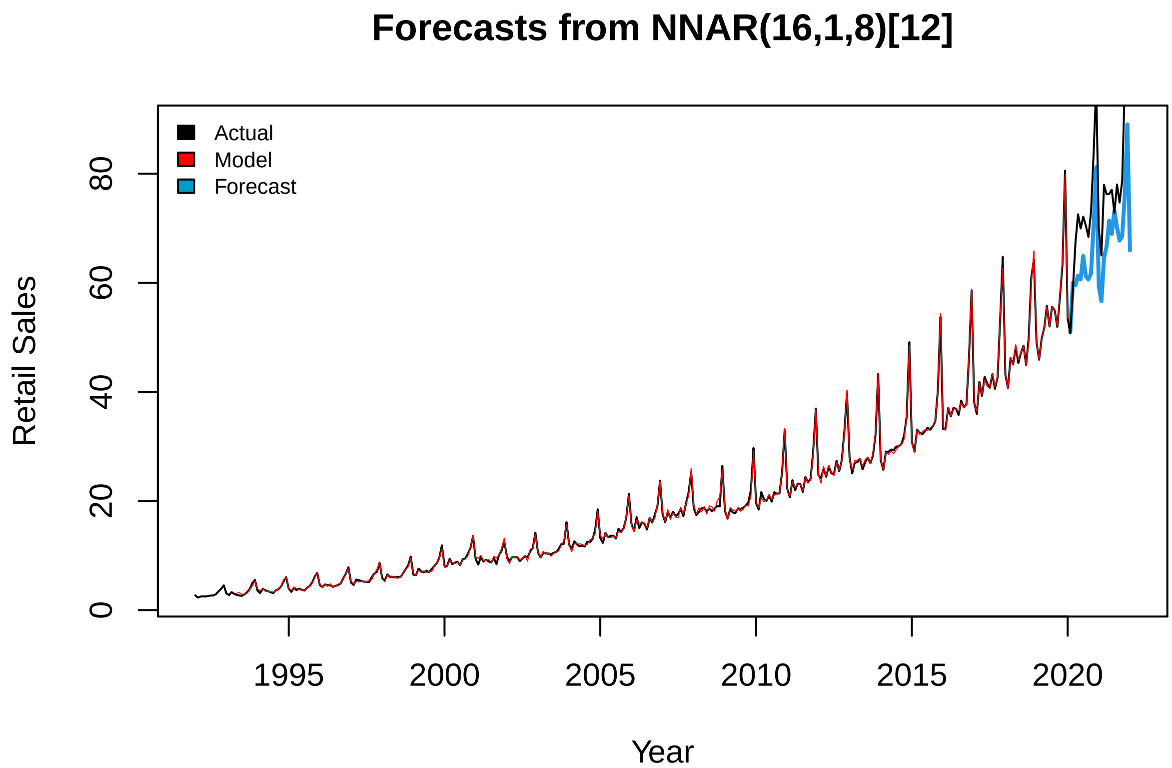
The NNETAR model selects a very large model of order 16, but still only a seasonal lag of 1. There is not much else we can tell from the model summary, as the nnet model is a "black box".

```
checkresiduals(nnet_model)
```



The residuals of the model are relatively small in scale, with low autocorrelation. Of particular note is the large negative correlation at lag 12, which indicates that the model is not capturing the seasonal variation as well as it could.

```
fit_nnet <- forecast(nnet_model, h = 24)

# Plot
plot(fit_nnet, ylab = "Retail Sales", xlab = "Year", xlim = c(1992, 2022))
lines(rs_test, col="black")
lines(fitted(fit_nnet), col = "red", lwd = 0.5)
legend("topleft", legend = c("Actual", "Model", "Forecast"),
       fill=c("black","red", "deepskyblue3"), cex = 0.7, bty = 'n')
```




```r
# Test set error statistics
accuracy(fit_nnet, rs_test)
```

```
##                         ME       RMSE       MAE        MPE      MAPE      MASE
## Training set -0.001795755  0.3988775 0.2952568 -0.1979438  2.228326 0.1457943
## Test set      8.943737924 10.2316713 9.1178769 11.3534068 11.644831 4.5022995
##                     ACF1 Theil's U
## Training set 0.004630946        NA
## Test set     0.442591301  1.197631
```

The model produces adequate forecasts, with a very good fit to the training set. The training set has better accuracy metrics (RMSE, MAPE, etc.) on the training set than any of the previous models, but it generalizes slightly worse. The MAPE on the test set is higher than the quadratic, ARIMA, and additive Holt-Winters models, though it does beat the ETS(M,A,M) model.

Of course, our nnet model does not have confidence interval bounds as it does not calculate a distribution for the forecasts.

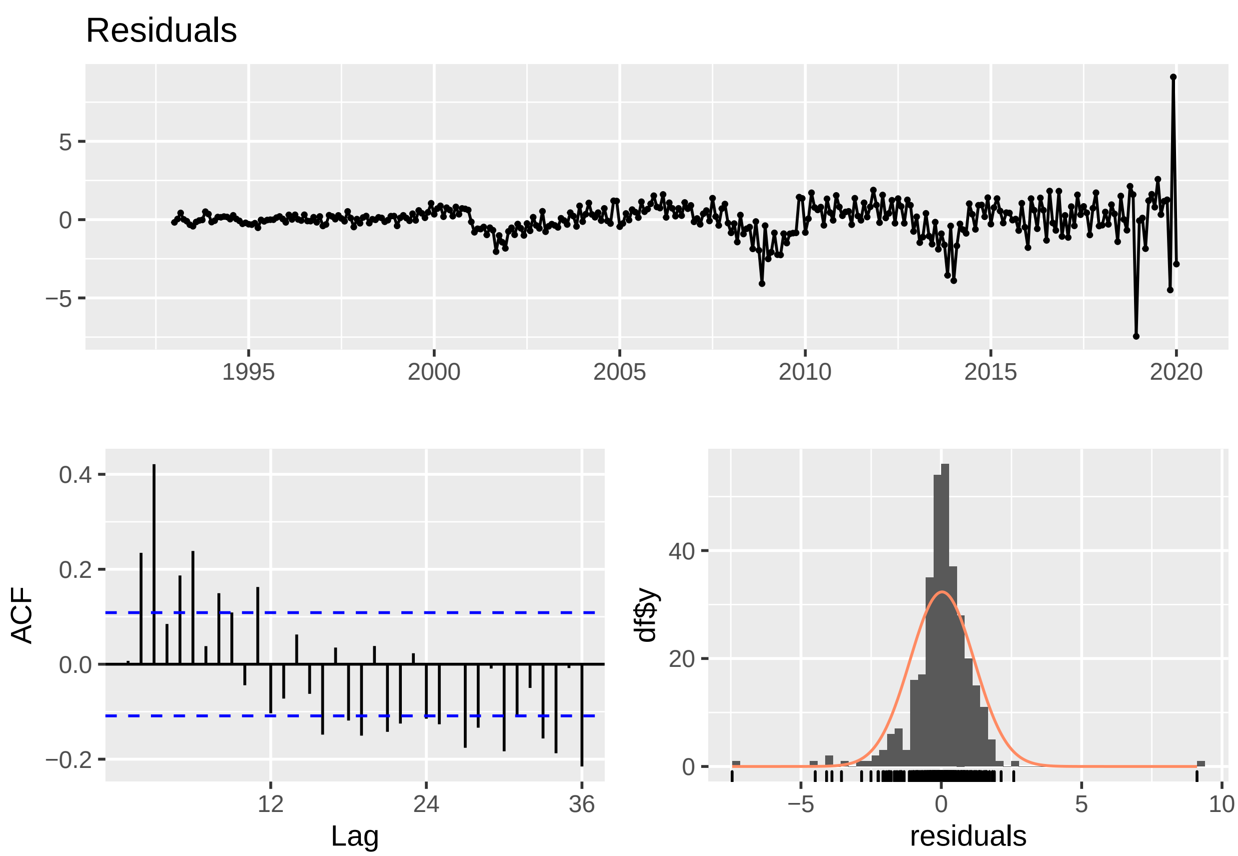
NNETAR with Transformation

```r
nnet_lambda_model <- nnetar(rs_train, lambda=0)
nnet_lambda_model
```

```
## Series: rs_train
## Model:  NNAR(1,1,2)[12]
## Call:   nnetar(y = rs_train, lambda = 0)
##
## Average of 20 networks, each of which is
## a 2-2-1 network with 9 weights
## options were - linear output units
##
## sigma^2 estimated as 0.002794
```

We also try transforming the data, to see if we can obtain a more parsimonious model than the previous nnetar model. We do obtain a much smaller model, with an order of only 1 and two seasonal autoregressive lags.

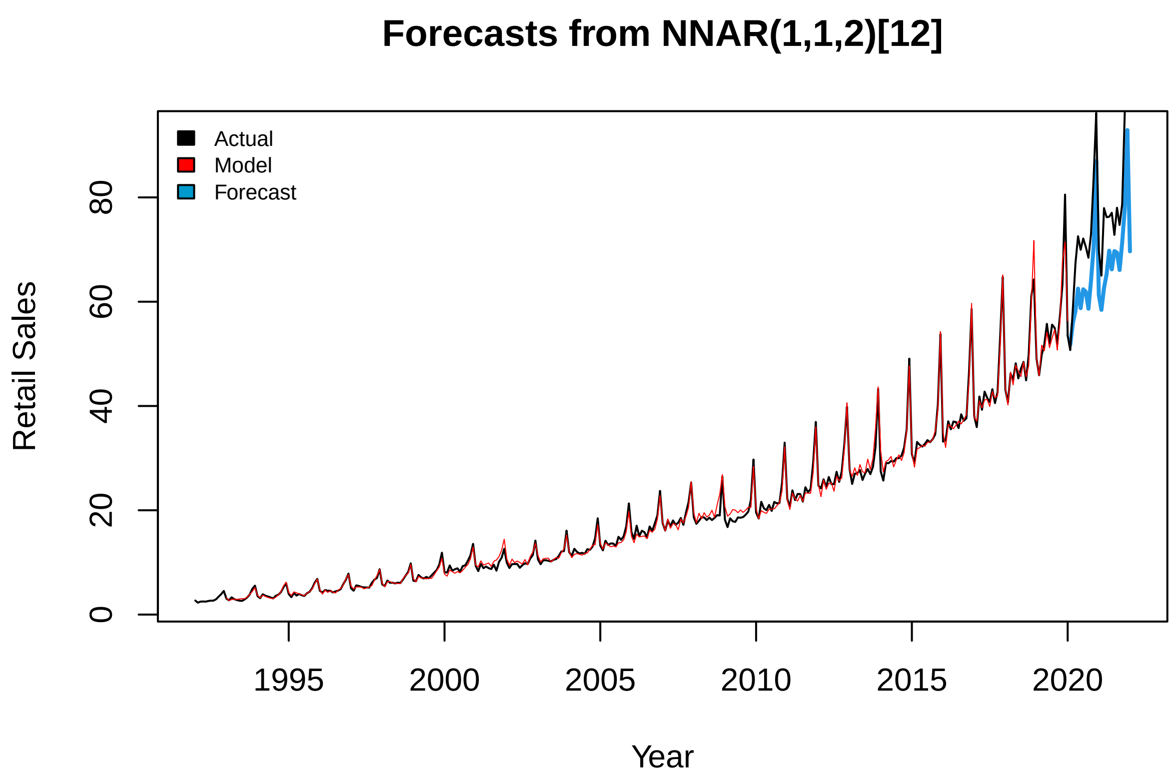
```
checkresiduals(nnet_lambda_model$residuals)
```



The residuals for the transformed model appear higher than for the original nnetar model. The scale of the residuals is larger (and increases with time), and there are many significant lags in the ACF.

```
fit_lambda_nnet <- forecast(nnet_lambda_model, h = 24)

# Plot
plot(fit_lambda_nnet, ylab = "Retail Sales", xlab = "Year", xlim = c(1992, 2022))
lines(rs_test, col="black")
lines(fitted(fit_lambda_nnet), col = "red", lwd = 0.5)
```

```
legend("topleft", legend = c("Actual", "Model", "Forecast"),
       fill=c("black","red", "deepskyblue3"), cex = 0.7, bty = 'n')
```




```
# Test set error statistics
accuracy(fit_lambda_nnet, rs_test)
```

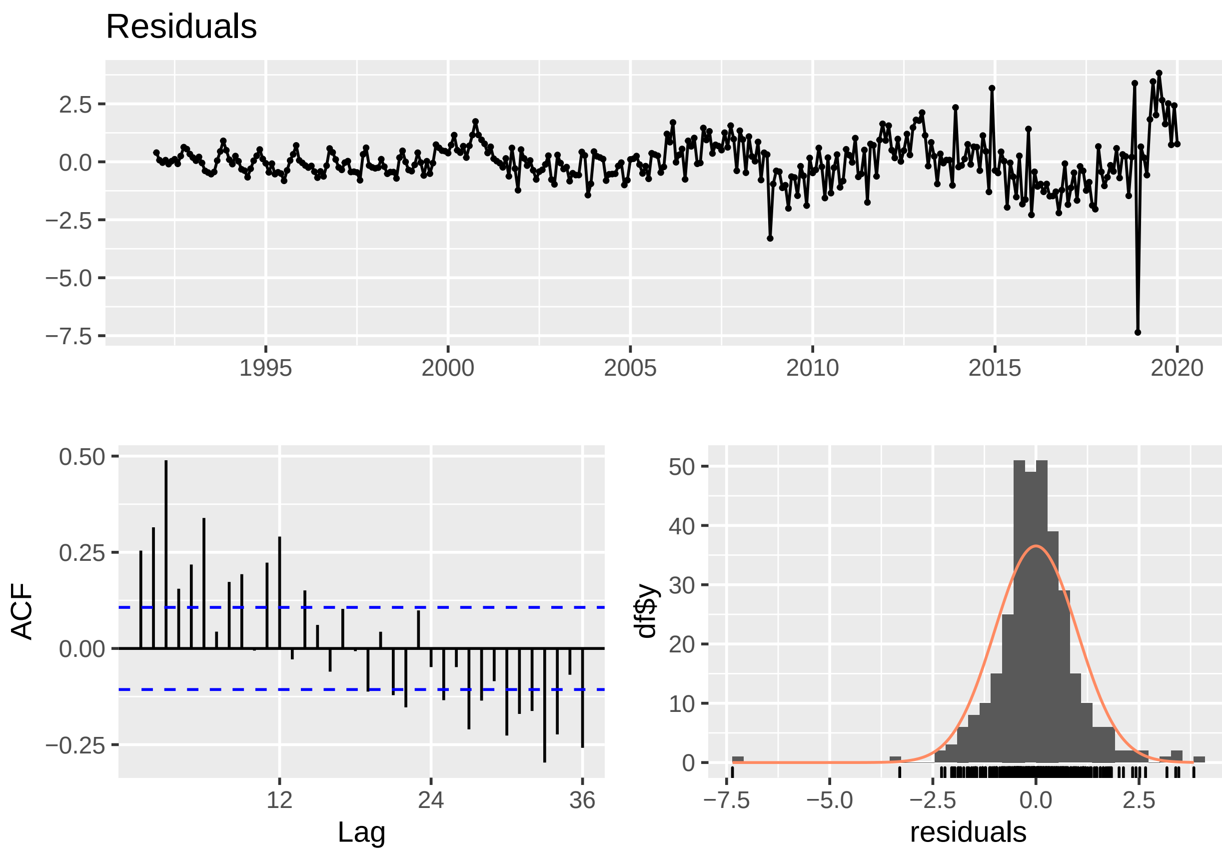
```
##                      ME     RMSE       MAE        MPE      MAPE      MASE
## Training set 0.02855593 1.146096 0.7145464 -0.1334426  4.051408 0.3528345
## Test set     9.05631714 9.882855 9.1446458 11.7096165 11.883780 4.5155177
##                    ACF1 Theil's U
## Training set 0.00707396        NA
## Test set     0.15238989  1.185973
```

While the transformed model does not fit the training set as well as the original model, it does generalize better, with MAPE comparable to our previous models and better than the original nnetar model.

Prophet

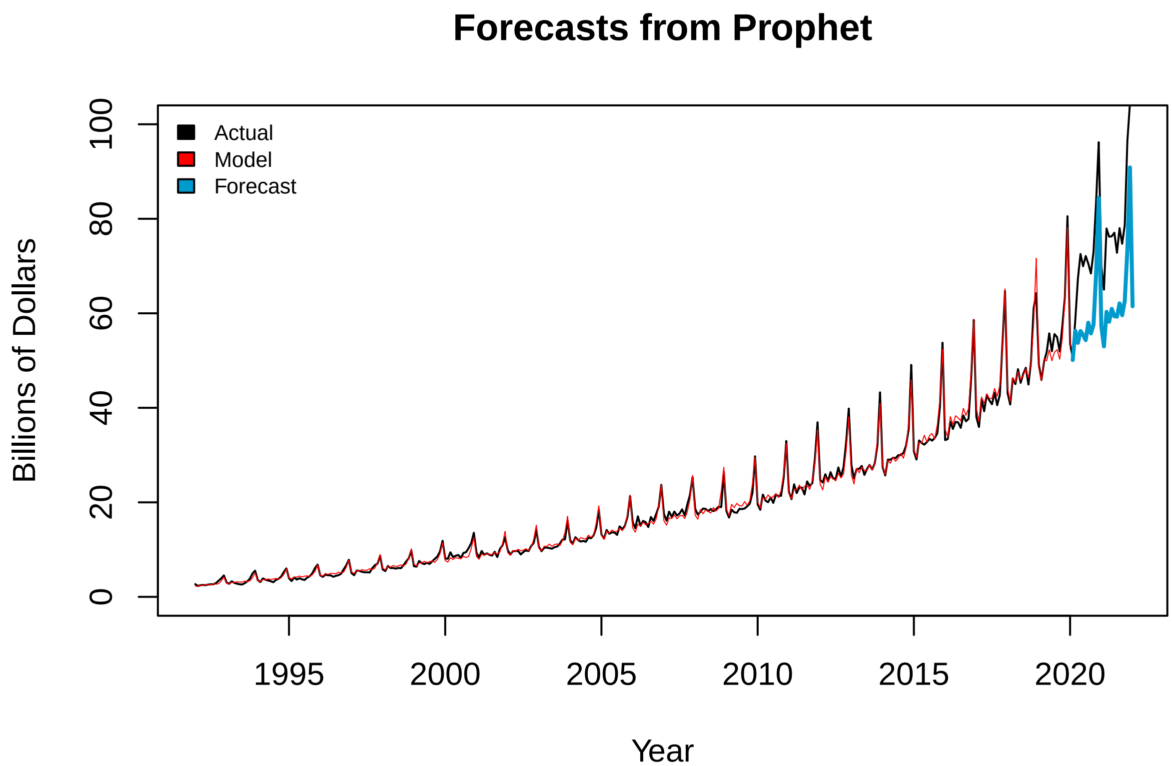
```
rs_train_tsibble <- as_tsibble(rs_train)
prophet_model <- rs_train_tsibble %>%
  model(prophet = prophet(value ~ season("year", type = "multiplicative")))
fit_prophet_fitted <- fitted(prophet_model)$.fitted
```

```r
fit_prophet_fitted <- ts(fit_prophet_fitted, start = c(1992, 1), freq = 12)
fit_prophet <- forecast(prophet_model, h = 24)
fit_prophet <- ts(fit_prophet$.mean, start = c(2020, 2), freq = 12)

checkresiduals(rs_train - fit_prophet_fitted)
```



The residuals appear somewhat cyclical, indicating that the model may not capture a multi-year cycle. The amplitude of the residuals increases with time, so the model may not capture the trend accurately. The ACF has many significant lags.

```r
# Plot
plot(rs_ts, ylab = "Billions of Dollars", xlab = "Year", ylim = c(0, 100), lwd = 1,
     main = "Forecasts from Prophet")
lines(fit_prophet_fitted, col = "red", lwd = 0.5)
lines(fit_prophet, col = "deepskyblue3", lwd = 2)
legend("topleft", legend = c("Actual", "Model", "Forecast"),
       fill=c("black","red", "deepskyblue3"), cex = 0.7, bty = 'n')
```




```
# Test set error statistics
accuracy(fit_prophet, rs_test)
```

```
##                 ME     RMSE      MAE     MPE    MAPE     ACF1 Theil's U
## Test set 14.05068 14.78702 14.05068 18.4015 18.4015 0.402842  1.774752
```

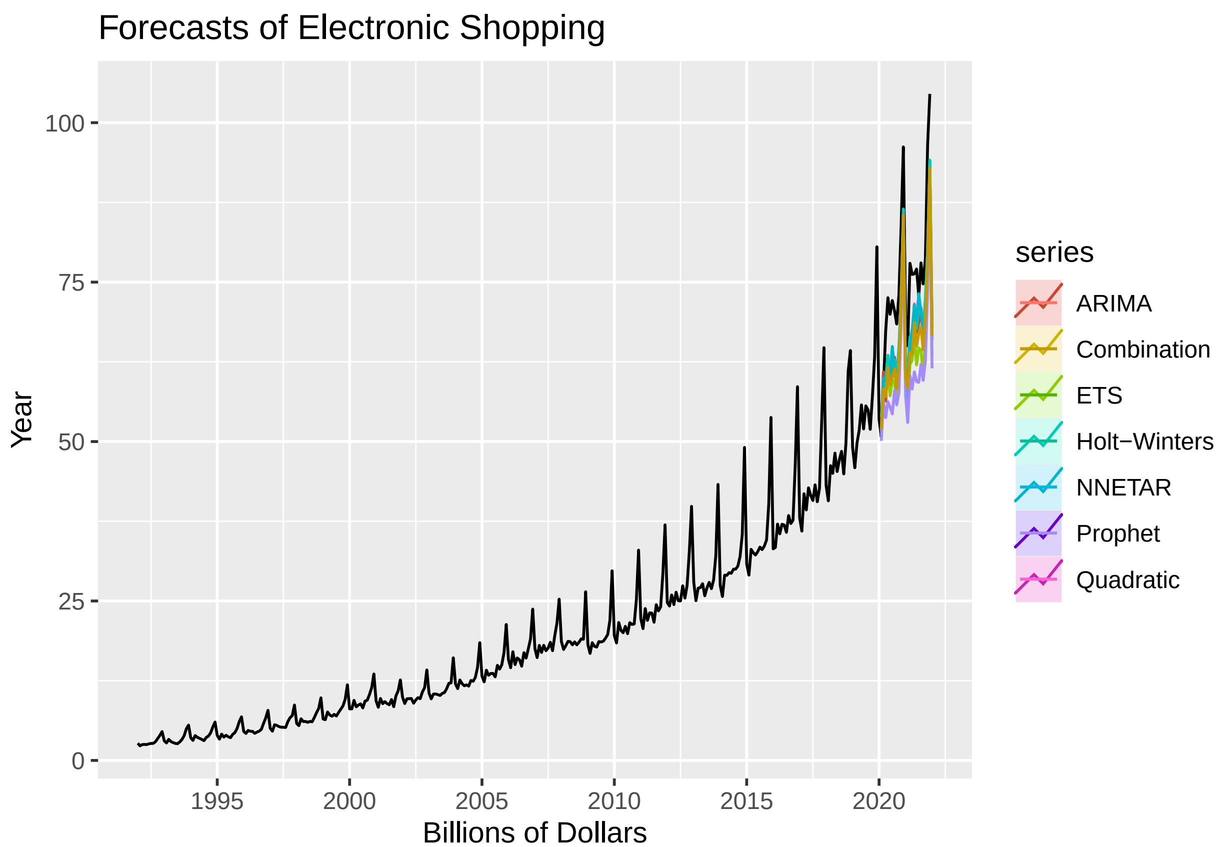
The prophet model performs the worst. The trend does not adjust fast enough to match the data, and as a result it underestimates the test set data. The test set accuracies are the worst of any of the models.

Forecast Combination

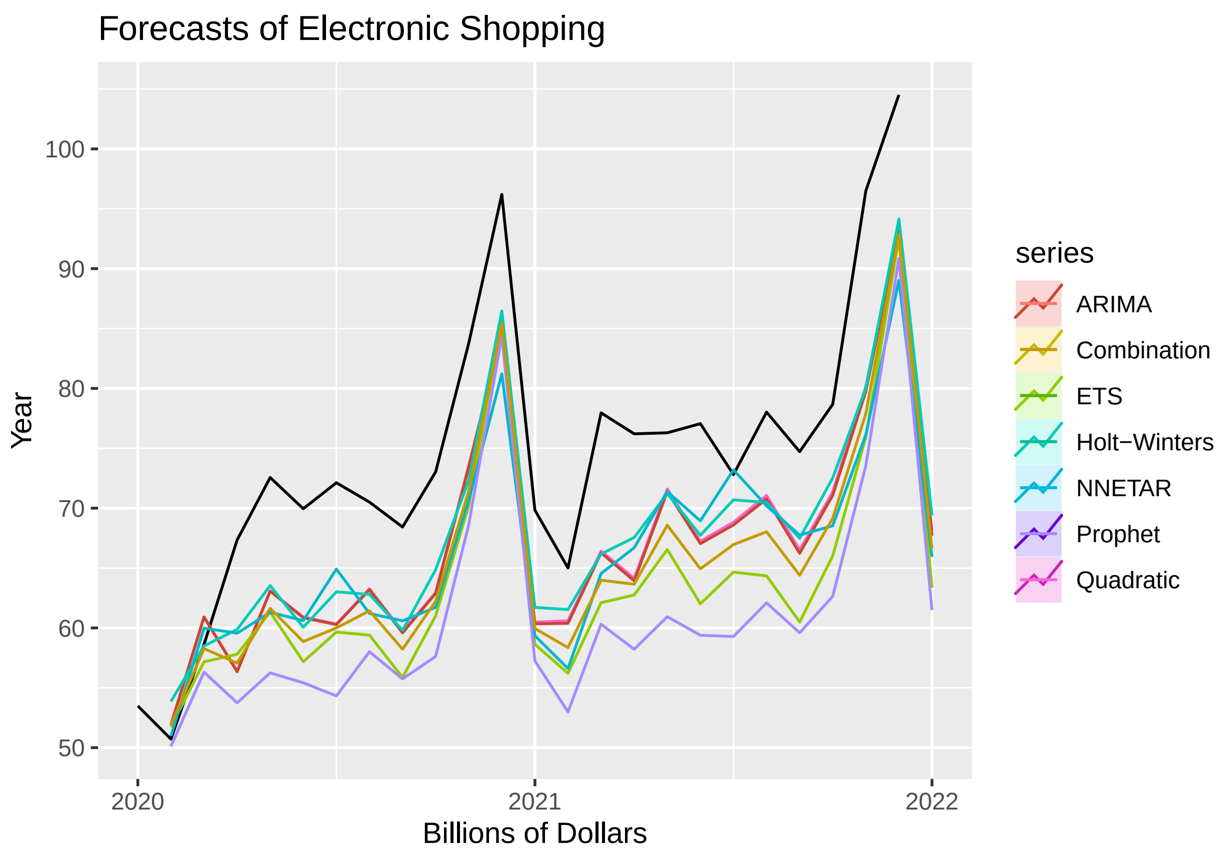
For our forecast combination, we choose to combine all of our previous models (using the transformed NNETAR). While the prophet model performs poorly compared to the other models, its fit on the training set data is still good, so we would not know its performance when making a forecast.

```
# Get the combination forecast
fit_combination <- (fit_custom + fit_arima[["mean"]] + fit_ets[["mean"]] +
                      fit_hw[["mean"]] + fit_lambda_nnet[["mean"]] + fit_prophet) / 6

# Plot all model forecasts together
autoplot(rs_ts) +
  autolayer(fit_custom, series="Quadratic") +
  autolayer(fit_arima, PI=FALSE, series="ARIMA") +
  autolayer(fit_ets, PI=FALSE, series="ETS") +
```

```
  autolayer(fit_hw, PI=FALSE, series="Holt-Winters") +
  autolayer(fit_nnet, PI=FALSE, series="NNETAR") +
  autolayer(fit_prophet, series="Prophet") +
  autolayer(fit_combination, series="Combination") +
  xlab("Billions of Dollars") +
  ylab("Year") +
  ggtitle("Forecasts of Electronic Shopping")
```




```
# Zoom in on forecast period
autoplot(rs_test) +
  autolayer(fit_custom, series="Quadratic") +
  autolayer(fit_arima, PI=FALSE, series="ARIMA") +
  autolayer(fit_ets, PI=FALSE, series="ETS") +
  autolayer(fit_hw, PI=FALSE, series="Holt-Winters") +
  autolayer(fit_nnet, PI=FALSE, series="NNETAR") +
  autolayer(fit_prophet, series="Prophet") +
  autolayer(fit_combination, series="Combination") +
  xlab("Billions of Dollars") +
  ylab("Year") +
  ggtitle("Forecasts of Electronic Shopping")
```




```r
# Get the MAPE for forecast
c(Custom = accuracy(fit_custom, rs_test)["Test set", "MAPE"],
  ARIMA = accuracy(fit_arima, rs_test)["Test set", "MAPE"],
  ETS = accuracy(fit_ets, rs_test)["Test set", "MAPE"],
  "Holt-Winters" = accuracy(fit_hw, rs_test)["Test set", "MAPE"],
  NNETAR = accuracy(fit_nnet, rs_test)["Test set", "MAPE"],
  Prophet = accuracy(fit_prophet, rs_test)["Test set", "MAPE"],
  Combination = accuracy(fit_combination, rs_test)["Test set", "MAPE"])
```

```
##       Custom        ARIMA          ETS Holt-Winters       NNETAR      Prophet
##     11.20274     11.40143     14.94857     10.19471     11.64483     18.40150
##  Combination
##     12.87513
```

The combination forecast performs about in the middle of the other forecasts, with neither the highest nor the lowest MAPE. The lowest is surprisingly from the additive Holt-Winters method (with ARIMA and our quadratic method close behind), with the highest coming from the prophet model.

III. Conclusions and Future Work

Conclusions:

These results are unsurprising, since all of the models we tested underestimated the data after 2020. We chose this dataset to see how the models would handle an unexpected change in the data, and it appears that exponential smoothing methods perform very well, as did our quadratic trend method. The other methods damped the trend too much, resulting in worse performance.

Apart from our custom model that we designed using the sequential model building process (as demonstrated by Professor Rojas in lecture), we used the following additional methods as tools of performance comparison and benchmarking:

1. S-ARIMA
2. ETS(M,A,M)
3. Additive Holt-Winters
4. NNETAR
5. NNETAR with log transformation
6. Prophet

To compare the forecasts produced by each of these methods, we plotted the forecasts from each method on a single plot. It is interesting to see where the forecasts from each method are the same and where they differ from each other.

We also created a combination forecasts by taking the average of the forecasts from all 6 models, to test combination forecasting methods.

Comparing the MAPE produced by each method, we found that additive Holt-Winters gives the best forecast performance, closely followed by our quadratic model and S-ARIMA.

Future Work

All of the models we tested underestimated online sales throughout 2020 and 2021, likely due to the sudden impact of the COVID-19 pandemic in the beginning of 2020, right at the start of the test set. Once more data is available beyond 2021, we believe it would be valuable to examine the lasting impacts of the pandemic to see whether online sales levels return to pre-pandemic levels, as predicted by the variety of models that we fitted. More data would also permit a comparison of how the different model types incorporate a sudden large change in 2020 in the training data to adjust their future forecasts.

IV. References

1. U.S. Census Bureau, Retail Sales: Electronic Shopping and Mail-Order Houses [MRTSSM4541USN], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/MRTSSM4541USN, March 10, 2022.

V. R Source Code.

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