SYS 5581 Project - Model Development

Nick Coronato

Version of 2021-03-31 | Due 2021-03-31

Contents:

SECTION I: The Data and Data-generating Process

SECTION II: Exploratory Data Analysis

SECTION III: Forecasting Model Development

#SECTION I: The Data and Data-generating Process.

Data was obtained from the BTS website: https://www.transtats.bts.gov/Fields.asp?gnoyr_VQ=FIM or https://www.transtats.bts.gov/DL_SelectFields.asp?gnoyr_VQ=FIM

This website allows the user to select key metrics of airline performance by market (domestic segment or international) and by year, along with several other filters. I extracted the raw data (2015-2020NOV) for the following features:

1. Departures Scheduled [integer]
2. Departures Performed [integer]
3. Payload (pounds gross weight) [integer]
4. Seats [integer]
5. Passengers [integer]
6. Distance (miles) [integer]
7. Air Time (minutes) [integer]
8. Unique Carrier (Airline identifier) [factor]
9. Origin Airport ID (five digit code) [factor]
10. Origin City Name [factor]
11. Origin State Name [factor]
12. Origin Country [factor]
13. Destination Airport ID (five digit code) [factor]
14. Destination City Name [factor]
15. Destination State Name [factor]
16. Aircraft Type [factor]
17. Year
18. Quarter
19. Month

The data I downloaded is structured as 1,031,938 rows of a .csv file, with headers that indicate each of the columns (features) identified above. Each row entry, or observation, is a flight segment. A domestic flight segment is defined as any route that terminated in the United States or its territories; it could have originated anywhere.

Most route segments were executed multiple times throughout the time period, as indicated by the "Departures Performed" column. If multiple departures of the same segment were performed, the data is already

captured in such a way that the other variables were updated; in other words, if two departures of a 100-passenger segment were conducted, then the corresponding "PASSENGERS" column indicates "200" for that segment. I did not have to multiply $(2 * 100)$ to manually calculate for repeat segments.

I processed the raw data to fit the requirements of a tidy time series set. This required an aggregation of metrics at a *monthly* frequency. For example, all of the United Airlines segments during January 2015 were summed to form new features: Total Passengers, Total Air Time, etc. The time period of interest was January 2015 to November 2020. This produced 71 monthly observations for each of the "Big 3" airlines, which is a total of 213 data points for each metric. A monthly interpretation of historical travel data seemed most relevant and stable, as it appropriately highlights the seasonal trends of the year without being too granular.

Organize the data into a *tidy* data frame.

Organize by taking out the non-useful variables. Make another new variable called RPM (revenue passenger miles), which is simply Passengers X Miles for each observation. Put the Index variable in the first column (eventually it will be 4dig year, 2dig month). Put the Key variables next (unique_carrier, Passengers, Distance, RPM). Then all the other (possibly) relevant variables.

Make a new variable called YRMO that concatenates Year and Month into a YR-MO format. Make it a time series column.

The table below is a sample of the data after initial processing.

```
## # A tsibble: 6 x 9 [1M]
## # Key:       UNIQUE_CARRIER [1]
## # Groups:    YEAR, MONTH @ YRMO [6]
##       YRMO  YEAR MONTH UNIQUE_CARRIER TotalPax TotalRPM TotalDistance TotalSeats
##      <mth> <int> <int> <fct>             <int>    <dbl>         <int>      <int>
## 1 2015 Jan  2015     1 UA              4847338   6.54e9       2446772    5956785
## 2 2015 Feb  2015     2 UA              4639362   6.08e9       2316983    5590926
## 3 2015 Mar  2015     3 UA              5823398   7.74e9       2474956    6831596
## 4 2015 Apr  2015     4 UA              5556445   7.43e9       2549360    6531908
## 5 2015 May  2015     5 UA              6048718   8.07e9       2585598    7051073
## 6 2015 Jun  2015     6 UA              6296079   8.62e9       2655756    7228780
## # ... with 1 more variable: TotalAIR_TIME <int>
```

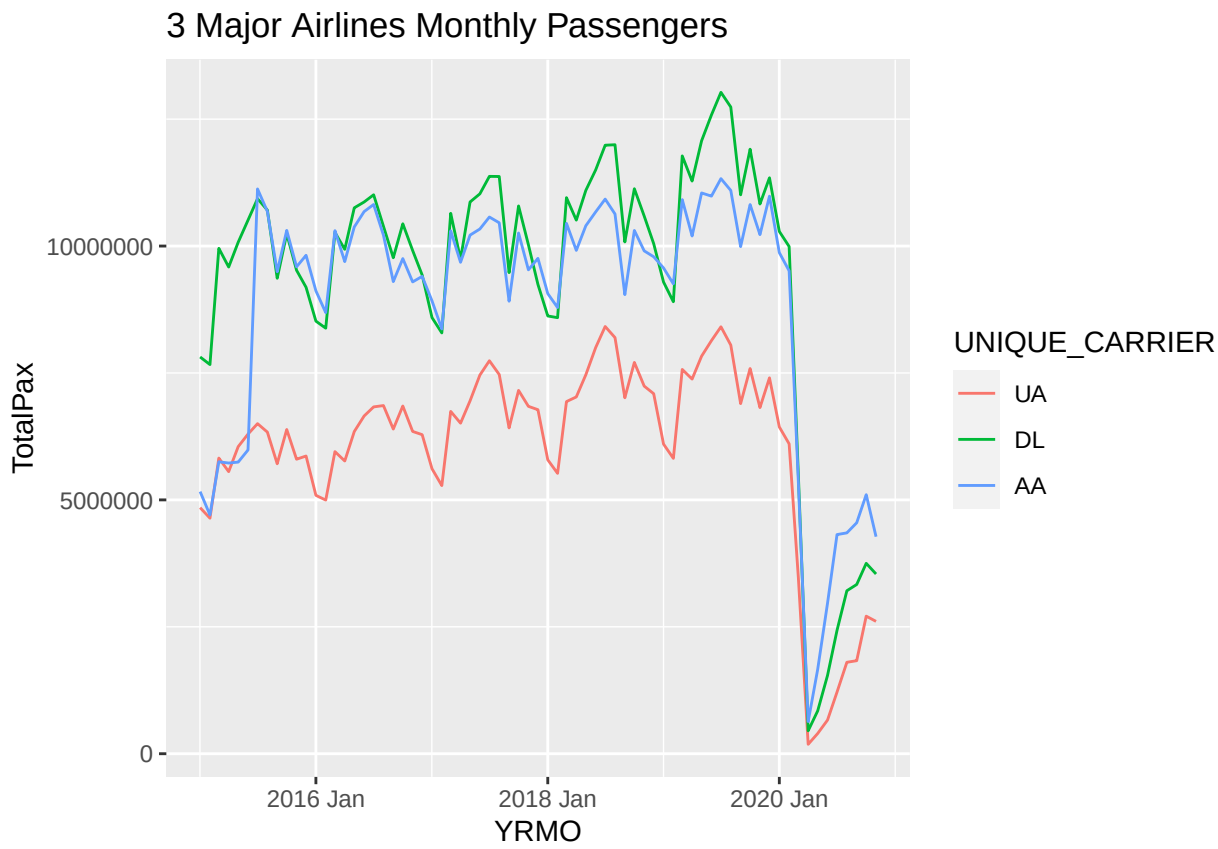
#SECTION II: Exploratory Data Analysis

Briefly characterize the data set.

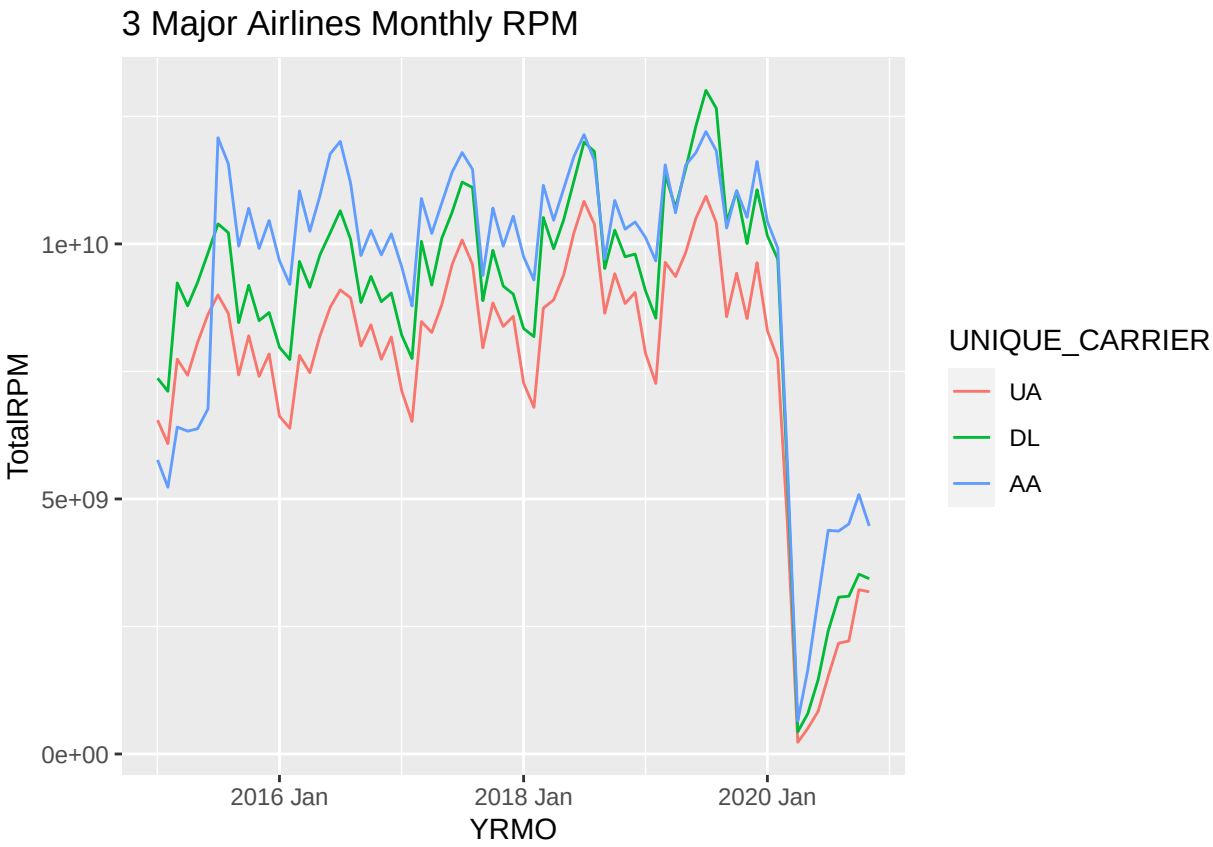
After extracting and transforming my data, I have a useful tibble with 204 observations. Each observation is a Monthly report of 9 variables. The time-series tibble called "big3" is the object I will be using for most analysis: it highlights the 3 U.S.-based airline giants that could give useful information about passenger trends in the domestic market.

I can view any of my trends for (Passengers, Seats, Miles Flown, Hours of Air Time) through ggplots like seen below.

```
#Show total passengers trend since 2015 Jan
ggplot(data = big3, mapping = aes(x = YRMO, y = TotalPax, color = UNIQUE_CARRIER)) + geom_line() + ggti
```



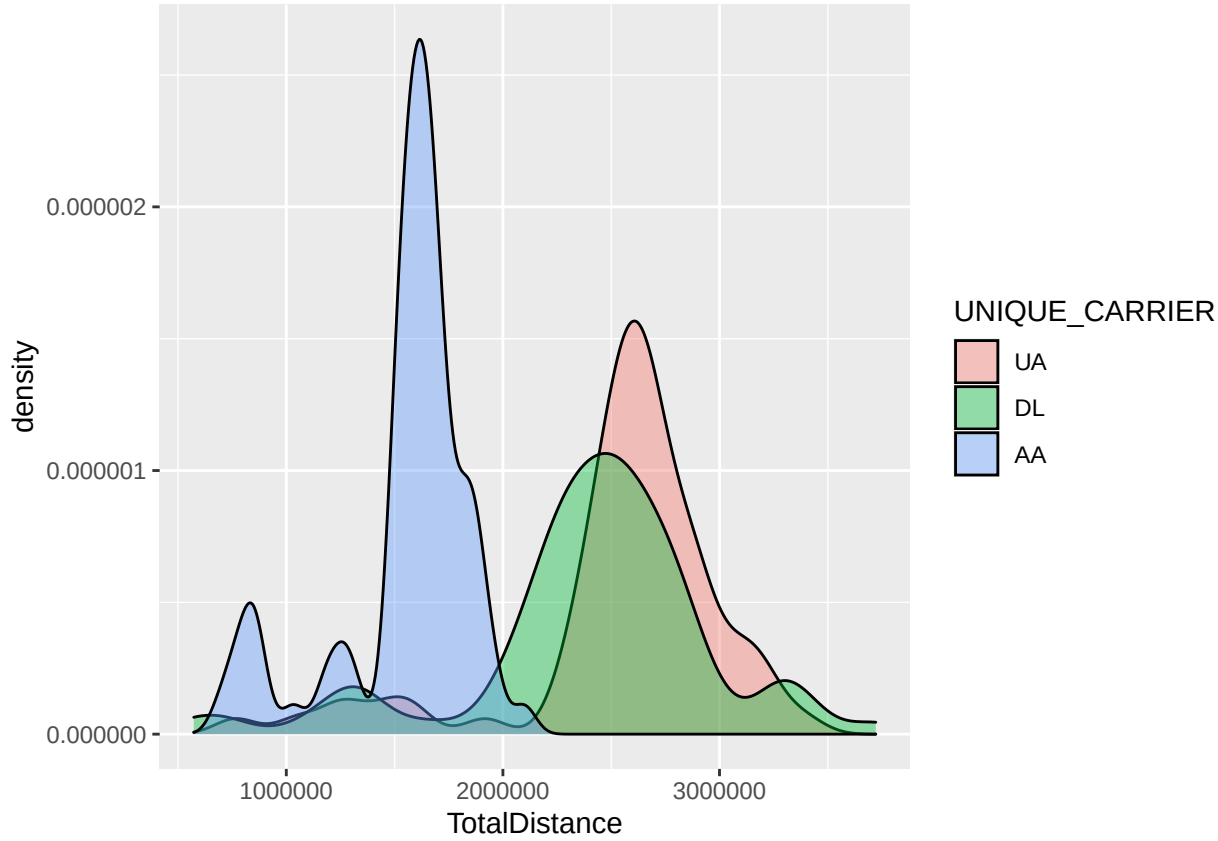

```
#Show total Revenue Passenger Miles (RPM) trend since 2015 Jan
ggplot(data = big3, mapping = aes(x = YRMO, y = TotalRPM, color = UNIQUE_CARRIER)) + geom_line() + ggti
```



Interesting note: United Airlines produces a consistently lower TotalPax count. The red line is a similar shape to DL and UA, but shifted down by about 5 million passengers per month. However, I noticed that they are near the same level of production of Revenue Passenger Miles. This makes me think that United Airlines moves way fewer passengers, but possibly over longer domestic routes, therefore producing a comparable RPM value every month.

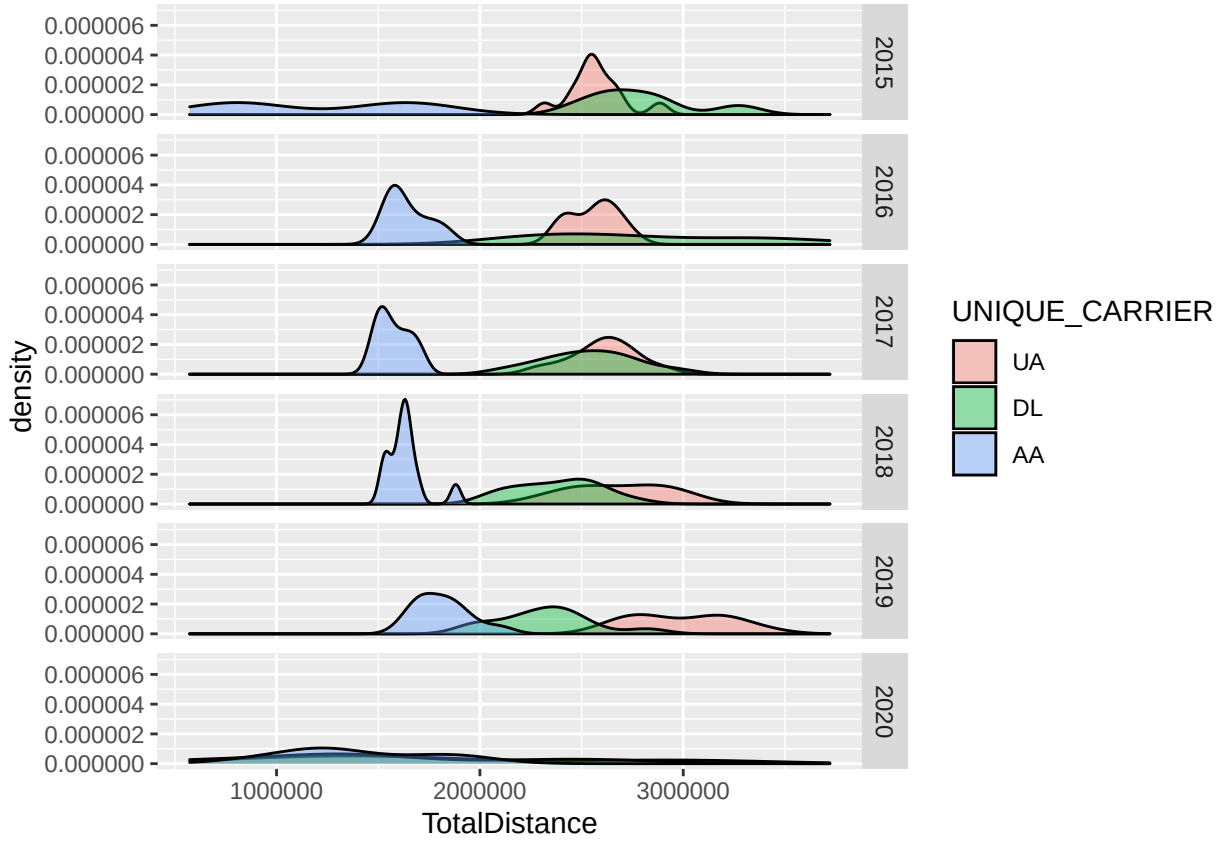
So I checked the distance flown statistics across the airlines.

```
#Display density plots of distance flown between the 3 carriers
ggplot(big3, aes(x = TotalDistance, fill = UNIQUE_CARRIER)) +
  geom_density(position = "identity", alpha = 0.4, color = "black")
```

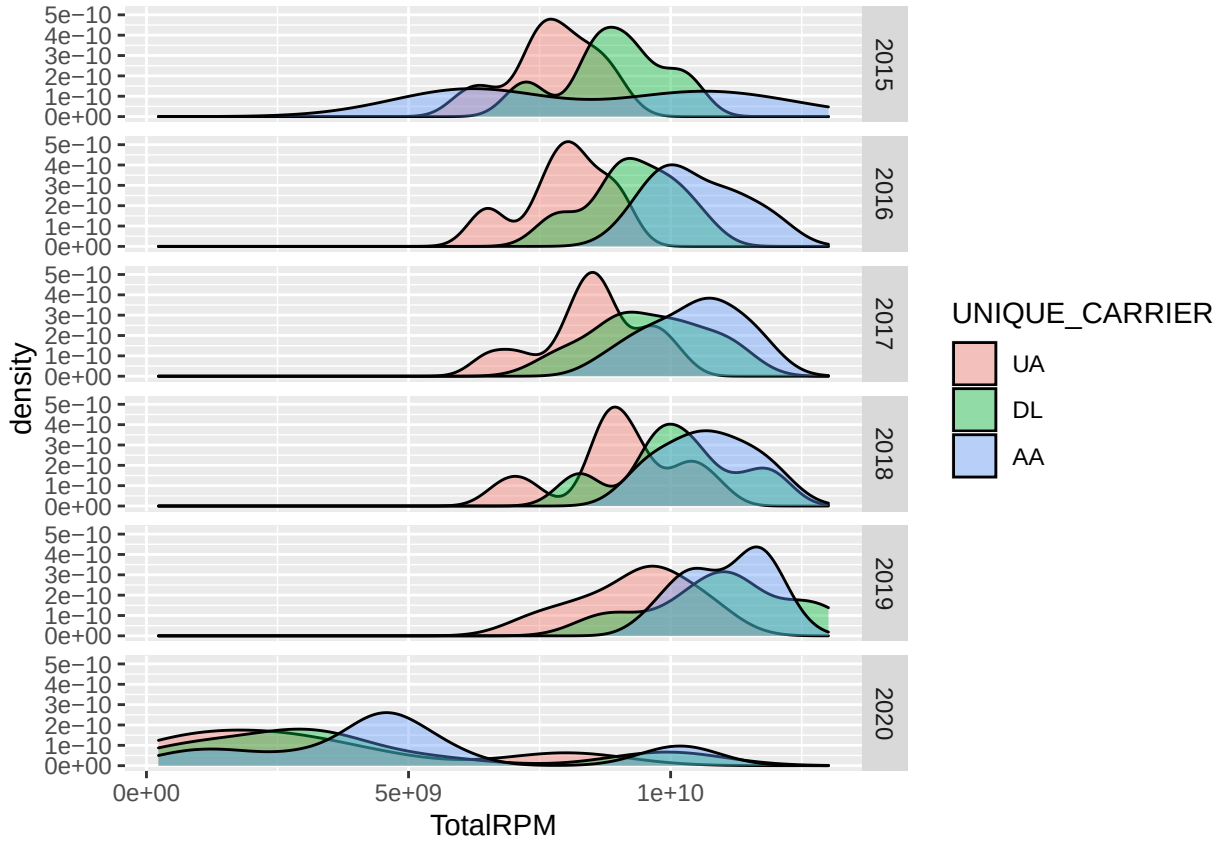


Yes, there seems to be evidence that the 3 airlines under investigation do have different strategies in terms of route lengths/distance flown in this data set. This could be a function of their actual corporate strategy, or just the nature of their "hub" locations, or even totally random. The chart below breaks it out by year and shows that United Airlines seems to fly longest distance routes (on average), American Airlines flies the shortest routes, and Delta's flight distance distribution is widely spread. Interesting stuff; maybe useful.

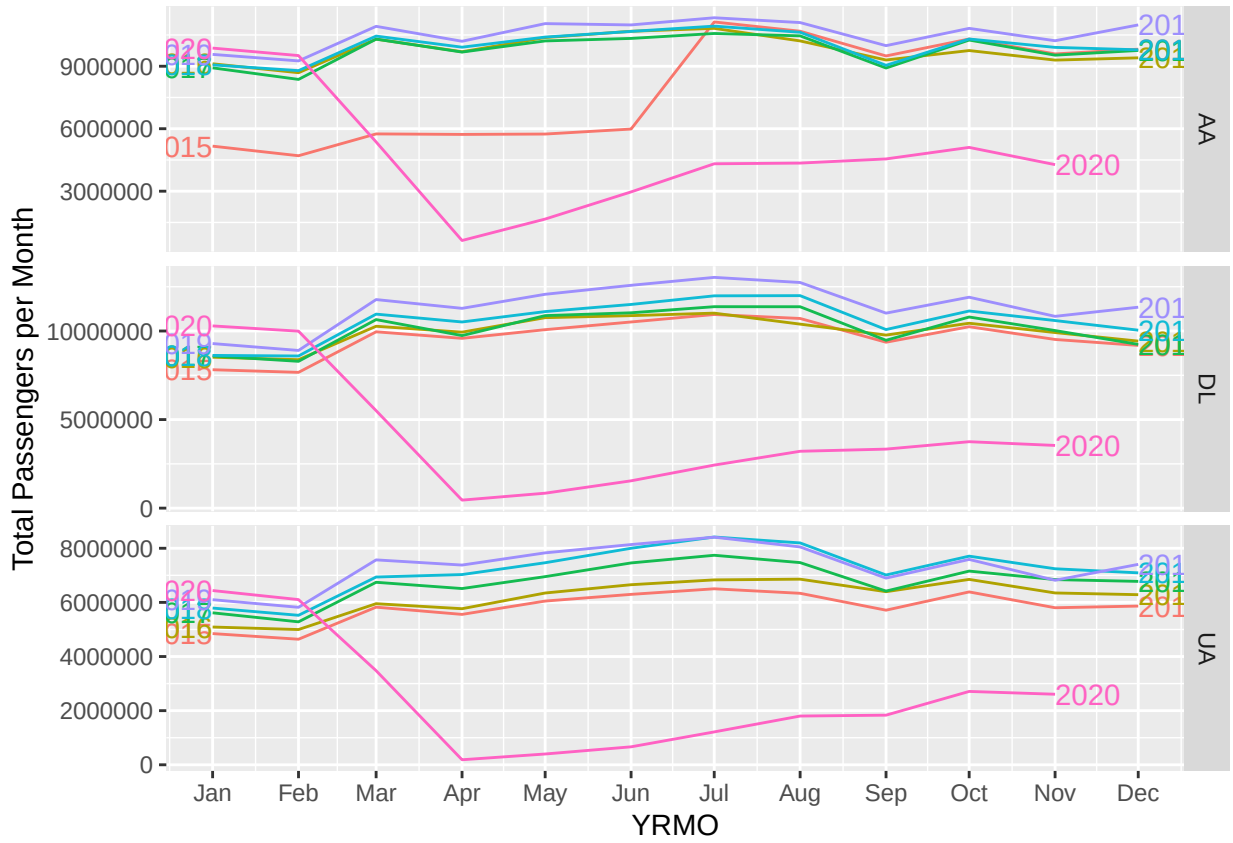
```
#Display density plots of distance flown between the 3 carriers
ggplot(big3, aes(x = TotalDistance, fill = UNIQUE_CARRIER)) +
  geom_density(position = "identity", alpha = 0.4, color = "black") +
  facet_grid(YEAR ~ .)
```




```
ggplot(big3, aes(x = TotalRPM, fill = UNIQUE_CARRIER)) +
  geom_density(position = "identity", alpha = 0.4, color = "black") +
  facet_grid(YEAR ~ .)
```




```
#Seasonal plot
big3 %>%
  feasts::gg_season(TotalPax, labels = "both") + ylab("Total Passengers per Month")
```



The Seasonal Plot above shows a pretty clear representation of the seasonal trends in air travel amongst these three airlines. (Note: Y-axis is not consistent; adjusted for each airline).

Prior to 2020, the most popular year for air travel was 2018 (according to most metrics of interest here).

Seasonal trends are somewhat as one would expect for air travel. We can see that February and September are the low points for passengers being transported, and the summer and holiday months create a spike for these 3 airlines.

There was an interesting phenomenon for American Airlines in summer 2015, where their Total Passenger count increased greatly. This could be due to general corporate expansion, acquisition of new (or bigger) airplanes, or initiation of new flight routes.

(A little research showed that AA did place "the 'largest aircraft order in history' in July 2011, purchasing 460 'next generation' Boeing 737 and Airbus A320 aircraft for delivery between 2013 and 2022. Also, 2015 was the year they completed a merger with US Airways: On April 8, 2015, the Federal Aviation Administration awarded American Airlines and US Airways a single operating certificate." At some point in 2015, I'm assuming the Transportation Bureau started counting the former-US Airways flights as part of American Airlines, and it was probably during the June or July timeframe.)

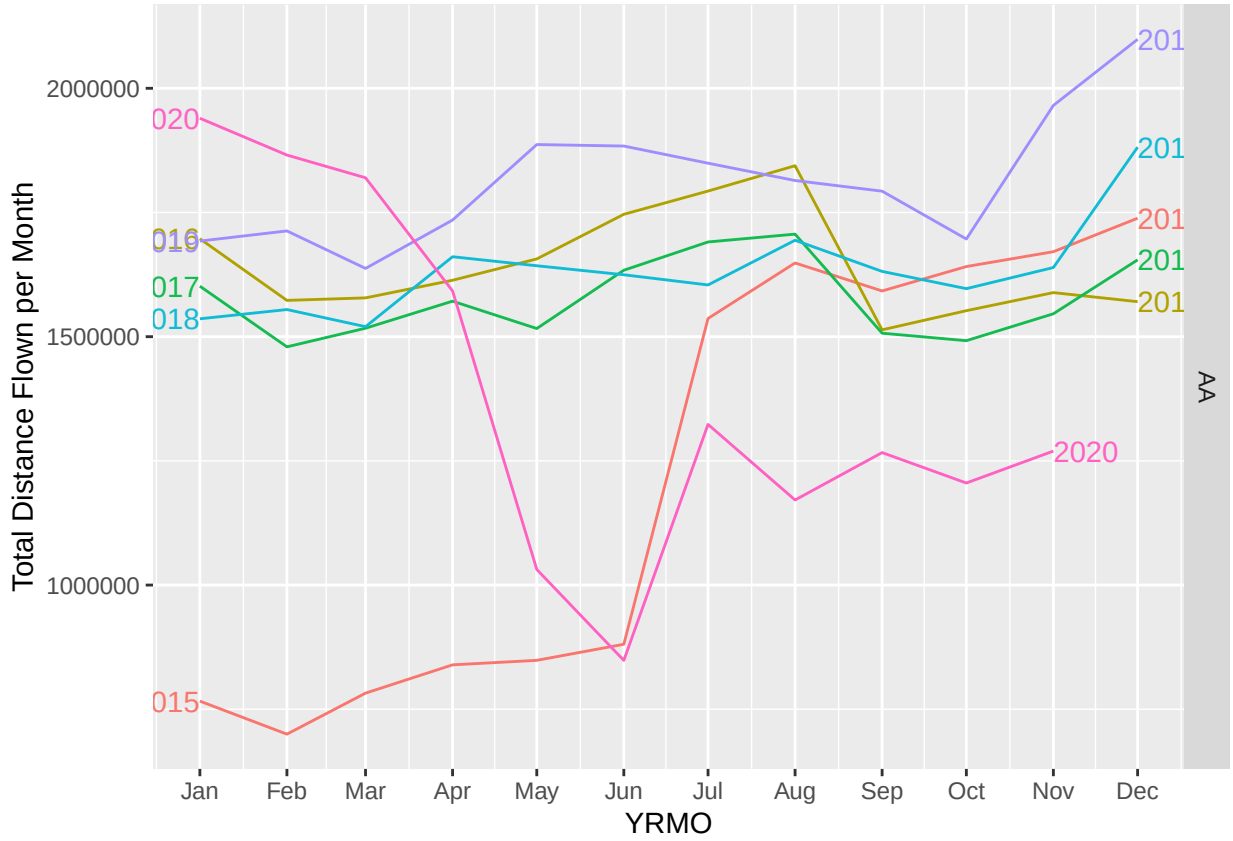
[https://en.wikipedia.org/wiki/History_of_American_Airlines]

I was interested to see what happened with the other metrics for AA during summer 2015.

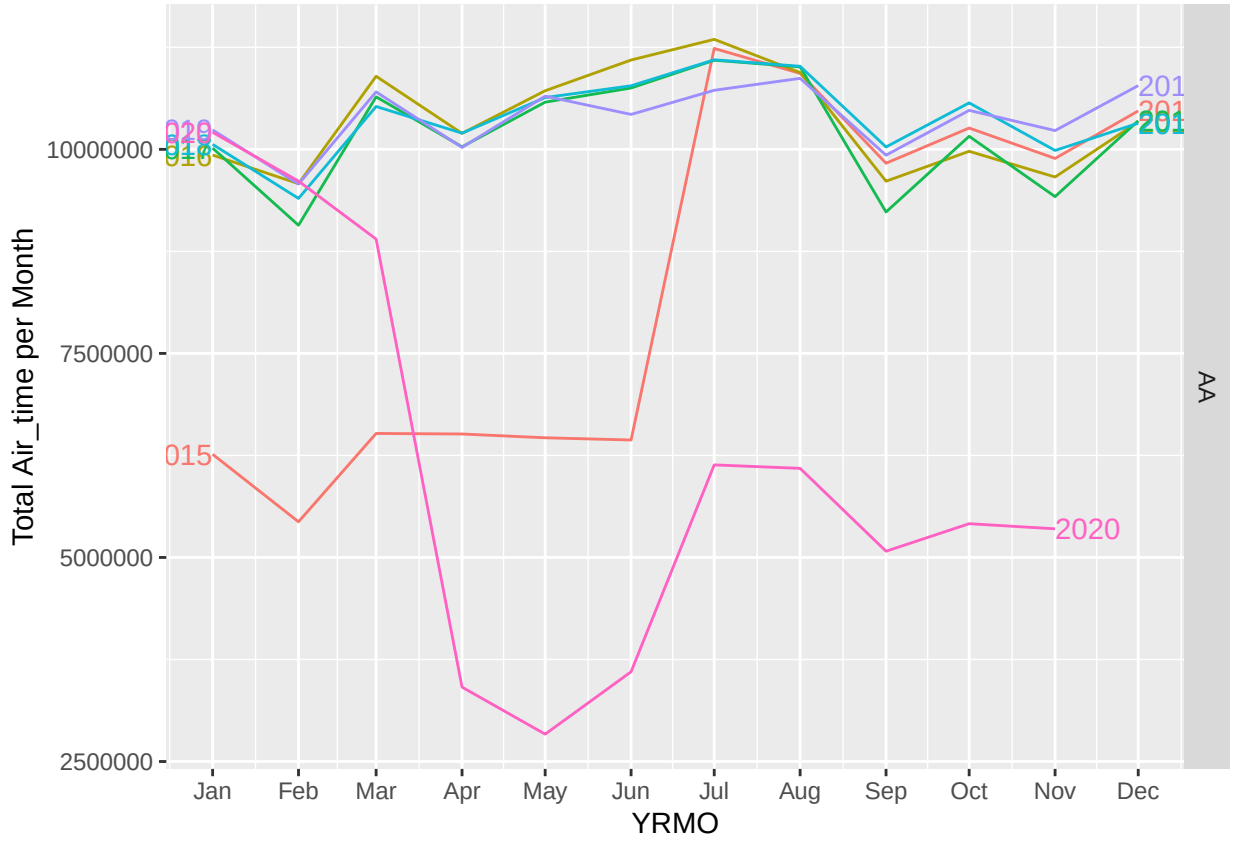
The first chart depicts "Total Distance flown" per month during 2015.

The second chart depicts "Total air time flown" per month during 2015.

```
big3 %>%
  filter(UNIQUE_CARRIER == "AA") %>%
  feasts::gg_season(TotalDistance, labels = "both") + ylab("Total Distance Flown per Month")
```




```
big3 %>%
  filter(UNIQUE_CARRIER == "AA") %>%
  feasts::gg_season(TotalAIR_TIME, labels = "both") + ylab("Total Air_time per Month")
```

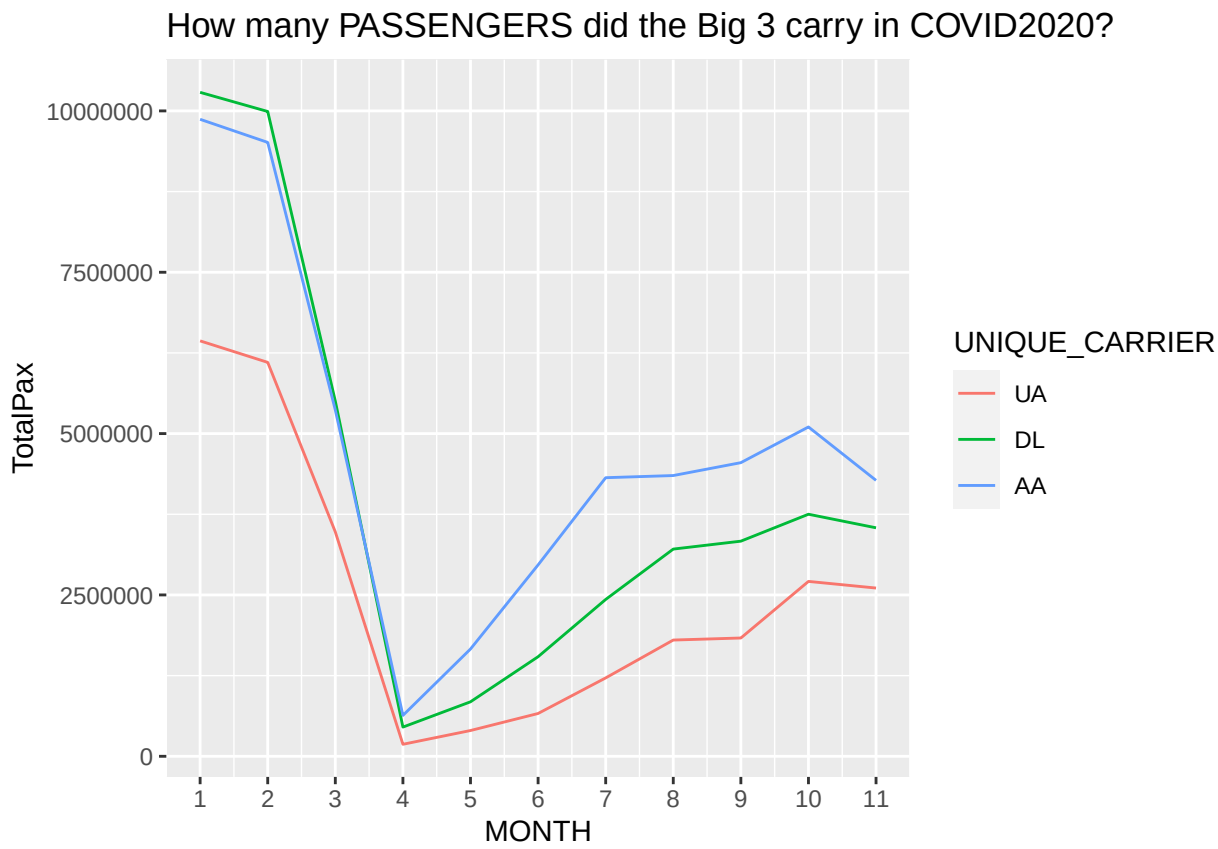


Sure enough, American Airlines experienced a huge jump in summer '15, to where Distance and Air Time flown became consistent with 2016-2019 levels. This leads me to believe that 2015 was a year in which external factors were at play – important to note but probably will not play into my analysis any further.

More directly to the questions I'm trying to answer:

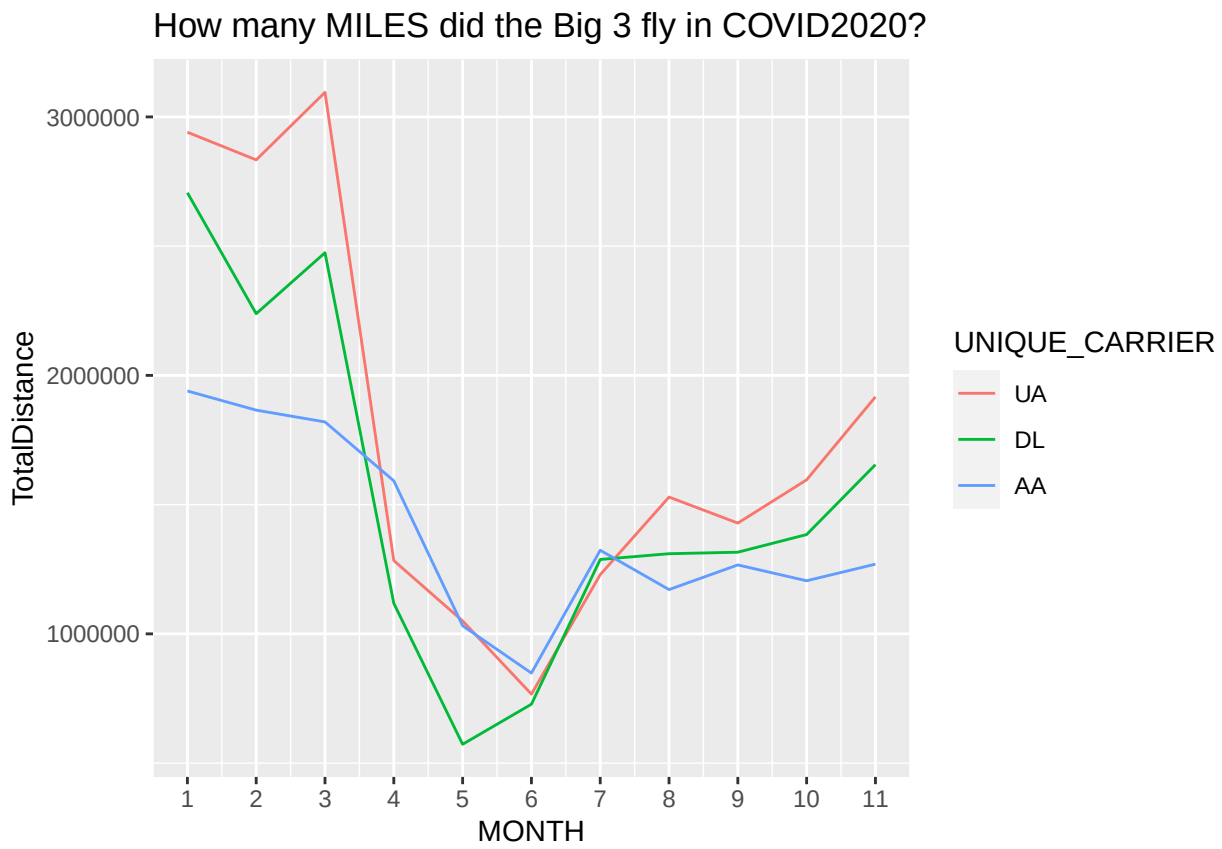
Another aspect of these gg_season plots that I see is the stabilization of Passengers flown as the year 2020 proceeded. The latest data in this set is from November 2020. We can see below how each airline performed through November 2020. Their monthly passenger numbers were essentially cut to near 500,000 by April 2020, and as of November they did climb back towards 5 million (with American Airlines moving the most pax). This is roughly half the pre-COVID19 passenger count for these airlines. The question we can address is how long it may take to recover to pre-COVID19 numbers.

```
#Show total passengers trend since 2020 Jan
filter(big3, YEAR == "2020") -> big32020
big32020 %>%
  ggplot(mapping = aes(x = MONTH, y = TotalPax, color = UNIQUE_CARRIER)) + geom_line() + ggtitle("How ma
```



This one shows a similar story, in terms of Distance Flown over 2020.

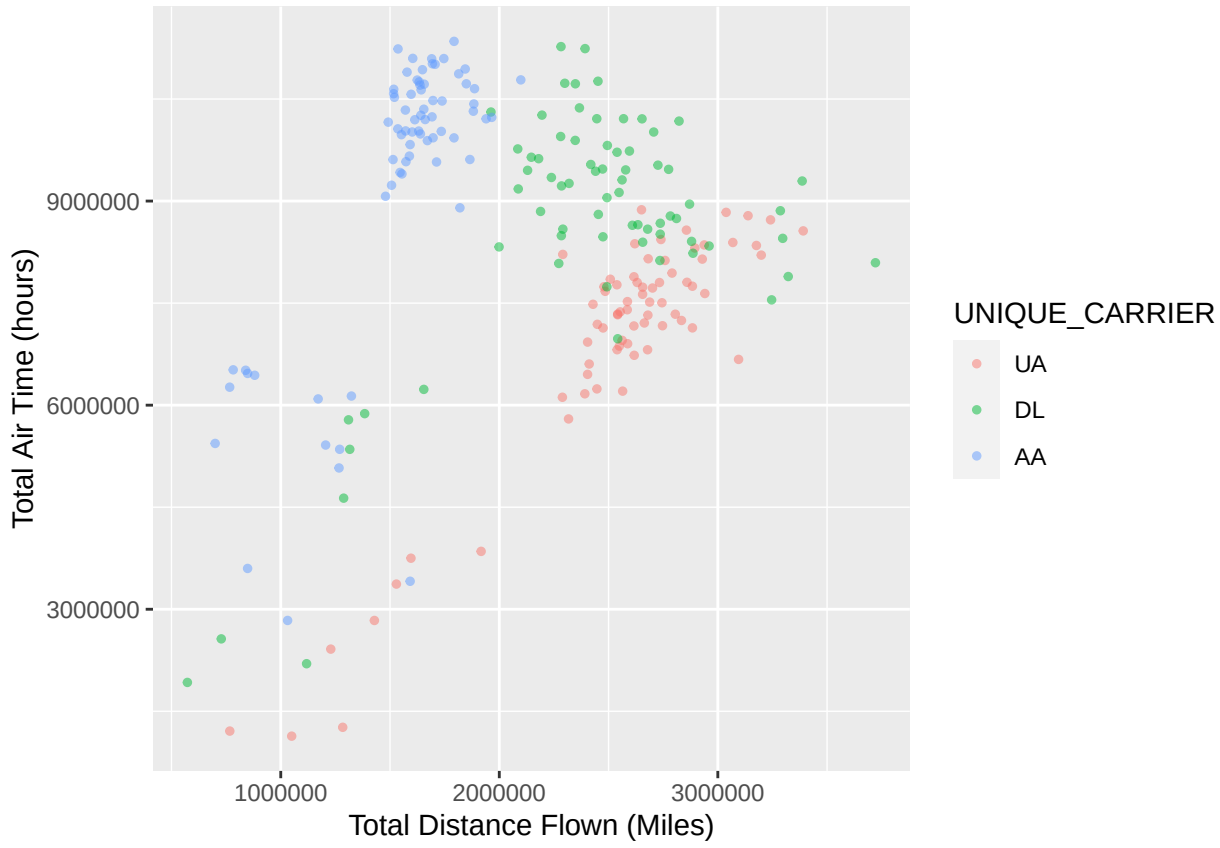
```
big32020 %>%
  ggplot(mapping = aes(x = MONTH, y = TotalDistance, color = UNIQUE_CARRIER)) + geom_line() + ggtitle("
```



I also wanted to plot some possibly related variables in a scatter plot, to see if there was anything to discover.

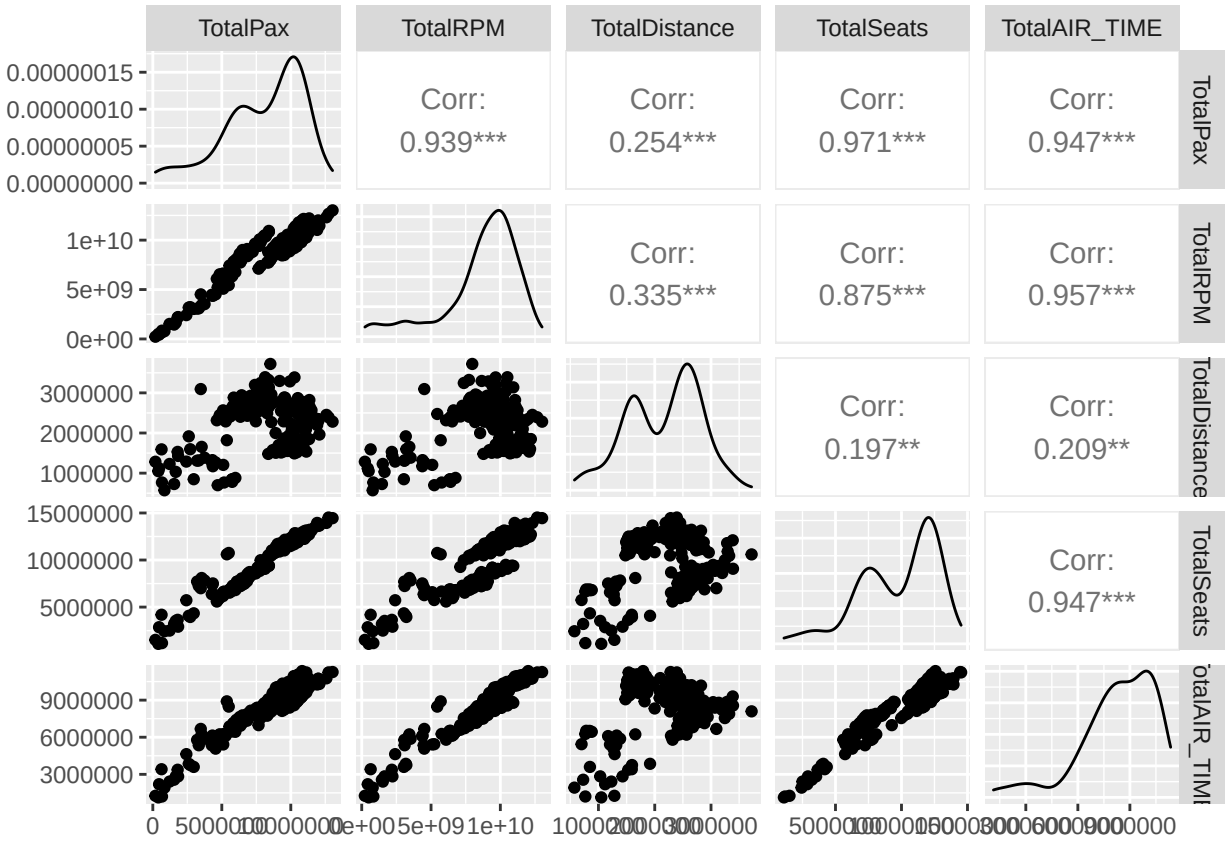
Below is Air Time (hours spent in flight for the month) vs. Distance Flown (miles). Of course, as distance flown increases, we expect to see an increase in air time, but is it linear?

```
#Is there a relationship between Total Air_Time and Distance Flown?
big3 %>%
  ggplot(aes(x = TotalDistance, y = TotalAIR_TIME))+
  geom_point(size=1, aes(colour = UNIQUE_CARRIER), alpha =0.5)+
  labs(y = "Total Air Time (hours)", x = "Total Distance Flown (Miles)")
```




```
timedistancemodel <- lm(TotalAIR_TIME ~ TotalDistance, data = big3)
summary(timedistancemodel)
```

```
##
## Call:
## lm(formula = TotalAIR_TIME ~ TotalDistance, data = big3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -6440964 -1247921   -96330  1905208  3353720
##
## Coefficients:
##                   Estimate   Std. Error t value Pr(>|t|)
## (Intercept)   6806163.4097  510856.4320  13.323  < 2e-16 ***
## TotalDistance       0.7006       0.2261   3.098  0.00221 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2117000 on 211 degrees of freedom
## Multiple R-squared:  0.04352,    Adjusted R-squared:  0.03899
## F-statistic:   9.6 on 1 and 211 DF,  p-value: 0.00221
```



The Ggally relationship plots above show correlation amongst features. Highly linear relationships seem to exist between many of these.

#SECTION III: FORMAL MODEL DEVELOPMENT

After careful consideration, I decided to attempt a model of each airline's Revenue Passenger Miles over time using an Autoregressive Integrated Moving Average (model) with seasonal and non-seasonal components. The procedures outlined below describe the process for model generation.

Modeling procedure borrowed from *Forecasting Principles and Practice, 3rd edition.*

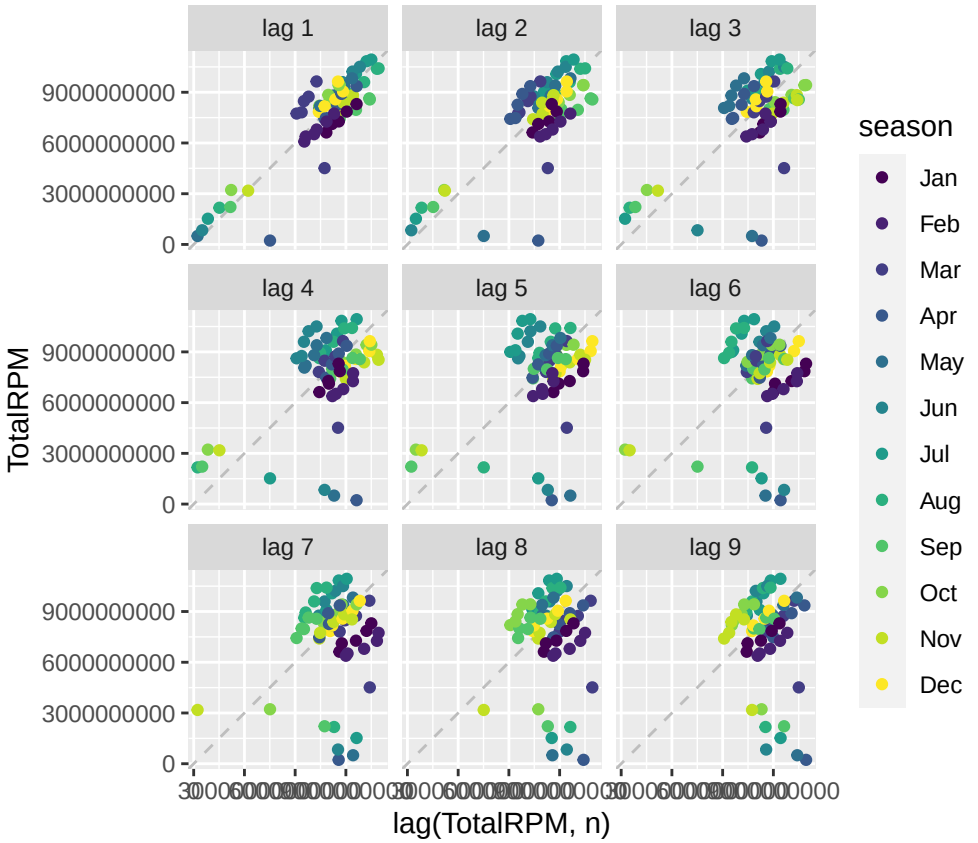
When fitting an ARIMA model to a set of time series data, the following procedure provides a useful general approach.

1. Plot the data and identify any unusual observations.

We did most of this in the exploratory data analysis above, but I wanted to take a look at a few more informative plots for ARIMA model generation.

Plot the lag plots. This helps us look for possible seasonality. There seems to be strong positive relationship in lag 1, indicating potential seasonality. For the other lag plots, it's hard to see a pattern.

```
big3 %>%
  group_by(YEAR, UNIQUE_CARRIER) %>%
  filter(UNIQUE_CARRIER == "UA") %>%
  gg_lag(TotalRPM, geom = "point")
```



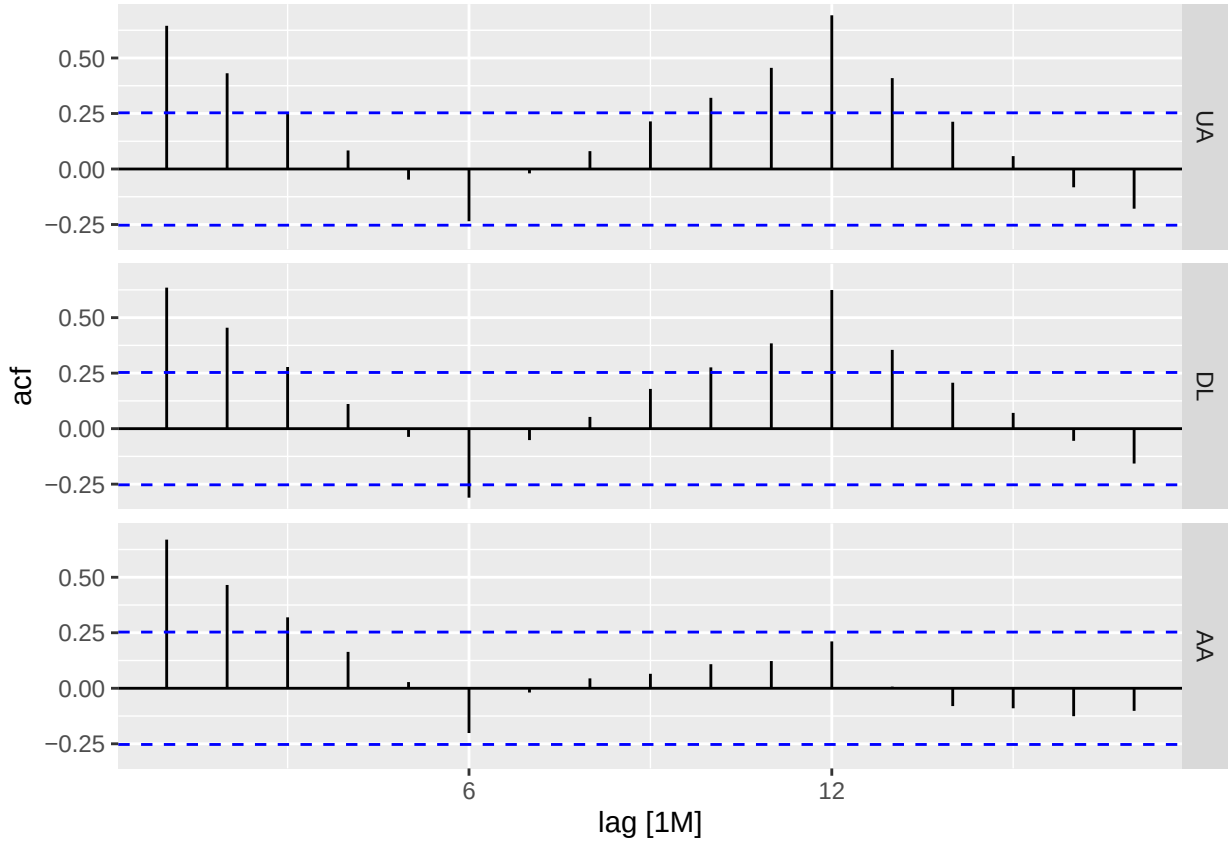
Plot the ACFs and PACFs for TotalRPM.

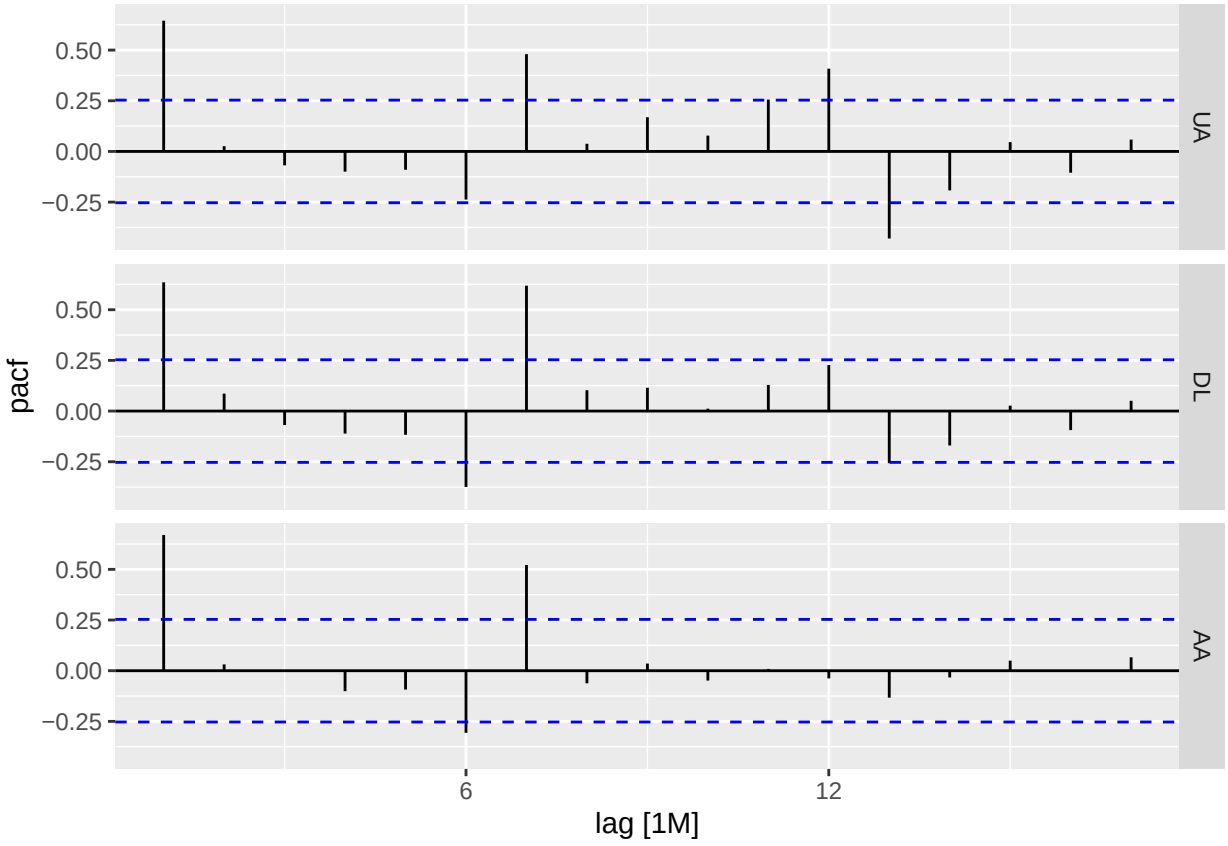
If one or more large spikes are outside these bounds, or if substantially more than 5% of spikes are outside these bounds, then the series is probably not white noise.

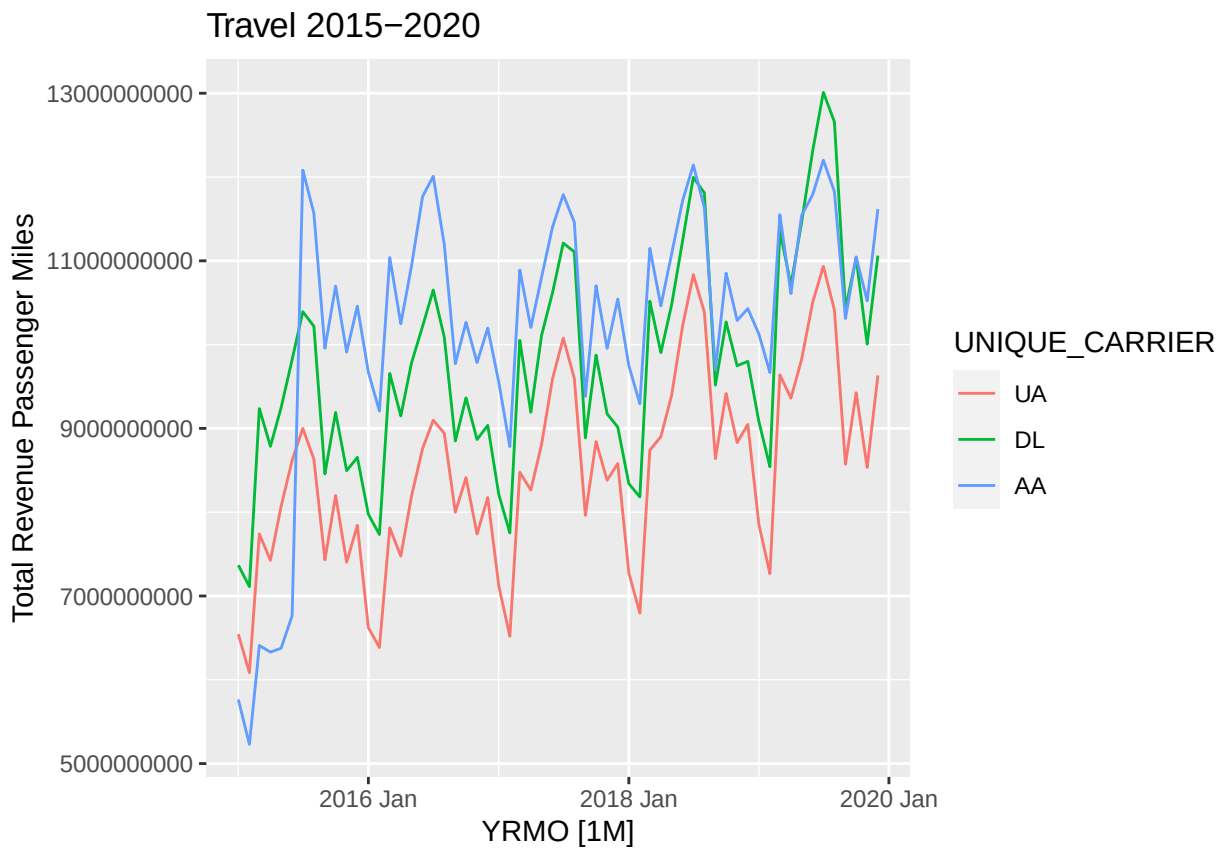
```
## # A tsibble: 180 x 3 [1M]
## # Key:       UNIQUE_CARRIER [3]
## # Groups:    UNIQUE_CARRIER [3]
##    UNIQUE_CARRIER     YRMO   TotalRPM
##    <fct>             <mth>      <dbl>
##  1 UA             2015 Jan 6544458813
##  2 UA             2015 Feb 6084625059
##  3 UA             2015 Mar 7740826373
##  4 UA             2015 Apr 7425806627
##  5 UA             2015 May 8069606981
##  6 UA             2015 Jun 8619127865
##  7 UA             2015 Jul 9000164187
##  8 UA             2015 Aug 8632732270
##  9 UA             2015 Sep 7431862199
## 10 UA             2015 Oct 8199013336
## # ... with 170 more rows
```

```
## # A tsibble: 36 x 3 [1M]
## # Key:       UNIQUE_CARRIER [3]
##    UNIQUE_CARRIER   lag     acf
##    <fct>          <lag>   <dbl>
```

```
##  1 UA                1M  0.645
##  2 UA                2M  0.431
##  3 UA                3M  0.248
##  4 UA                4M  0.0835
##  5 UA                5M -0.0479
##  6 UA                6M -0.235
##  7 UA                7M -0.0196
##  8 UA                8M  0.0805
##  9 UA                9M  0.214
## 10 UA               10M  0.321
## # ... with 26 more rows
```




pacf
0.50
0.25
0.00
−0.25
UA
DL
AA
6
12
lag [1M]



2. If necessary, transform the data (using a Box-Cox transformation) to stabilise the variance.

Here I split the project: in order to provide adequate analysis, I decided to formulate 3 separate models with the same structure. Each airline's forecast will be best predicted if optimizing the parameters based on that particular carrier's historical data. Each airline carrier has its own model:

a. UAtravel is modeled with 'fitUA'

b. DLtravel is modeled with 'fitDL'

c. AAtravel is modeled with 'fitAA'

NOTE: For American Airlines *only*, I excluded 2015 data altogether. As noted in my exploratory data analysis, 2015 seemed to be an anomaly year with an unprecedented RPM increase, likely attributed to corporate re-structuring and acquisition of another airline's assets. So AAtravel models are built from the 48 months between 2016JAN and 2019DEC. The final AA model ended up being quite good when the 2015 data was discounted.

```
#create a UAtravel subset, which is United Airlines pre-2020
UAtravel <- preCovid %>%
  filter(UNIQUE_CARRIER=="UA") %>%
  ungroup()
  # select(TotalRPM, YRMO)

#create a DLtravel subset, which is Delta Airlines pre-2020
DLtravel <- preCovid %>%
  filter(UNIQUE_CARRIER=="DL")
```

```
  # select(YEAR, MONTH, TotalRPM, -YRMO)

#create a AAtravel subset, which is American Airlines Jan2016-Dec2019

#NOTE: For UA only, I excluded 2015 altogether, as it seemed to be an anomaly year with an unprecedente

AAtravel <- big3 %>%
  group_by(UNIQUE_CARRIER) %>%
  filter(UNIQUE_CARRIER =='AA') %>%
  filter(YEAR < 2020) %>%
  filter(YEAR > 2015) %>%
  select(YRMO, TotalRPM) %>%
  ungroup()
```

Data transformation - Box Cox - Using Guerrero features search for lambda values. The goal is to make the size of seasonal variation about the same across time series. Once we transform the data, we can look at the first-difference and second-difference to help determine what value of parameters to use for (p,d,q) and (P,D,Q).

```
#extract the best lambda using guerrero features
(lambdaUA <- UAtravel %>%
  features(TotalRPM, features = guerrero) %>%
  pull(lambda_guerrero))
```

```
## [1] -0.795114
```

```
(lambdaDL <- DLtravel %>%
  features(TotalRPM, features = guerrero) %>%
  pull(lambda_guerrero))
```

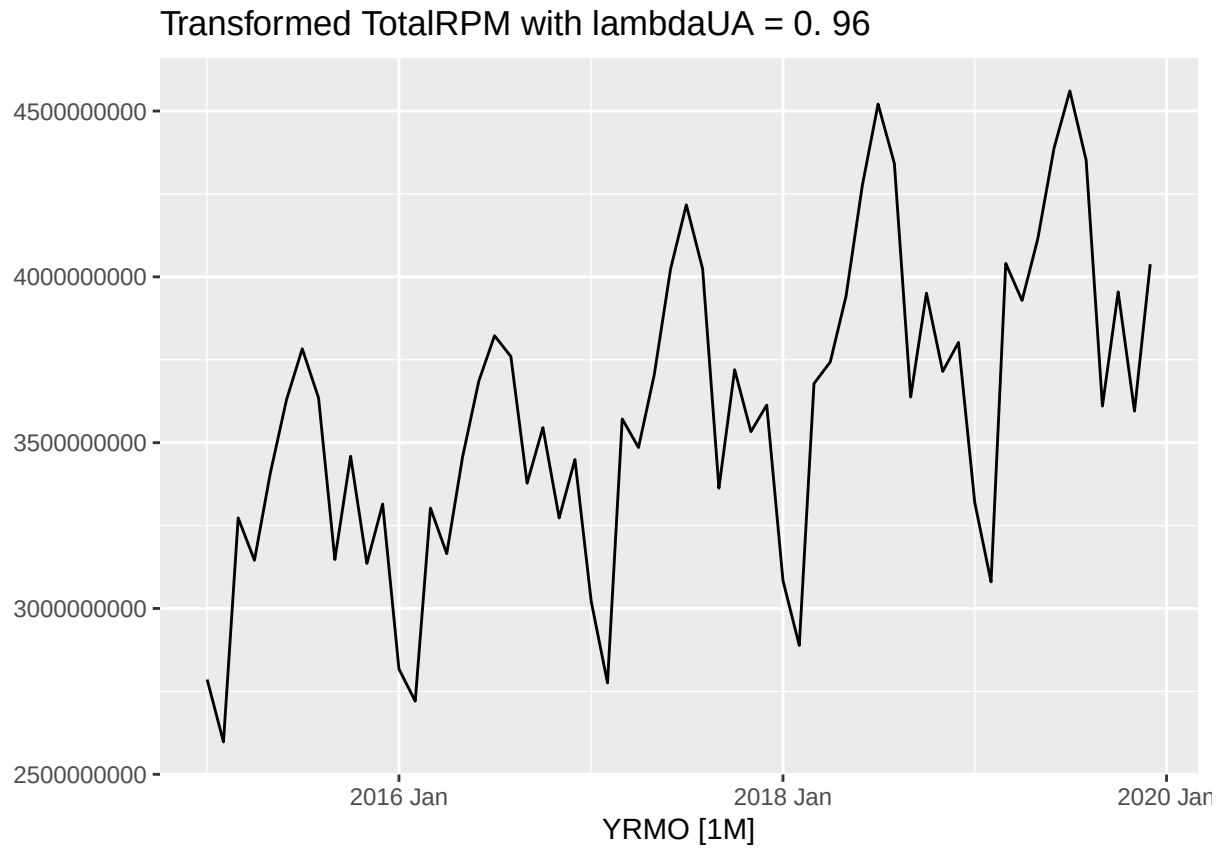
```
## [1] -0.7071929
```

```
(lambdaAA <- AAtravel %>%
  features(TotalRPM, features = guerrero) %>%
  pull(lambda_guerrero))
```

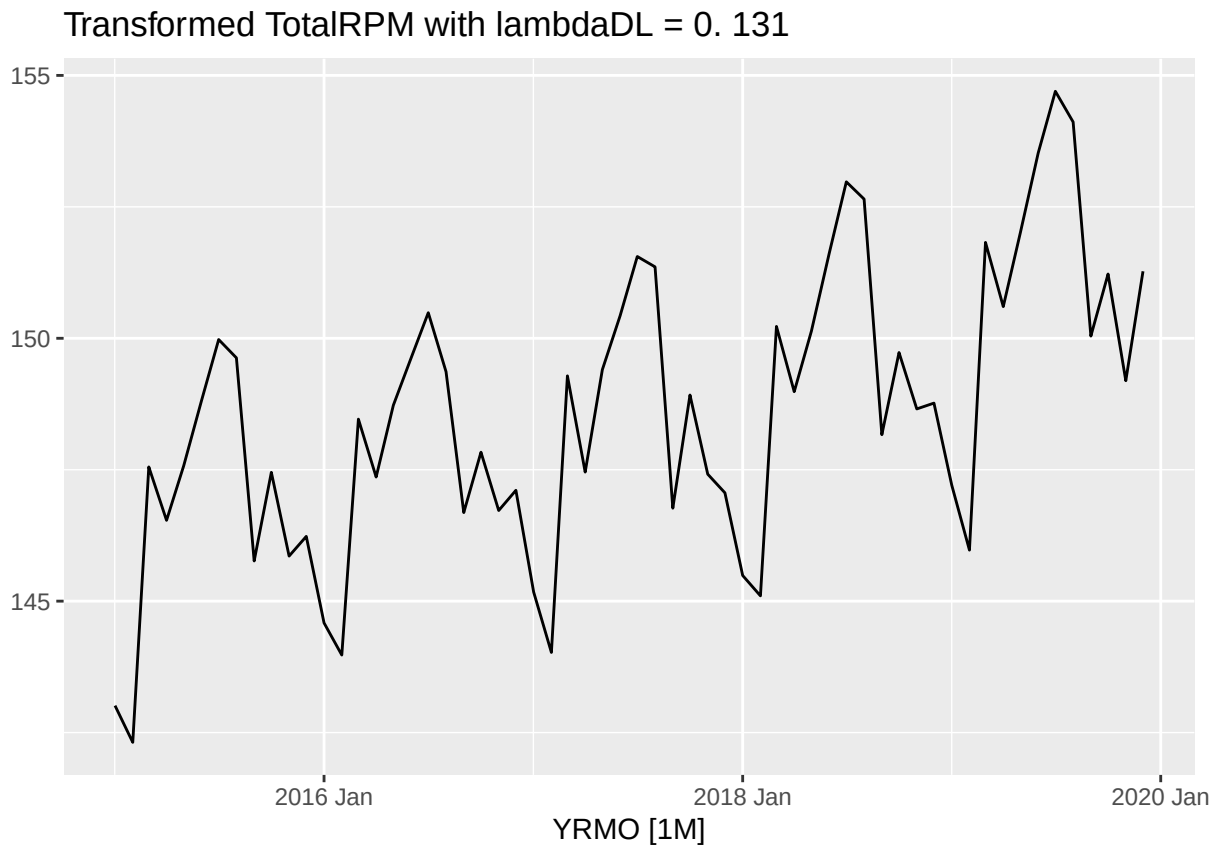
```
## [1] 1.999927
```

```
#alternate way to automatically extract optimal Lambda:
EnvStats::boxcox(UAtravel$TotalRPM,  optimize = TRUE)$data %>% as_tibble() ->UAboxcox
EnvStats::boxcox(DLtravel$TotalRPM,  optimize = TRUE)$data %>% as_tibble() ->DLboxcox
EnvStats::boxcox(AAtravel$TotalRPM,  optimize = TRUE)$data %>% as_tibble() ->AAboxcox

# transform the data: new preCovid data sets are UAtransform, DLtransform, AAtransform
UAtransform <- bind_cols(UAtravel, UAboxcox)
#print(EnvStats::boxcox(UAtravel$TotalRPM,  optimize = TRUE))

DLtransform <- bind_cols(DLtravel, DLboxcox)
#print(EnvStats::boxcox(DLtravel$TotalRPM,  optimize = TRUE))

AAtransform <- bind_cols(AAtravel, AAboxcox)
```

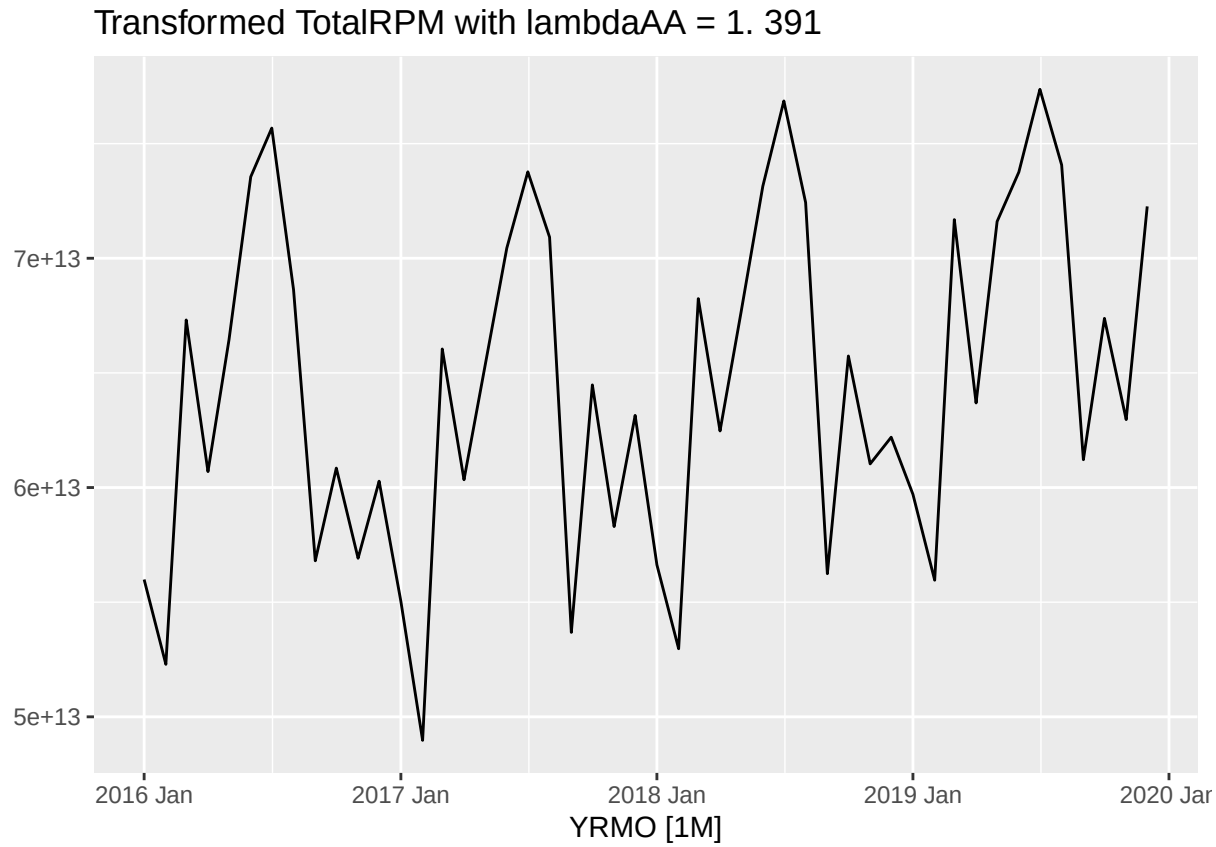
```
#print(EnvStats::boxcox(AAtravel$TotalRPM,  optimize = TRUE))

#plot box_cox for UA
UAtravel %>%
  autoplot(box_cox(TotalRPM, .960425)) +
  labs(y = "",
       title = latex2exp::TeX(paste0(
         "Transformed TotalRPM with $\\lambdaUA$ = ",
         round(.960425,3))))
```




```
#plot box_cox for DL
DLtravel %>%
  autoplot(box_cox(TotalRPM,  0.131388)) +
  labs(y = "",
       title = latex2exp::TeX(paste0(
         "Transformed TotalRPM with $\\lambdaDL$ = ",
         round( 0.131388,3))))
```




```
#plot box_cox for AA
AAtravel %>%
  autoplot(box_cox(TotalRPM, 1.391173)) +
  labs(y = "",
       title = latex2exp::TeX(paste0(
         "Transformed TotalRPM with $\\lambdaAA$ = ",
         round(1.391173,3))))
```



Use feat_acf to get a summary of the autocorrelation features in our preCovid data

```
UAtransform %>% features(TotalRPM, feat_acf)
```

```
## # A tibble: 1 x 8
##   UNIQUE_CARRIER  acf1 acf10 diff1_acf1 diff1_acf10 diff2_acf1 diff2_acf10
##   <fct>          <dbl> <dbl>      <dbl>       <dbl>      <dbl>       <dbl>
## 1 UA             0.645 0.884     -0.216       0.473     -0.568        1.08
## # ... with 1 more variable: season_acf1 <dbl>
```

```
DLtransform %>% features(TotalRPM, feat_acf)
```

```
## # A tibble: 1 x 8
##   UNIQUE_CARRIER  acf1 acf10 diff1_acf1 diff1_acf10 diff2_acf1 diff2_acf10
##   <fct>          <dbl> <dbl>      <dbl>       <dbl>      <dbl>       <dbl>
## 1 DL             0.635 0.911     -0.275       0.811     -0.615        1.62
## # ... with 1 more variable: season_acf1 <dbl>
```

```
AAtransform %>% features(TotalRPM, feat_acf)
```

```
## # A tibble: 1 x 8
##   UNIQUE_CARRIER  acf1 acf10 diff1_acf1 diff1_acf10 diff2_acf1 diff2_acf10
##   <fct>          <dbl> <dbl>      <dbl>       <dbl>      <dbl>       <dbl>
## 1 AA             0.358 0.614     -0.396       0.887     -0.657        1.69
## # ... with 1 more variable: season_acf1 <dbl>
```

Use feat_stl to compute STL-based features of each airline's historical series.

```
UAtransform %>% features(TotalRPM, feat_stl) -> UAstl
UAstl
```

```
## # A tibble: 1 x 10
##   UNIQUE_CARRIER trend_strength seasonal_streng~ seasonal_peak_y~
##   <fct>                  <dbl>            <dbl>            <dbl>
## 1 UA                     0.931            0.967                7
## # ... with 6 more variables: seasonal_trough_year <dbl>, spikiness <dbl>,
## #   linearity <dbl>, curvature <dbl>, stl_e_acf1 <dbl>, stl_e_acf10 <dbl>
```

```
DLtransform %>% features(TotalRPM, feat_stl) -> DLstl
DLstl
```

```
## # A tibble: 1 x 10
##   UNIQUE_CARRIER trend_strength seasonal_streng~ seasonal_peak_y~
##   <fct>                  <dbl>            <dbl>            <dbl>
## 1 DL                     0.945            0.970                7
## # ... with 6 more variables: seasonal_trough_year <dbl>, spikiness <dbl>,
## #   linearity <dbl>, curvature <dbl>, stl_e_acf1 <dbl>, stl_e_acf10 <dbl>
```

```
AAtransform %>% features(TotalRPM, feat_stl) -> AAstl
AAstl
```

```
## # A tibble: 1 x 10
##   UNIQUE_CARRIER trend_strength seasonal_streng~ seasonal_peak_y~
##   <fct>                  <dbl>            <dbl>            <dbl>
## 1 AA                     0.775            0.972                7
## # ... with 6 more variables: seasonal_trough_year <dbl>, spikiness <dbl>,
## #   linearity <dbl>, curvature <dbl>, stl_e_acf1 <dbl>, stl_e_acf10 <dbl>
```

feat_spectral will compute the (Shannon) spectral entropy of a time series, which is a measure of how easy the series is to forecast. A series which has strong trend and seasonality (and so is easy to forecast) will have entropy close to 0. A series that is very noisy (and so is difficult to forecast) will have entropy close to 1. As seen below, the noisiest series is United Airlines, but they are all <0.55 and thus we continue with the modeling with this in mind.

```
feat_spectral(UAtransform$TotalRPM, 12)
```

```
## spectral_entropy
##        0.5330575
```

```
feat_spectral(DLtransform$TotalRPM, 12)
```

```
## spectral_entropy
##        0.2167018
```

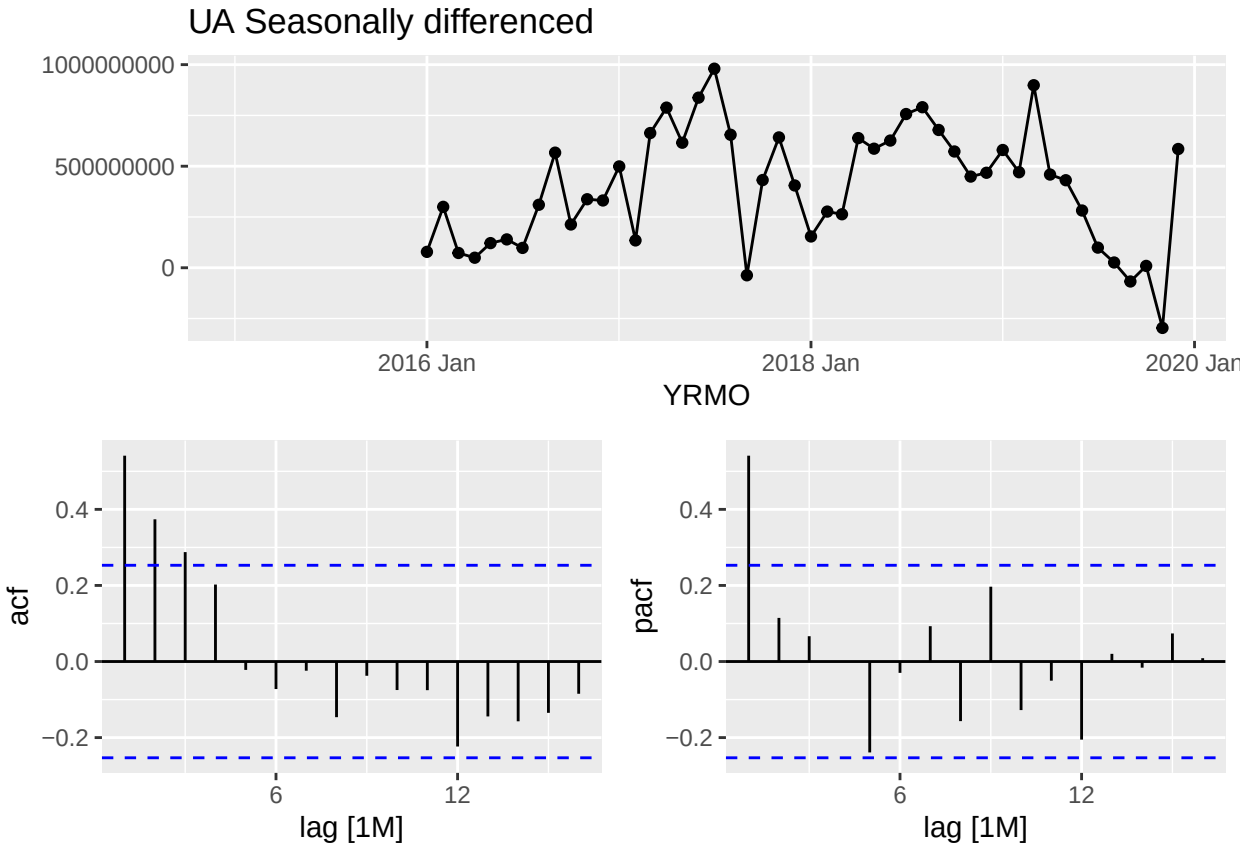
```
feat_spectral(AAtransform$TotalRPM, 12)
```

```
## spectral_entropy
##        0.3617763
```

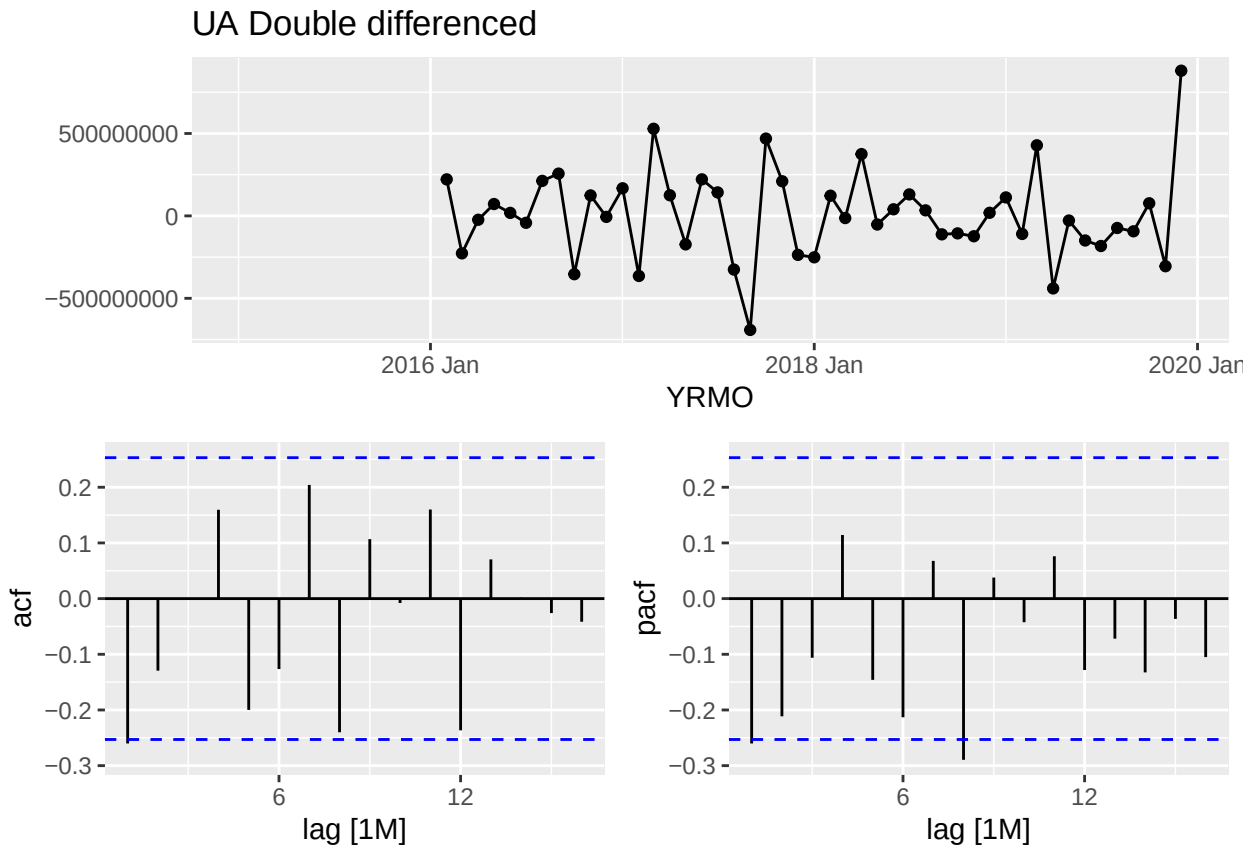
3. If the data are non-stationary, take first differences of the data until the data are stationary.

The data for all three airlines appears non-stationary, with an overall decline.

Below is a **United Airlines** Difference plot.

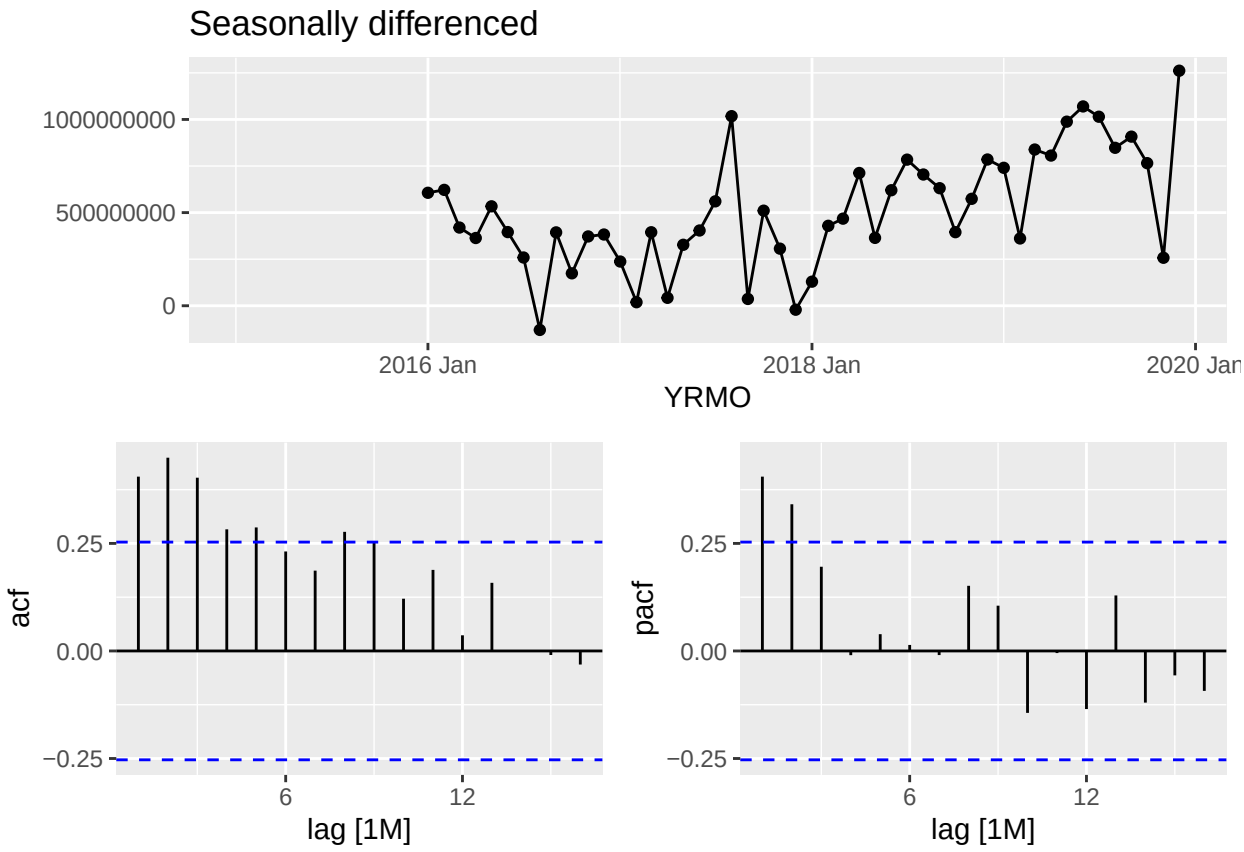


There is still clearly non-stationarity, so we take a further first difference.

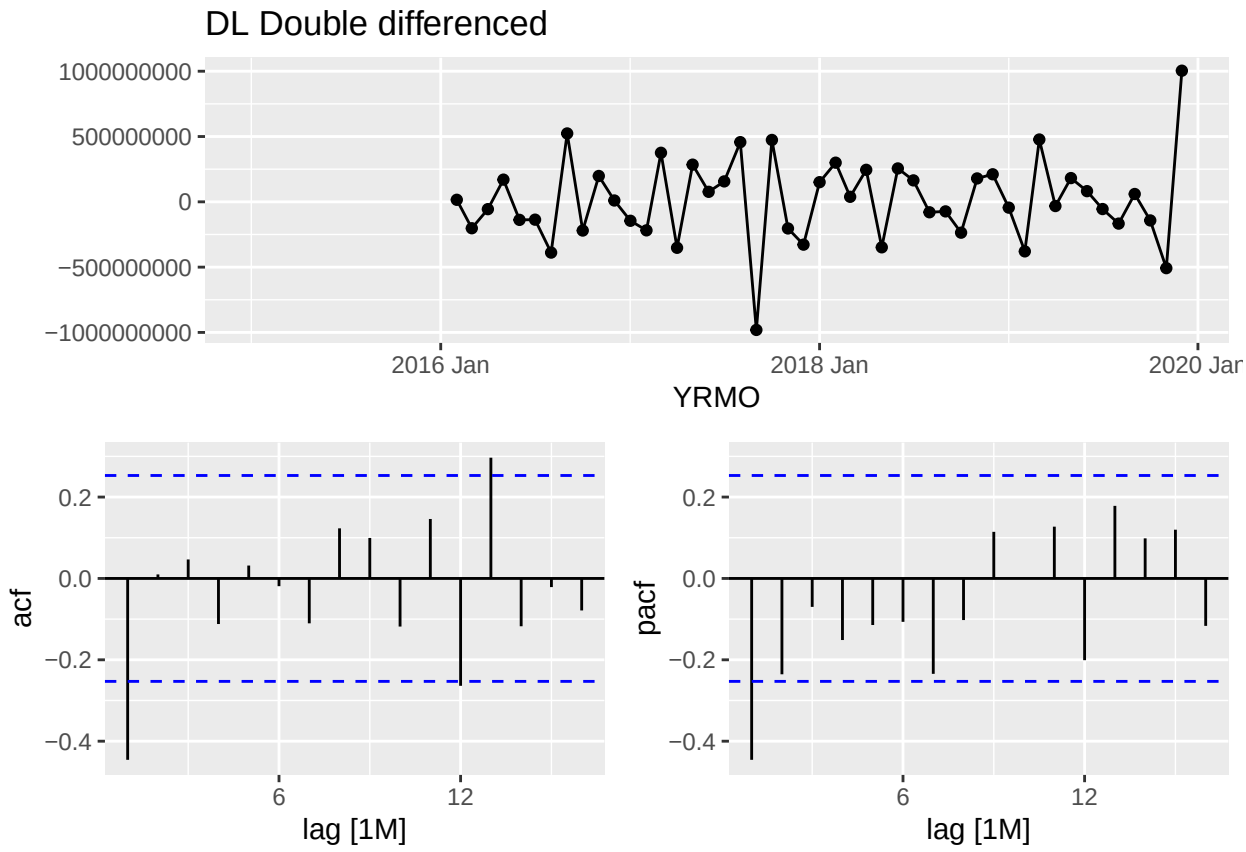


Now we have a double-differenced ACF and PACF from which to generate our ARIMA model.

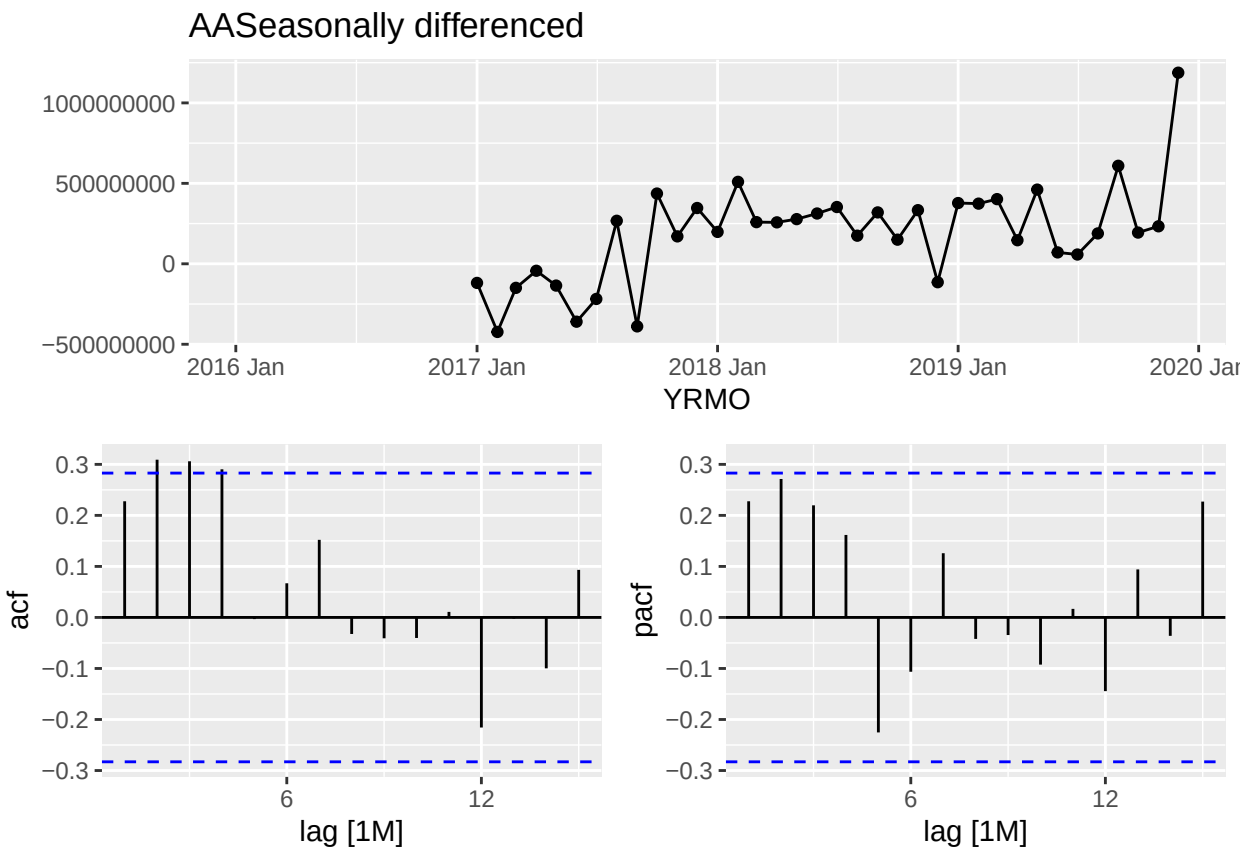
Below is a **Delta Airlines** Difference plot.



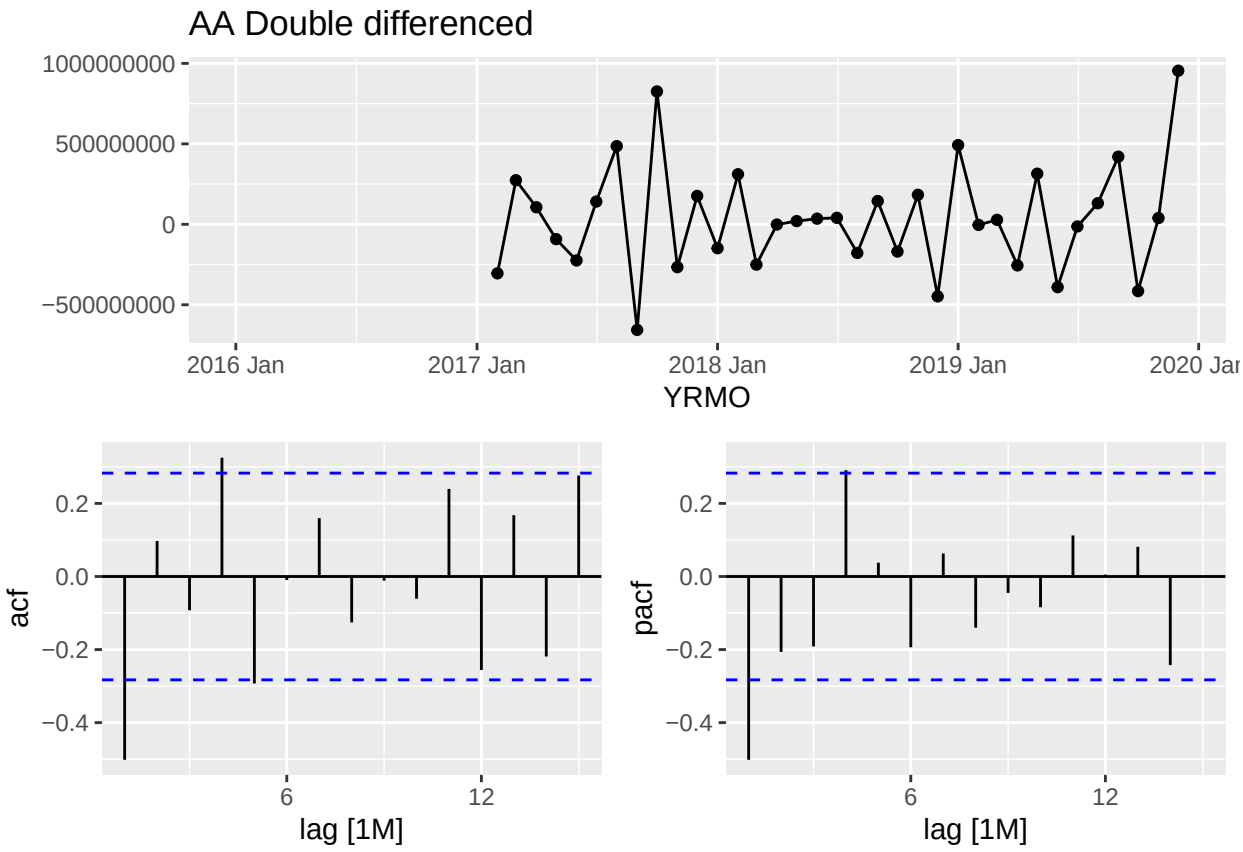
Non-stationarity is still observed, so we proceed with the double-differencing.



Below is an **American Airlines** Difference plot.



Non-stationarity is still observed, so we proceed with the double-differencing.



4. Examine the ACF/PACF: Which model coefficients/type is most appropriate?

UAtravel: For United Airlines, we have a significant ACF spike at lag 1, suggesting a non-seasonal MA(1) component. The PACF shows a significant relationship at lag 8 as well, but none beyond lag 8. Altogether, these indicate an $ARIMA(1,1,0)(8,2,0)_{12}$ model.

DLtravel: After double differencing: Analysis of these ACF and PCF plots indicates an appropriate ARIMA model may be $ARIMA(12,2,0)(1,2,0)_{12}$

AAtravel: After double-differencing, now the data is generally stationary around 0. Analysis of these ACF and PCF plots indicates an appropriate ARIMA model may be $ARIMA(4,2,0)(5,2,0)_{12}$

With these potential model parameters in mind, I proceeded to test them against the automatic ARIMA models in the *fable* package. The result is shown below. Ultimately, *fable* decided that the automatically generated models were better than my proposed models. Optimal model parameters are indicated in the tables. For forecasting, I used the models formulated by *fable* instead of the ARIMA models I proposed above.

```
#Let's try to auto build a model

# When we do the auto ARIMA functions, the data transformations are done automatically. So we will call
fitUA <- UAtravel %>%
  model(
    ARIMA110820 = ARIMA(TotalRPM ~ pdq(1,0,0) + PDQ(8,0,0)),
    UAauto = ARIMA(TotalRPM, stepwise = FALSE, approx = FALSE),
    UAstepwise = ARIMA(TotalRPM, stepwise = TRUE)
    )
```

```
fitDL <- DLtravel %>%
  model(
    ARIMA1220120 = ARIMA(TotalRPM ~ pdq(12,0,0) + PDQ(1,0,0)),
    DLauto = ARIMA(TotalRPM, stepwise = FALSE, approx = FALSE),
    DLstepwise = ARIMA(TotalRPM, stepwise = TRUE)
    )

fitAA <- AAtravel %>%
  model(
    ARIMA420520 = ARIMA(TotalRPM ~ pdq(4,2,0) + PDQ(5,2,0)),
    AAauto = ARIMA(TotalRPM, stepwise = FALSE, approx = FALSE),
    AAstepwise = ARIMA(TotalRPM, stepwise = TRUE)
    )
```

United Airlines model:

```
glance(fitUA)
```

```
## # A tibble: 2 x 9
##   UNIQUE_CARRIER .model      sigma2 log_lik   AIC  AICc   BIC ar_roots  ma_roots
##   <fct>          <chr>        <dbl>   <dbl> <dbl> <dbl> <dbl> <list>    <list>
## 1 UA             UAauto     6.32e16   -996. 1998. 1999. 2004. <cpl [1]> <cpl [1~
## 2 UA             UAstepwise 6.32e16   -996. 1998. 1999. 2004. <cpl [1]> <cpl [1~
```

```
broom::tidy(fitUA) #model coefficients
```

```
## # A tibble: 4 x 7
##   UNIQUE_CARRIER .model     term  estimate std.error statistic  p.value
##   <fct>          <chr>      <chr>    <dbl>     <dbl>     <dbl>    <dbl>
## 1 UA             UAauto     ar1      0.948    0.0441     21.5  2.68e-26
## 2 UA             UAauto     ma1     -0.474    0.172      -2.76 8.16e- 3
## 3 UA             UAstepwise ar1      0.948    0.0441     21.5  2.68e-26
## 4 UA             UAstepwise ma1     -0.474    0.172      -2.76 8.16e- 3
```

Delta Airlines model:

```
glance(fitDL)
```

```
## # A tibble: 2 x 9
##   UNIQUE_CARRIER .model      sigma2 log_lik   AIC  AICc   BIC ar_roots  ma_roots
##   <fct>          <chr>        <dbl>   <dbl> <dbl> <dbl> <dbl> <list>    <list>
## 1 DL             DLauto     6.99e16   -978. 1960. 1960. 1964. <cpl [0]> <cpl [1~
## 2 DL             DLstepwise 6.99e16   -978. 1960. 1960. 1964. <cpl [0]> <cpl [1~
```

```
broom::tidy(fitDL) #model coefficients
```

```
## # A tibble: 2 x 7
##   UNIQUE_CARRIER .model     term  estimate std.error statistic       p.value
##   <fct>          <chr>      <chr>    <dbl>     <dbl>     <dbl>         <dbl>
## 1 DL             DLauto     ma1     -0.740     0.104     -7.09 0.00000000606
## 2 DL             DLstepwise ma1     -0.740     0.104     -7.09 0.00000000606
```

American Airlines model:

```
glance(fitAA)
```

```
## # A tibble: 2 x 9
##   UNIQUE_CARRIER .model     sigma2 log_lik   AIC  AICc   BIC ar_roots   ma_roots
##   <fct>          <chr>       <dbl>   <dbl> <dbl> <dbl> <dbl> <list>     <list>
## 1 AA             AAauto    5.22e16   -725. 1458. 1459. 1464. <cpl [14]> <cpl [0~
## 2 AA             AAstepwi~ 5.48e16   -725. 1458. 1459. 1464. <cpl [12]> <cpl [2~
```

```
broom::tidy(fitAA) #model coefficients
```

```
## # A tibble: 6 x 7
##   UNIQUE_CARRIER .model     term  estimate std.error statistic      p.value
##   <fct>          <chr>      <chr>    <dbl>     <dbl>     <dbl>        <dbl>
## 1 AA             AAauto     ar1     -0.859     0.162     -5.29 0.00000667
## 2 AA             AAauto     ar2     -0.554     0.170     -3.26 0.00247
## 3 AA             AAauto     sar1    -0.600     0.172     -3.49 0.00132
## 4 AA             AAstepwise ma1     -0.929     0.135     -6.89 0.0000000537
## 5 AA             AAstepwise ma2      0.510     0.182      2.81 0.00815
## 6 AA             AAstepwise sar1    -0.513     0.190     -2.70 0.0106
```

5. Try your chosen model(s), and use the AICc to search for a better model.

So the models that minimize estimated information loss (lowest AICc):

United: UAauto

fitUA Model formulation:

$$(1-\phi_1 B)\ (1-\Phi_1 B^{12})(1-B)(1-B^{12})y_t = (1+\theta_1 B)\ (1+\Theta_1 B^{12})\varepsilon_t$$

and the optimal parameters:

[AR(1) component] $\phi = 0.9481228$

[MA(1) component] $\Theta = -0.4742058$

Delta: DLauto

fitDL Model formulation:

$$(1-\phi_1 B)\ (1-\Phi_1 B^{12})(1-B)(1-B^{12})y_t = (1+\theta_1 B)\ (1+\Theta_1 B^{12})\varepsilon_t$$

and the optimal parameters:

[MA(1) component] $\Theta = -0.7399767$

American: AAauto

fitAA Model formulation:

$$(1-\phi_1 B)\ (1-\Phi_1 B^{12})(1-B)(1-B^{12})y_t = (1+\theta_1 B)\ (1+\Theta_1 B^{12})\varepsilon_t$$

and the optimal parameters:

[AR(1) component] $\phi 1 = -0.8590480$

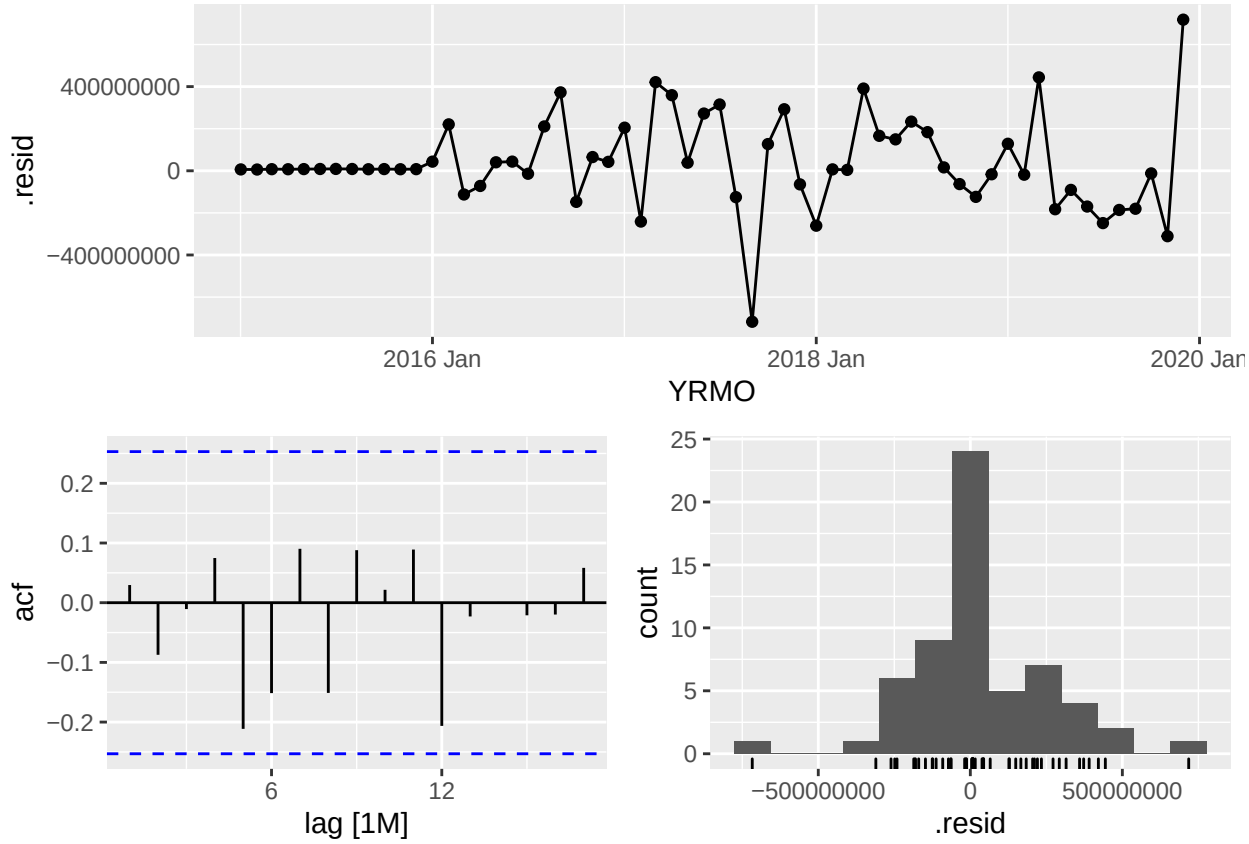
[AR(2) component] $\phi_2 = -0.5542507$

[seasonal AR(1) component] $\Phi_1 = -0.6003123$

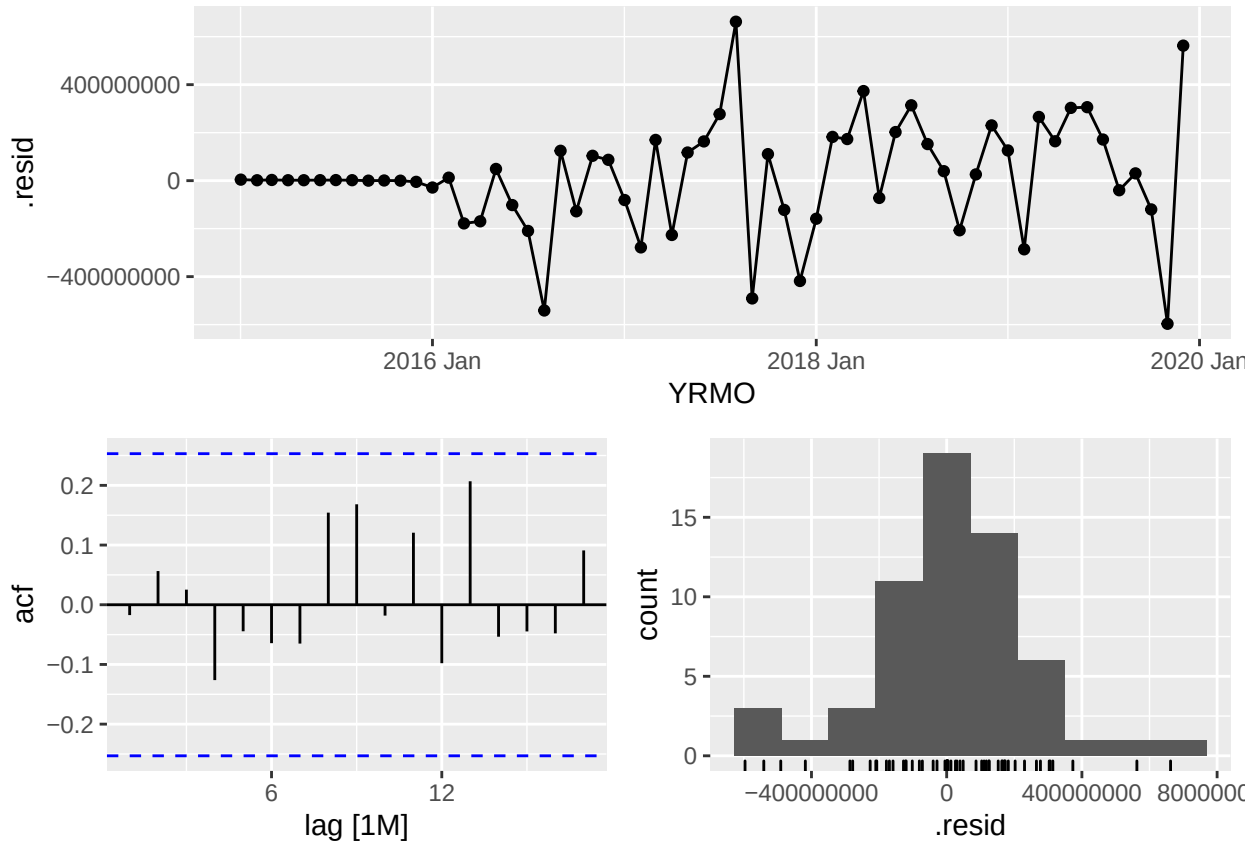
6. Check the residuals from your chosen model by plotting the ACF of the residuals, and doing a portmanteau test of the residuals. If they do not look like white noise, try a modified model.

Look at the residuals

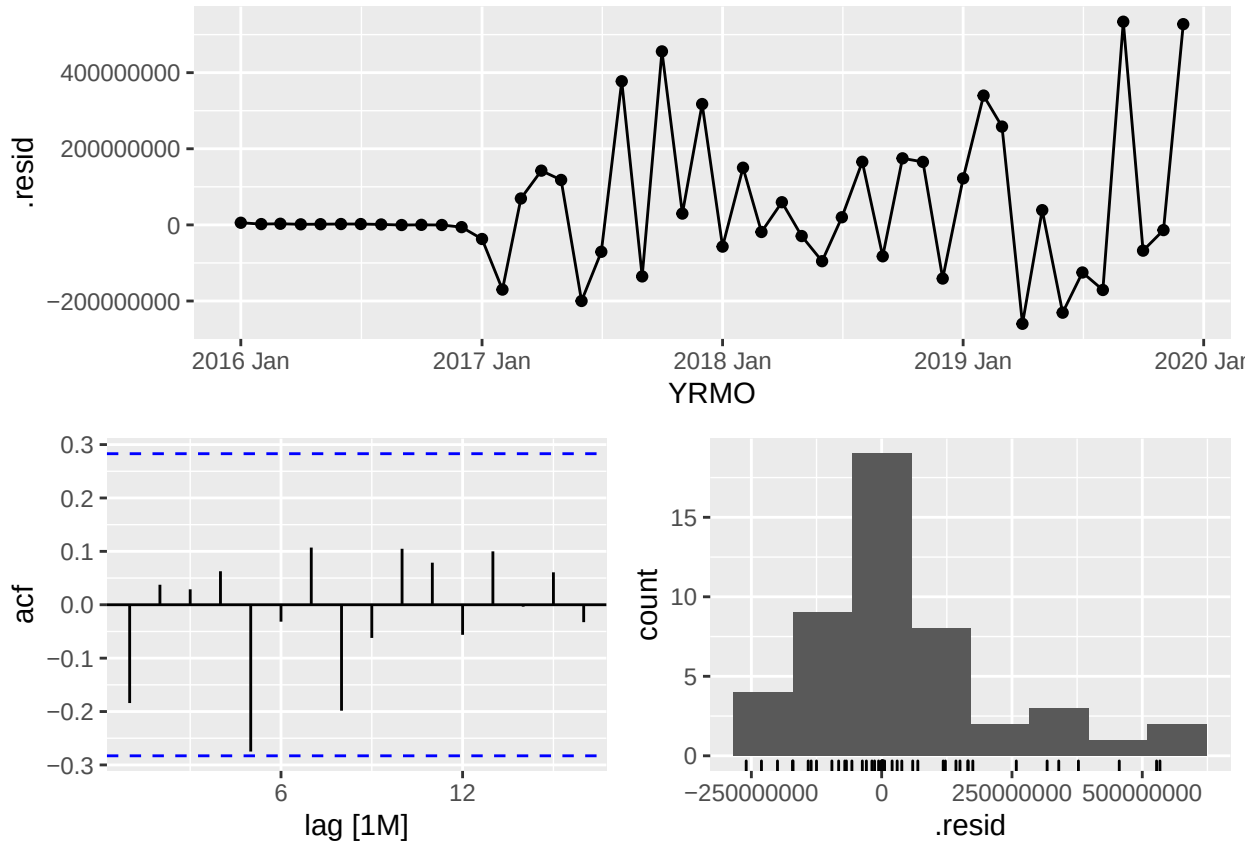
```
fitUA %>% select(UAauto) %>%
  gg_tsresiduals()
```




```
fitDL %>% select(DLauto) %>%
  gg_tsresiduals()
```




```
fitAA %>% select(AAauto) %>%
  gg_tsresiduals()
```



Use a Ljung-Box test to confirm that the residuals are consistent with white noise. The large p-values shown below indicate that our seasonal ARIMA models have all passed the checks and are ready for forecasting.

```
augment(fitUA) %>% features(.innov, ljung_box, lag=24, dof=4)
```

```
## # A tibble: 3 x 4
##   UNIQUE_CARRIER .model      lb_stat lb_pvalue
##   <fct>          <chr>         <dbl>     <dbl>
## 1 UA             ARIMA110820    NA      NA
## 2 UA             UAauto         18.0     0.590
## 3 UA             UAstepwise     18.0     0.590
```

```
augment(fitDL) %>% features(.innov, ljung_box, lag=24, dof=4)
```

```
## # A tibble: 3 x 4
##   UNIQUE_CARRIER .model       lb_stat lb_pvalue
##   <fct>          <chr>          <dbl>     <dbl>
## 1 DL             ARIMA1220120    NA      NA
## 2 DL             DLauto          17.0     0.651
## 3 DL             DLstepwise      17.0     0.651
```

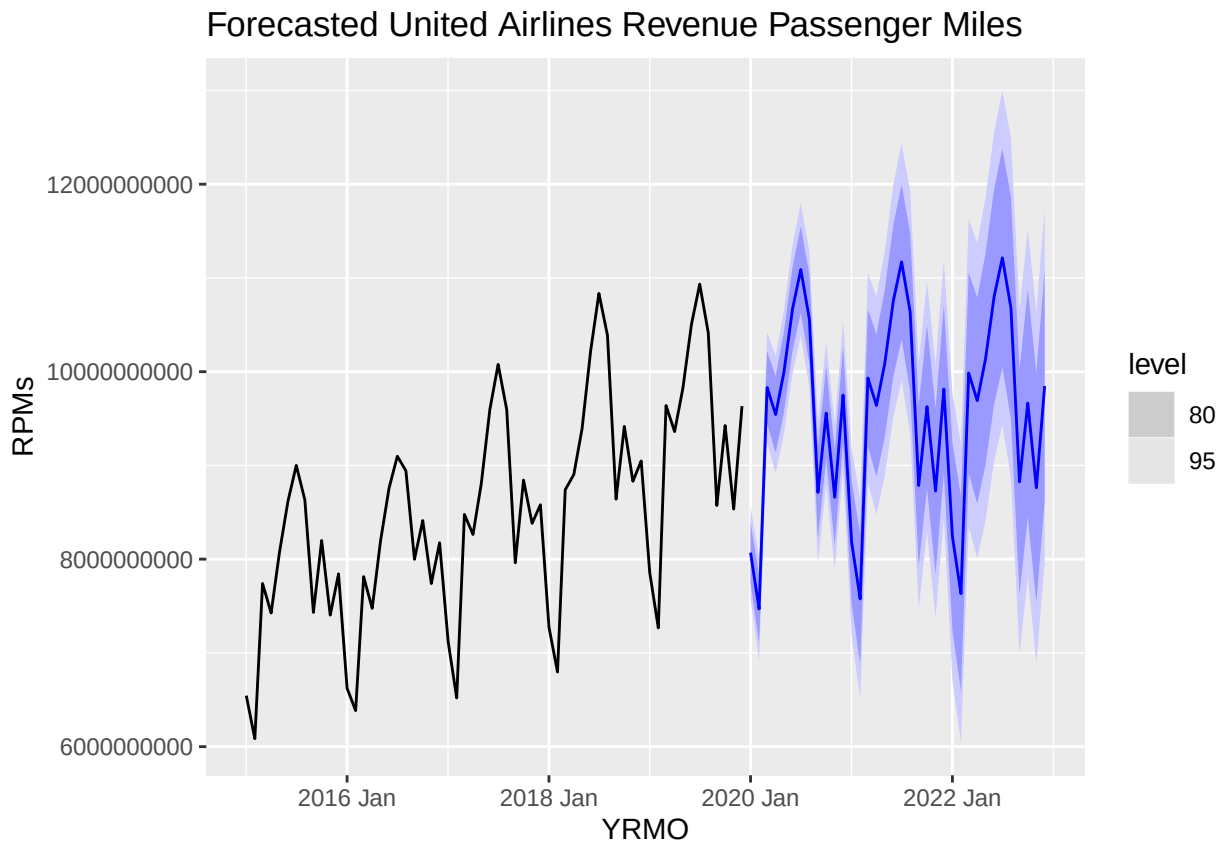
```
augment(fitAA) %>% features(.innov, ljung_box, lag=24, dof=4)
```

```
## # A tibble: 3 x 4
```

```
##   UNIQUE_CARRIER .model      lb_stat lb_pvalue
##   <fct>          <chr>         <dbl>     <dbl>
## 1 AA             AAauto         22.6     0.310
## 2 AA             AAstepwise     23.7     0.255
## 3 AA             ARIMA420520    NA      NA
```

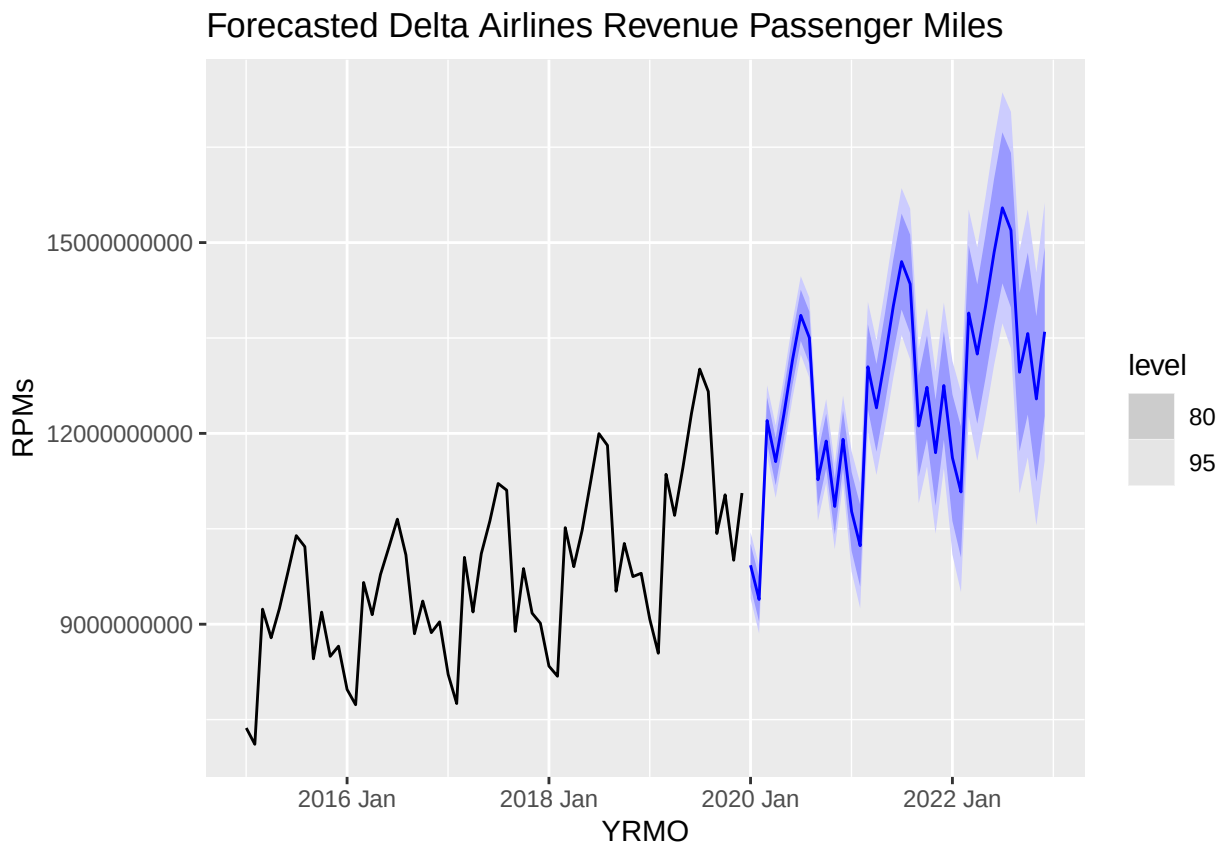
7. Once the residuals look like white noise, calculate forecasts.

```
forecast(fitUA, h=36) %>%
  filter(.model=='UAauto') %>%
  autoplot(UAtravel) +
  labs(title = "Forecasted United Airlines Revenue Passenger Miles",
       y="RPMs")
```

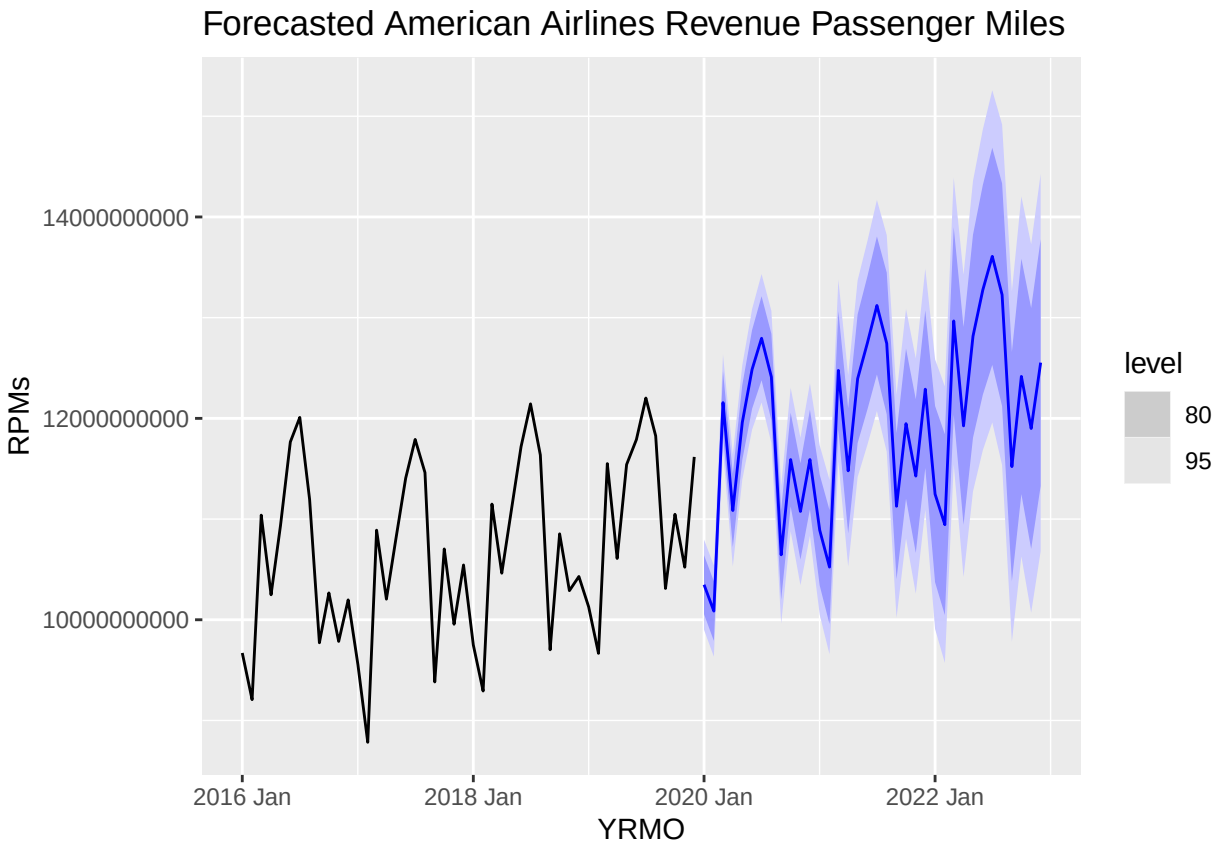



```
forecast(fitDL, h=36) %>%
  filter(.model=='DLauto') %>%
  autoplot(DLtravel) +
  labs(title = "Forecasted Delta Airlines Revenue Passenger Miles",
       y="RPMs")
```

```
## `mutate_if()` ignored the following grouping variables:
## Column `UNIQUE_CARRIER`
```




```
forecast(fitAA, h=36) %>%
  filter(.model=='AAauto') %>%
  autoplot(AAtravel) +
  labs(title = "Forecasted American Airlines Revenue Passenger Miles",
       y="RPMs")
```


Forecasted American Airlines Revenue Passenger Miles
RPMs
14000000000
12000000000
10000000000
2016 Jan
2018 Jan
2020 Jan
2022 Jan
YRMO
level
80
95