Applied Econometrics Assignment 3

Name: Siddarth Gopalakrishnan **ID:** 2017B3A71379H

#Collecting the data and loading necessary libraries

This data is the 'AirPassengers' dataset which is a monthly time-series dataset (univariate) about the monthly totals of international airline passengers in the United States of America, in thousands, from 1949 to 1960.

```
> AirPassengers <- read_excel("E:/STUFF/3-2/ECON F342 ApEc/Assignment 3/AirPassengers.xlsx", col_types = c("blank", "numeric"))
> View(AirPassengers)
```

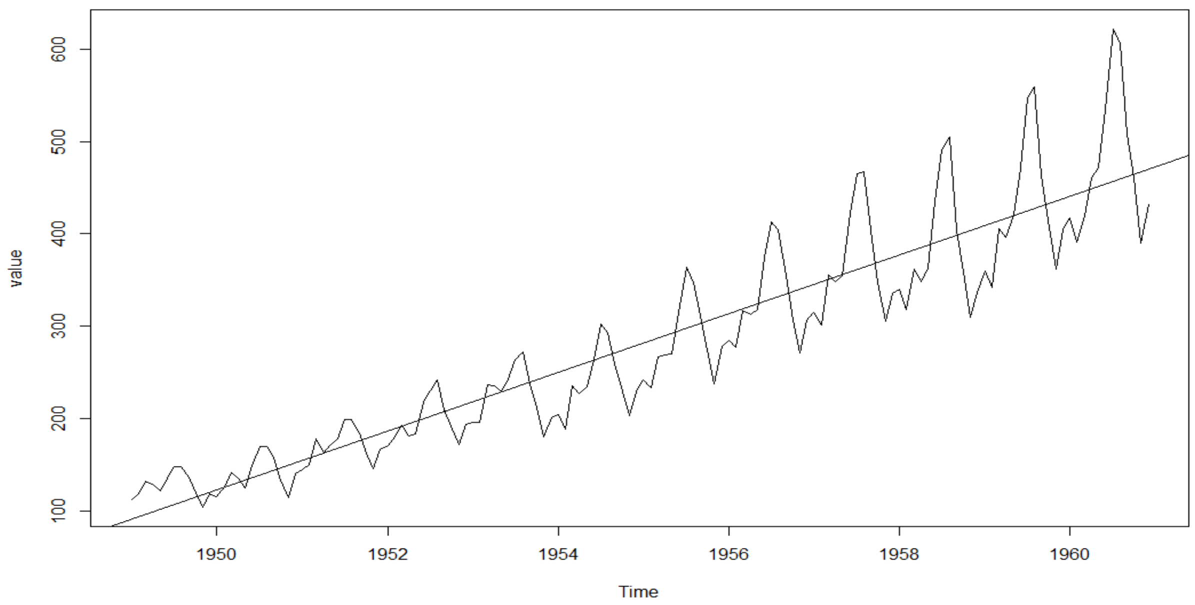
We then load the libraries: 'tseries', 'aTSA', 'forecast', 'urca'.

#Converting data into a time-series

```
> APts = ts(AirPassengers, frequency = 12, start = 1949)
> APts
```

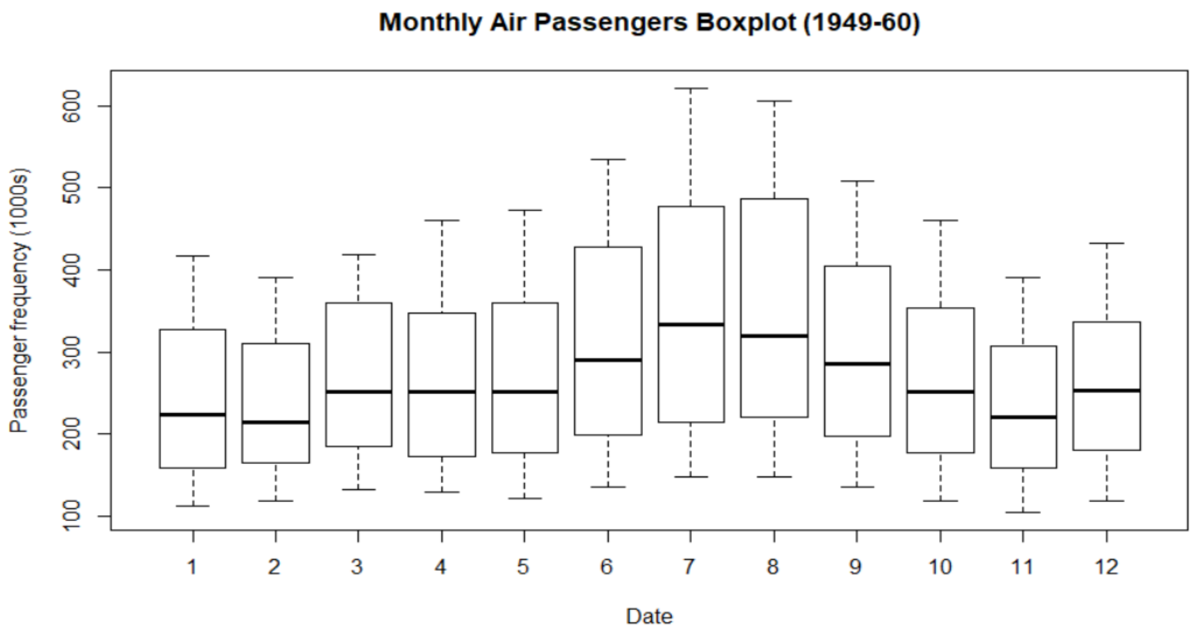
```
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1949 112 118 132 129 121 135 148 148 136 119 104 118
1950 115 126 141 135 125 149 170 170 158 133 114 140
1951 145 150 178 163 172 178 199 199 184 162 146 166
1952 171 180 193 181 183 218 230 242 209 191 172 194
1953 196 196 236 235 229 243 264 272 237 211 180 201
1954 204 188 235 227 234 264 302 293 259 229 203 229
1955 242 233 267 269 270 315 364 347 312 274 237 278
1956 284 277 317 313 318 374 413 405 355 306 271 306
1957 315 301 356 348 355 422 465 467 404 347 305 336
1958 340 318 362 348 363 435 491 505 404 359 310 337
1959 360 342 406 396 420 472 548 559 463 407 362 405
1960 417 391 419 461 472 535 622 606 508 461 390 432
```

```
> plot.ts(APts) # Plotting the time series
> abline(reg = lm(APts ~ time(APts))) # Fitting a trend line
```



From the above plot, we can clearly see that this dataset has a very strong seasonal component and a trend component, hence, we will need to use seasonal differencing. We can also observe that the frequency of airplane travels increases on an average along the years and is highest in the summer. We can confirm this using a box plot.

```
> boxplot(APts ~ cycle(APts), xlab = "Date", ylab = "Passenger frequency (
1000s)", main = "Monthly Air Passengers Boxplot (1949-60)")
```

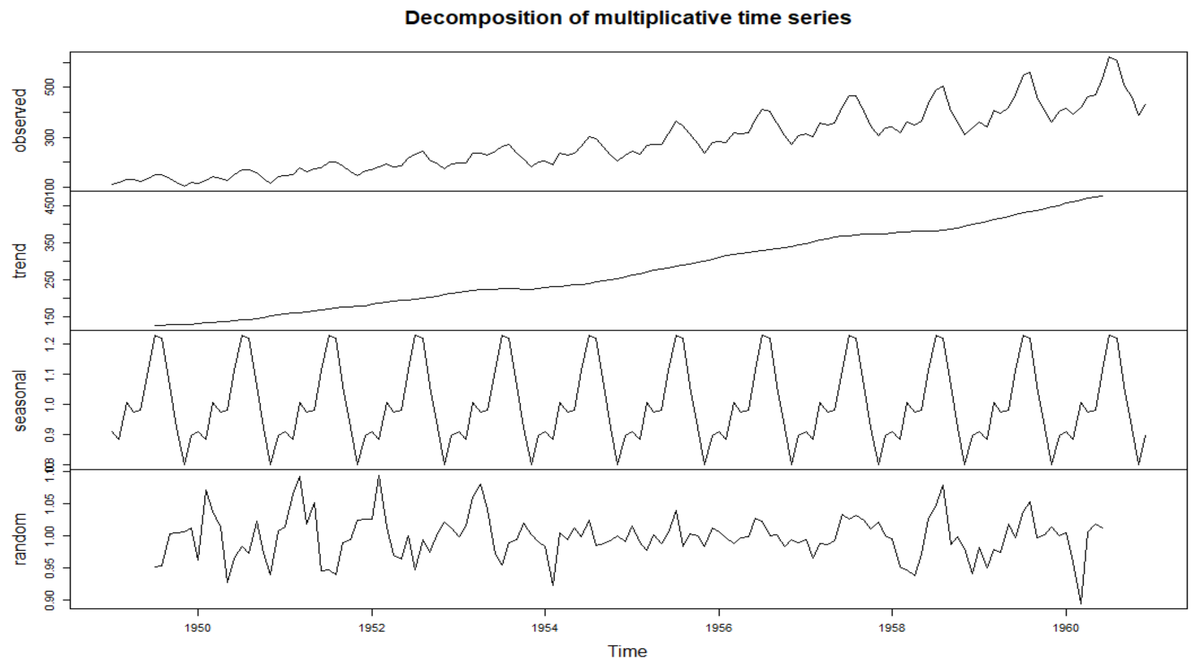


From this Box plot we can observe that the frequency of travels on an average is highest in the month of July (7) which is in fact summer, thus confirming our analysis.

Q1.) a) Decomposing the time-series and adjusting for seasonality

The first and most important thing we can notice from the previous graph is that our data is of course, not stationary and that seasonality increases with the general trend. This indicates that our data might be a multiplicative one rather than an additive one. Thus, performing the decomposition:

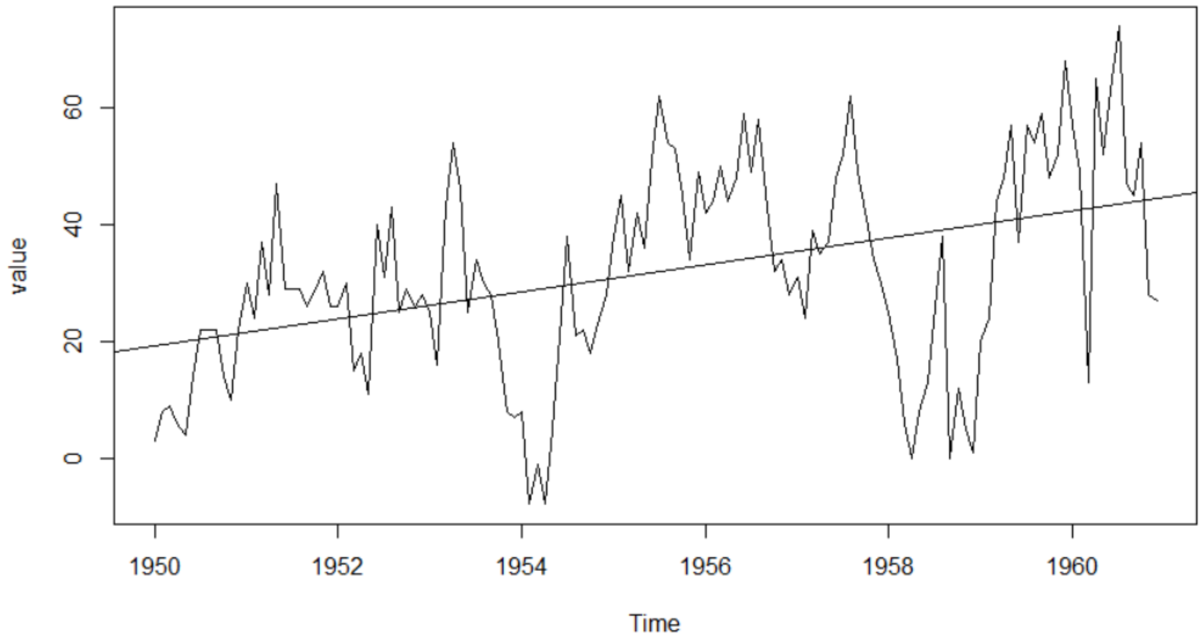
```
> APtsDecomp = decompose(APts, type = "multiplicative")
> plot(APtsDecomp)
```



We can also notice that there is an increasing trend and also that there isn't much randomness in our model. Also, since we already can observe the trend of the model, there really is no need to smoothen our model.

Now in order to remove the stationarity, we use the following command:

```
> APtsadj = diff(APts, lag = 12)
> plot(APtsadj)
> abline(reg = lm(APtsadj ~ time(APtsadj)))
```

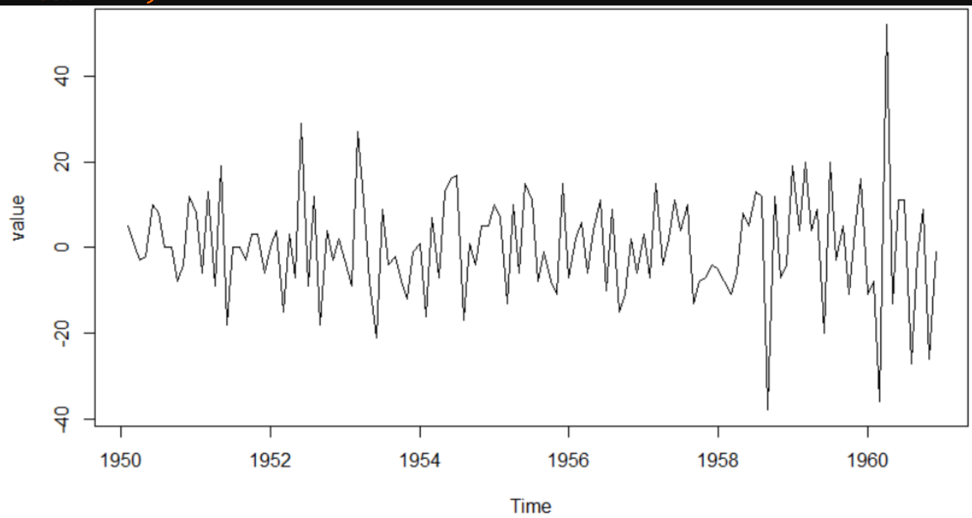


From the above graph we can clearly observe a <u>significant increasing trend</u> implying that the model is not stationary yet and that we might have to difference the series once more in order to homogenize the mean of the series.

Q2.) a) Testing and Adjusting for Stationarity

From the graph plot of our seasonal adjusted time series, we can clearly notice that out time-series is non-stationary as there is clearly still an increasing trend in our series. In order to convert it to a stationary series we have to difference the series to homogenize the mean by taking the difference.

```
> APtsdiff1 = diff(APtsadj, differences = 1)
> plot(APtsdiff1)
```



The above graph is the stationary series we get after the transformation of the original series. We can confirm this with the different stationarity tests:

```
> stationary.test(APtsdiff1, method = “adf”)
> stationary.test(APtsdiff1, method = "pp")
> stationary.test(APtsdiff1, method = "kpss")
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag    ADF p.value
[1,]   0 -15.65    0.01
[2,]   1  -9.01    0.01
[3,]   2  -7.48    0.01
[4,]   3  -7.29    0.01
[5,]   4  -6.01    0.01
Type 2: with drift no trend
     lag    ADF p.value
[1,]   0 -15.60    0.01
[2,]   1  -8.98    0.01
[3,]   2  -7.45    0.01
[4,]   3  -7.27    0.01
[5,]   4  -5.99    0.01
Type 3: with drift and trend
     lag    ADF p.value
[1,]   0 -15.56    0.01
[2,]   1  -8.96    0.01
[3,]   2  -7.44    0.01
[4,]   3  -7.27    0.01
[5,]   4  -5.98    0.01
```

```
Phillips-Perron Unit Root Test
alternative: stationary

Type 1: no drift no trend
 lag Z_rho p.value
   4  -160    0.01
-----
 Type 2: with drift no trend
 lag Z_rho p.value
   4  -160    0.01
-----
 Type 3: with drift and trend
 lag Z_rho p.value
   4  -160    0.01
```

```
KPSS Unit Root Test
alternative: nonstationary

Type 1: no drift no trend
 lag    stat p.value
   2 0.0893     0.1
-----
 Type 2: with drift no trend
 lag    stat p.value
   2 0.0443     0.1
-----
 Type 1: with drift and trend
 lag    stat p.value
   2 0.0269     0.1
```

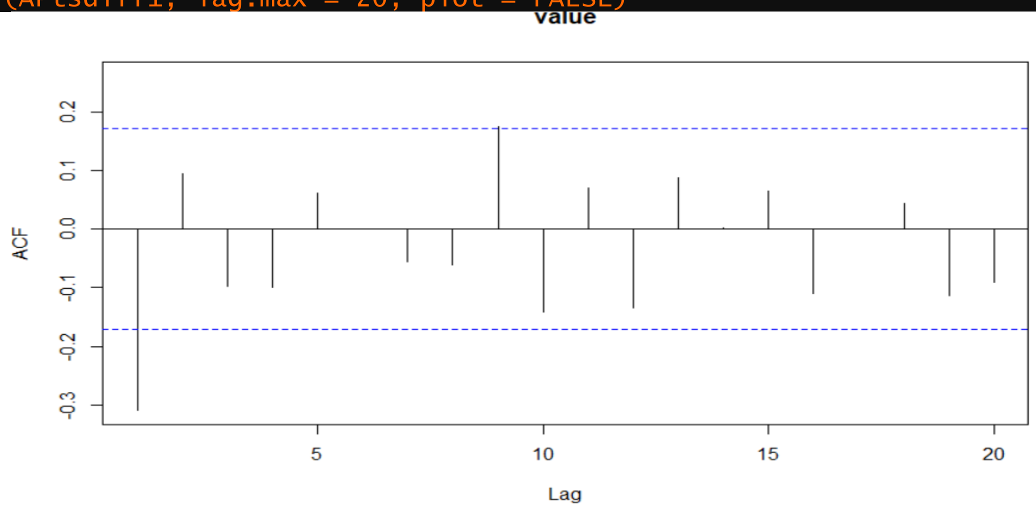
Thus, we can observe that the p-values from the augmented Dickey-Fuller test and Philips-Perron test are lesser than 0.05, due to which we will reject the null hypothesis and accept the alternate hypothesis that out model is stationary. We can confirm this result by the 'kpss' test which has an opposite null and alternate hypothesis as compared to the Philips-Perron test. From the p-values of this test, which are greater than 0.05, we fail to reject the null hypothesis and conclude that the resulting series is stationary. Thus, we would have to fit an ARIMA(p,1,q) model to our dataset, i.e. we have to difference it once.

Q2.) b) Fitting the proper model

Taking the ACF and PACF plots:

(Note: Here we're using the 'Acf' and 'Pacf' functions rather than 'acf' and 'pacf' as they improve the latter functions when applied to univariate time series such as our dataset. The main difference is that they don't plot a spike at lag 0 as it's redundant and the horizontal axis shows lags in time units rather than the seasonal units).

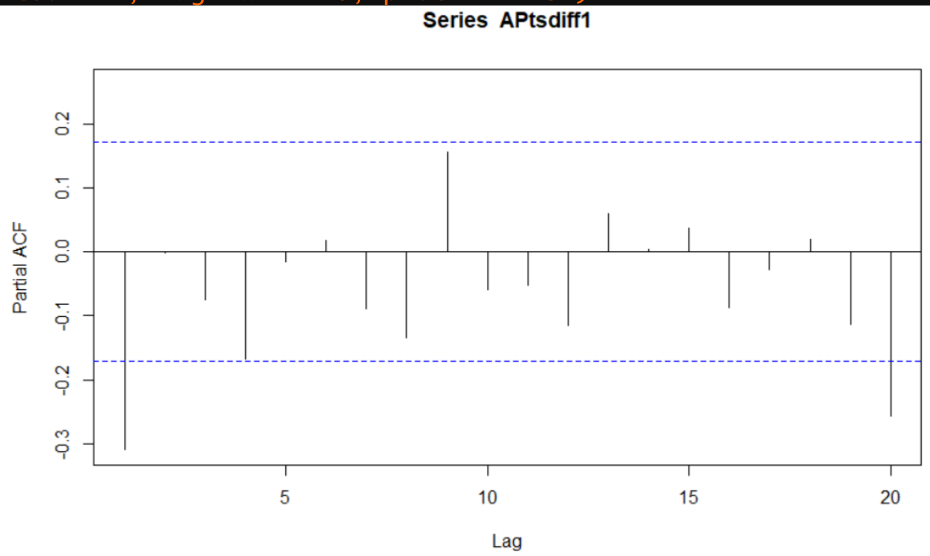
```
> Acf(APtsdiff1, lag.max = 20)
> Acf(APtsdiff1, lag.max = 20, plot = FALSE)
```



```
Autocorrelations of series 'APtsdiff1', by lag

     0      1      2      3      4      5      6      7      8      9
 1.000 -0.310  0.095 -0.097 -0.099  0.061  0.000 -0.056 -0.061  0.176
    10     11     12     13     14     15     16     17     18     19
-0.140  0.070 -0.134  0.087  0.002  0.065 -0.109  0.000  0.044 -0.114
    20
-0.091
```

From the ACF plots and values, we can see that there is only one peak at lag 1 indicating a non-seasonal MA (1) component. Therefore, one possible model could be ARMA(0,1) and we have differenced our series once. (It is important to notice that there is no significant peak at lag 12).

```
> Pacf(APtsdiff1, lag.max = 20)
> Pacf(APtsdiff1, lag.max = 20, plot = FALSE)
```




```
Partial autocorrelations of series 'APtsdiff1', by lag

     1      2      3      4      5      6      7      8      9     10
-0.310 -0.001 -0.075 -0.167 -0.015  0.018 -0.088 -0.134  0.156 -0.059
    11     12     13     14     15     16     17     18     19     20
-0.052 -0.115  0.060  0.004  0.037 -0.087 -0.028  0.019 -0.114 -0.257
```

From the above PACF plot and values we can observe that there is a single peak at lag 1 indicating that this is an AR (1) model. Therefore, another possible model could be ARMA(1,0) and we have differenced our series once. (It is important to notice that there is no significant peak at lag 12).

Therefore, the possible models we could fit are:

1. ARIMA(0, 1, 1), as ACF is insignificant after lag 1 and quickly drops to 0.
2. ARIMA(1, 1, 0), as PACF is insignificant after lag 1 and quickly drops to 0.
3. ARIMA(p, 1, q), a mixed model as both ACF and PACF geometrically tend to 0.

Note that when we fit the appropriate model on our original series (for forecasting), it will be of the form ARIMA(p,1,q) as we have differenced our model once to obtain the stationary series.

Now we apply the auto ARIMA function on the differenced model:

```
> auto.arima(APtsdiff1)
Series: APtsdiff1
ARIMA(2,0,1) with zero mean

Coefficients:
         ar1     ar2      ma1
      0.5960  0.2143  -0.9819
s.e.  0.0888  0.0880   0.0292

sigma^2 estimated as 132.3:  log likelihood=-504.92
AIC=1017.85   AICc=1018.17   BIC=1029.35
```

We can see that the function predicted an ARIMA(2,0,1) model instead of ARIMA(1,0,1) model as estimated from the ACF and PACF plots. This result varies from the one which we predicted using the ACF and PACF plots, because firstly, the ACF and PACF plots aren't the best ways to predict the parameters which best suit our model. Also, the ARIMA model that we fit on our model is on the basis of the information criterion may sometimes yield a different order than the Box-Jenkins method as it does in this case. (On analysing the values of the information criteria for both the models (ARIMA(1,0,1) and ARIMA(2,0,1)) by setting the trace as 'TRUE', we will be able to observe that the difference in their values is very little, implying that both of these models could somewhat similarly estimate our data). Due to this, the predicted ARIMA model from the ACF and PACF plots of the differenced series may not be the most accurate predictions always.

```
> auto.arima(APts)
```

Applying auto ARIMA on our original time-series model, we see the following model fitting best:

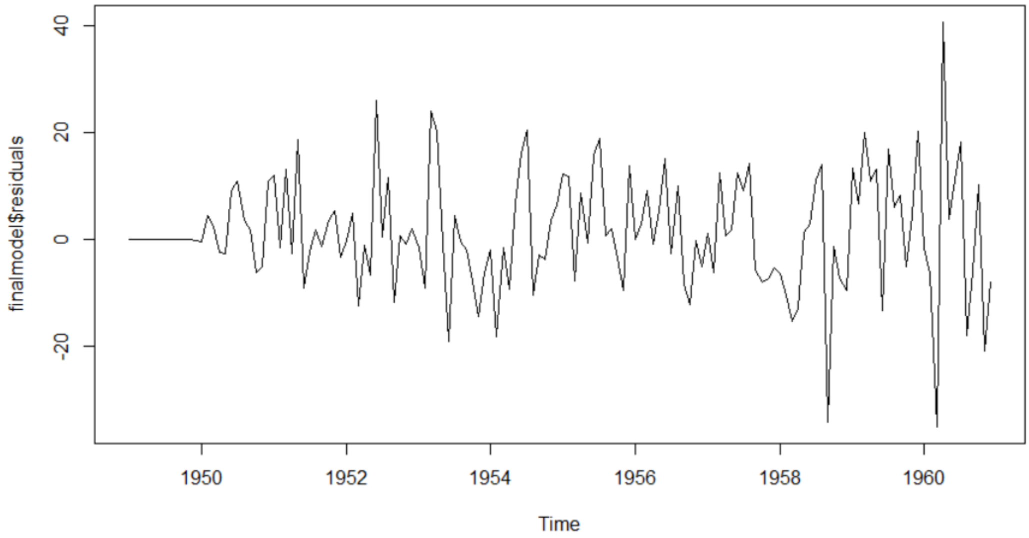
```
Series: APts
ARIMA(2,1,1)(0,1,0)[12]

Coefficients:
         ar1     ar2      ma1
      0.5960  0.2143  -0.9819
s.e.  0.0888  0.0880   0.0292

sigma^2 estimated as 132.3:  log likelihood=-504.92
AIC=1017.85   AICc=1018.17   BIC=1029.35
```

Thus, we see that an ARIMA (2,1,1) fitting our model the best along with a seasonal component of (0,1,0) with a period of 12. We can observe that the 'd' parameter is 1 in this case as we had differenced our model once in order to make it stationary. As our model does have seasonality, as can been observed from the initial graph of the time-series, we can see that the predicted model on the original series has a seasonality component as well. As pointed out while analysing the ACF and PACF plots, since there are no peaks at lag 12 (which is the period of our dataset (monthly)) in both the ACF as well as the PACF plots, the 'p' and 'q' parameters for the seasonal components are 0 and as expected the 'd' parameter is 1.

Q2.) c) Making forecasts using our fitted model

We will, therefore, fit the new model to our original dataset in order to make predictions

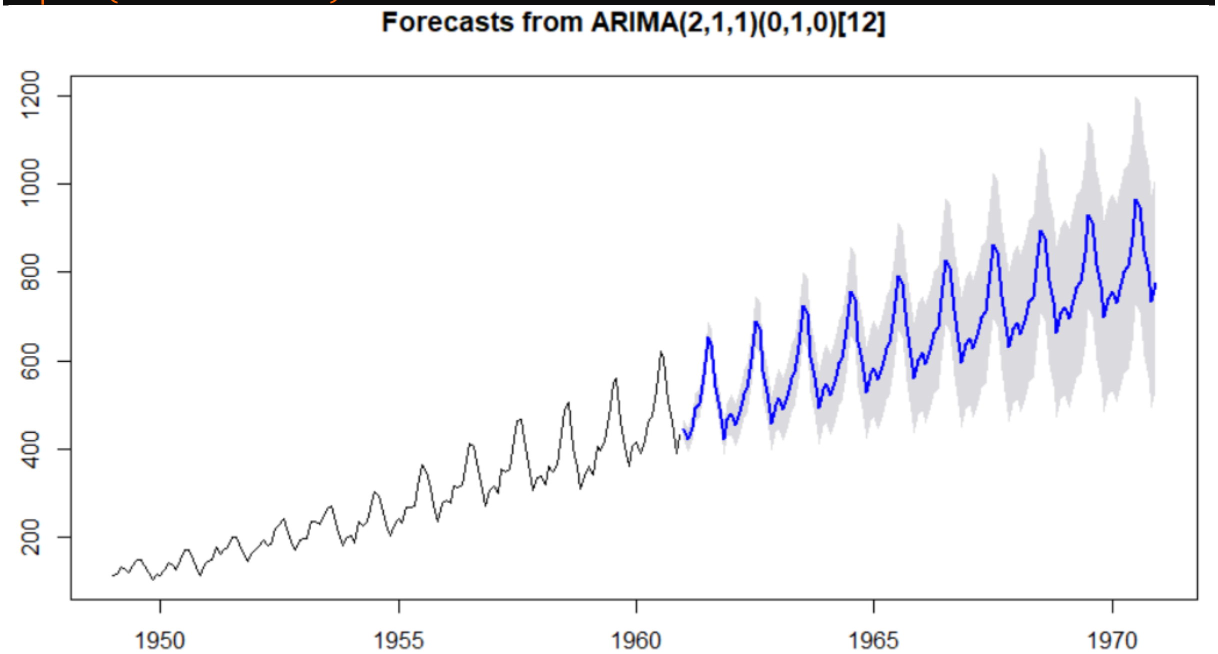
```
> finalmodel = Arima(APts, order=c(2,1,1), seasonal=list(order=c(0,1,0), pe
riod=12))
> plot.ts(finalmodel$residuals)
```



We can notice that the residuals have a mean value of 0. Now we make forecast for the next 10 years with 95% confidence level:

(We have taken h = 10*12 as our data has a monthly period)

```
> modelforecast = forecast(finalmodel, level=c(95), h=10*12)
> modelforecast
> plot(modelforecast)
```



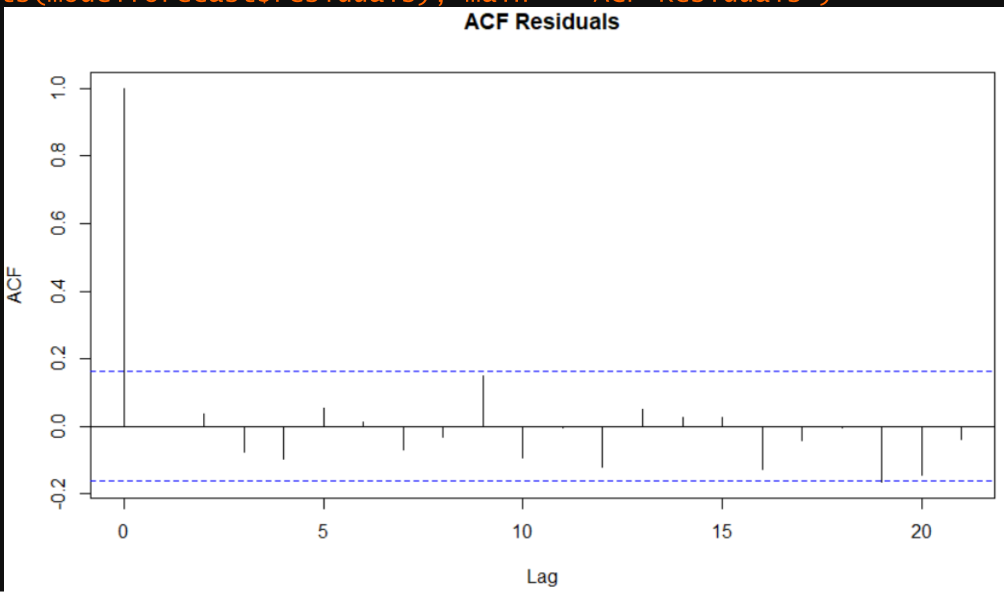
Thus, by making the forecast for the next ten years and observing the plot of the forecast, we can observe that our model has learnt the pattern very well from our previous data and is observed to make quite accurate predictions for the next ten years.

Our forecast rightly predicts the increase in frequency of airplane travels in the summer and also correctly predicts the increasing trend as the number of people travelling via airplanes are naturally expected to increase in the future. (Please do take a look at the values of the 'modelforecast' data in the R Workspace in order to get a clearer picture of the point forecasts).

Q2.) d) Testing validity of the forecast

In order to test the validity of the forecast, we first have to see if there are any significant auto-correlations:

```
> acf(ts(modelforecast$residuals), main = "ACF Residuals")
```



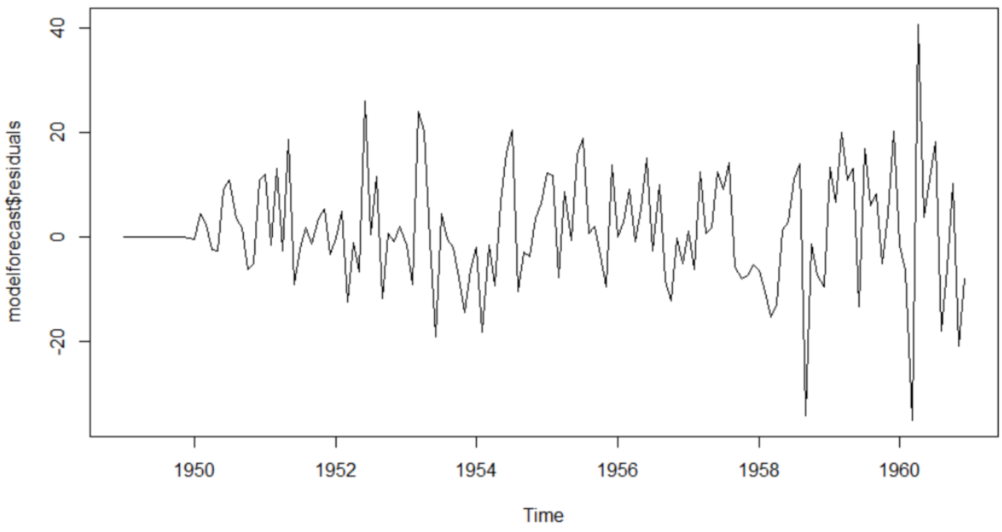
Thus, we can observe that there is no significant auto-correlations, which can also be confirmed by the Ljung-Box test:

```
> Box.test(modelforecast$residuals, lag=10, type="Ljung-Box")

	Box-Ljung test

data:  modelforecast$residuals
X-squared = 8.6878, df = 10, p-value = 0.562
```

We can see that the p-value is greater than 0.05 which implies that we fail to reject the null hypothesis that there is no significant autocorrelation. From ACF & Box-Ljung test, we conclude that there is very little evidence for non-zero autocorrelations in the forecast errors at lags 1-20. Thus, our model is fairly accurate.

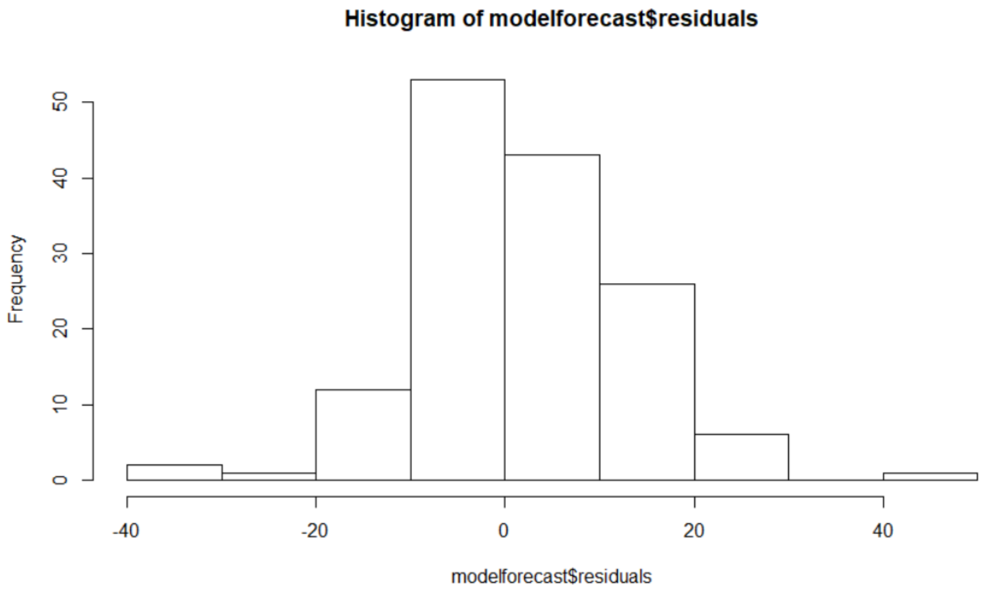
Now to check for normal distribution with zero mean and constant variance, we plot the residuals of the forecasts and its histogram.

```
> plot.ts(modelforecast$residuals) # make time plot of forecast errors
> hist(modelforecast$residuals) # make a histogram
```



We can see that the distribution of the residuals has zero mean.

Now we analyse the histogram to check whether the forecast residuals follow a normal distribution.



Thus, we can see that our data is in fact fairly normally distributed around mean 0, due to which we can conclude that our model is a fairly accurate representation fitting the given dataset.