



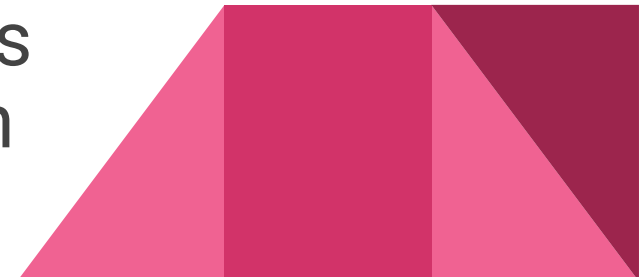
R PROJECT– DATA ANALYTICS

TOPIC: MONTHLY AIRLINE PASSENGERS FROM 1949 TO 1960

US airline passengers from 1949 to 1960

Introduction:

- The classic Box & Jenkins airline data. Monthly totals of international airline passengers, 1949 to 1960
- This dataset is taken from an inbuilt dataset of R called AirPassengers
- This dataset provides monthly totals of a US airline passengers from 1949 to 1960. Total number of passengers travelled on that particular month



Dataset of AirPassengers:

```
In [1]: data(AirPassengers)
```

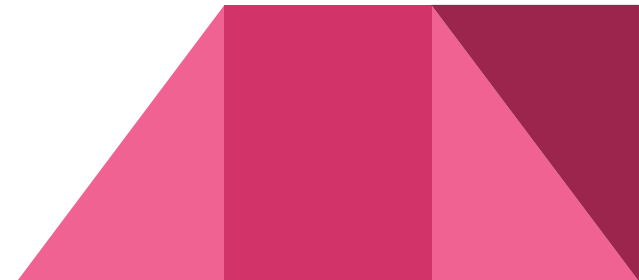
```
In [2]: dataSet<-AirPassengers
```

```
In [3]: dataSet
```

	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

Time series Analysis :

- ❏ A time-series is a set of observations on a quantitative variable collected over time.
- ❏ **Example :** Historical data on sales, inventory, customer counts, interest rates, costs, etc.
- ❏ Businesses are often very interested in forecasting time series variables.
- ❏ Often, independent variables are not available to build a regression model of a time series variable.
- ❏ In time series analysis, we analyze the past behavior of a variable in order to predict its future behavior.



Basic structure and nature of dataset

- The data is basically a time series dataset where it starts from January of 1949 and ends at December of 1960.
- The frequency which is the total number of months is 12.

```
In [4]: #The below command is to know about the basic structure in data, also to know nature of this dataset
class(dataSet) #ts - time series
'ts'
```

```
In [5]: str(dataSet) #it gives structure of a dataset
Time-Series [1:144] from 1949 to 1961: 112 118 132 129 121 135 148 148 136 119 ...
```

```
In [6]: attributes(dataSet)
$ts
1949 1960.916666666667 12
$class
'ts'
```

```
In [7]: start(dataSet)
1949 1
```

```
In [8]: end(dataSet)
1960 12
```

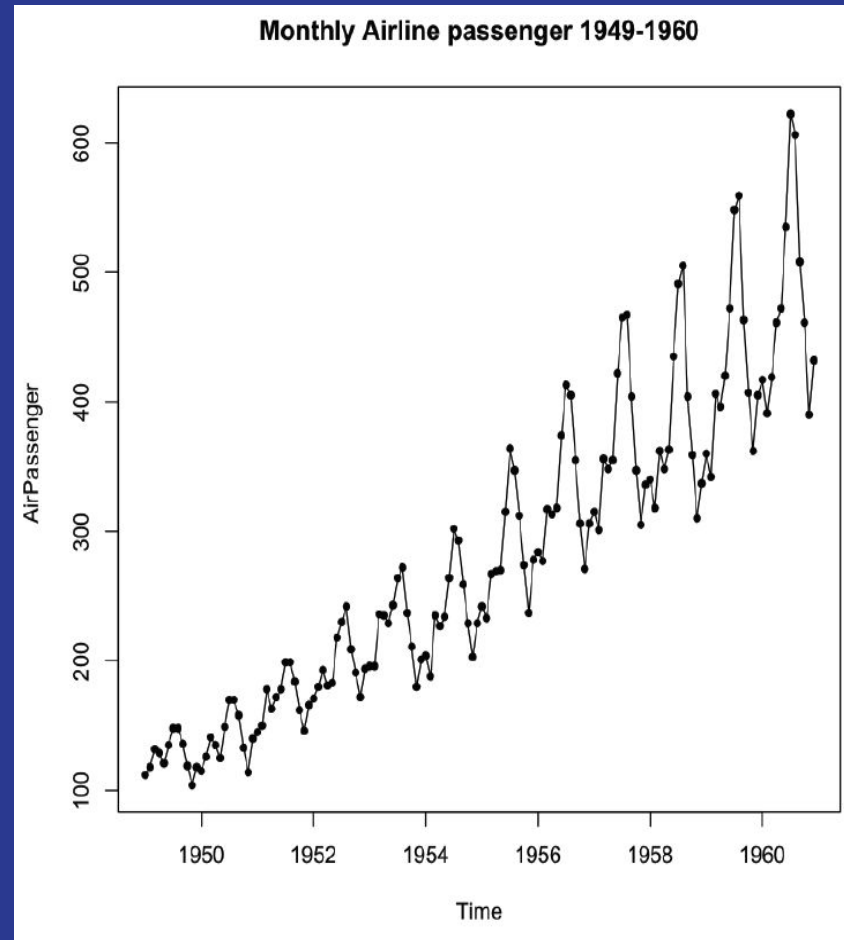
```
In [9]: mode(dataSet)
'numeric'
```

```
In [12]: plot(dataSet,ylab="AirPassenger ",type="o",pch = 20,main = "Monthly Airline passenger 1949-1960")
```

□ This plot will give a trend line, the plots are called as spark lines...ex: in stock market, if we put a trend line

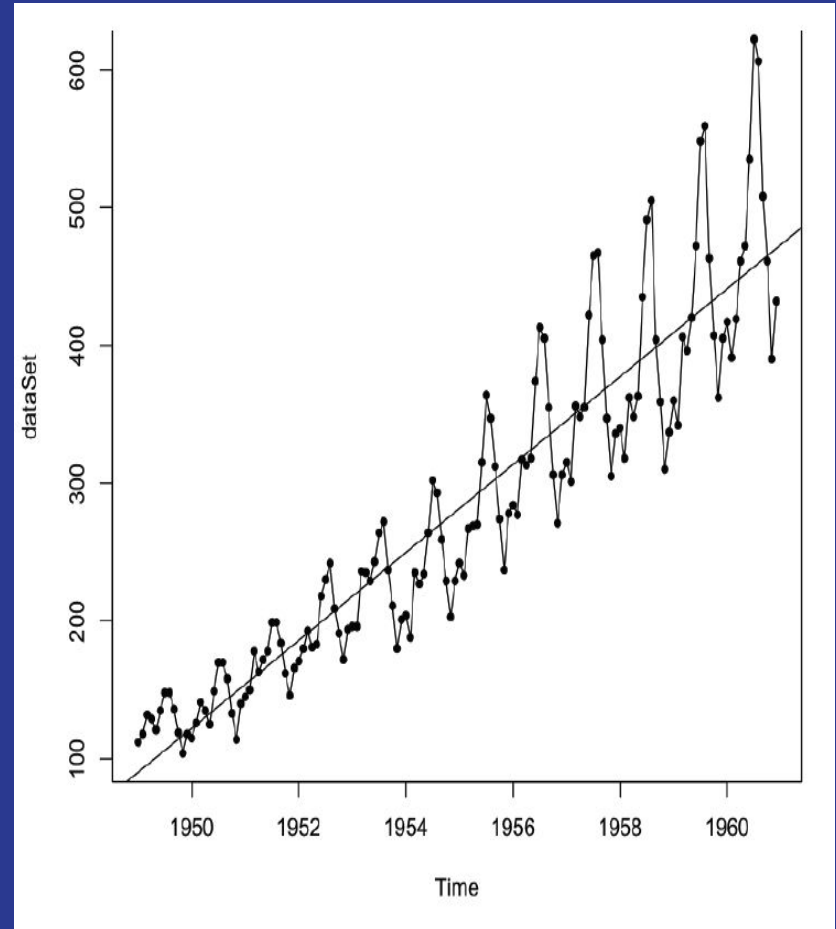
Observations :

1. There is a trend component which grows the passenger year by year.
2. There looks to be a seasonal component which has a cycle less than 12 months.
3. The variance in the data keeps on increasing with time.

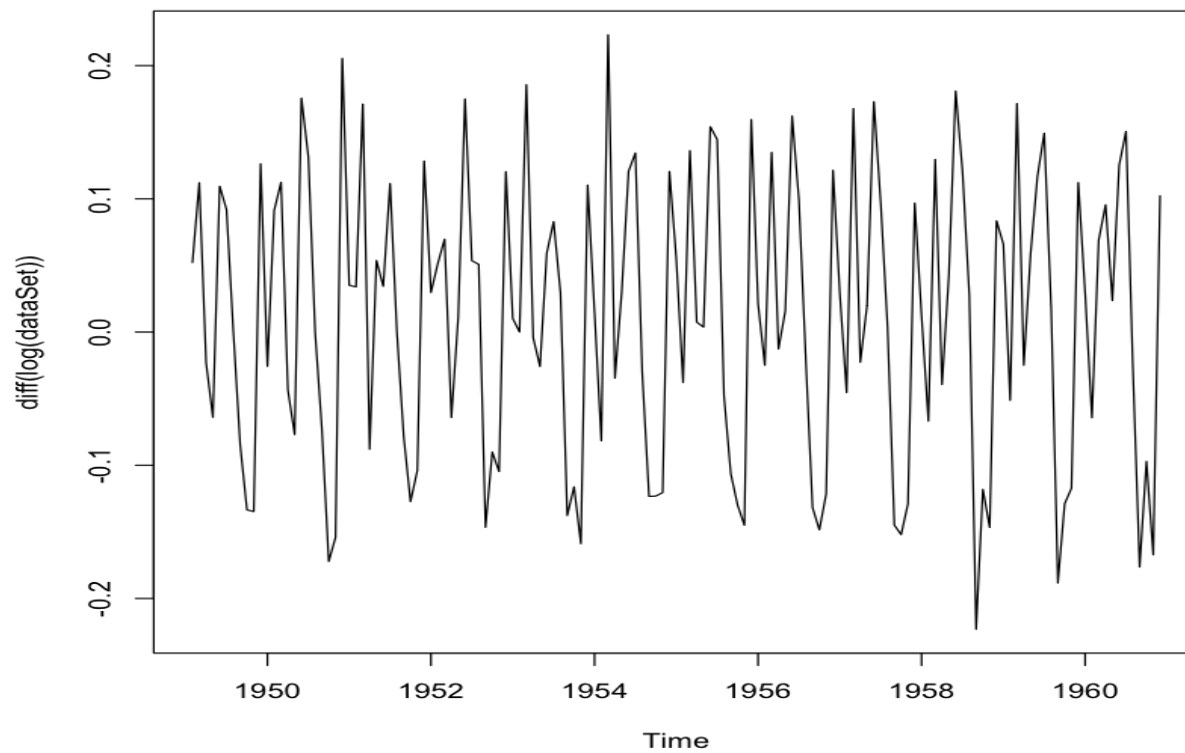


```
In [13]: # reg=regression model,lm=linear modeling
plot(dataSet,type="o",pch=20)
abline(reg = lm(dataSet~time(dataSet)))
```

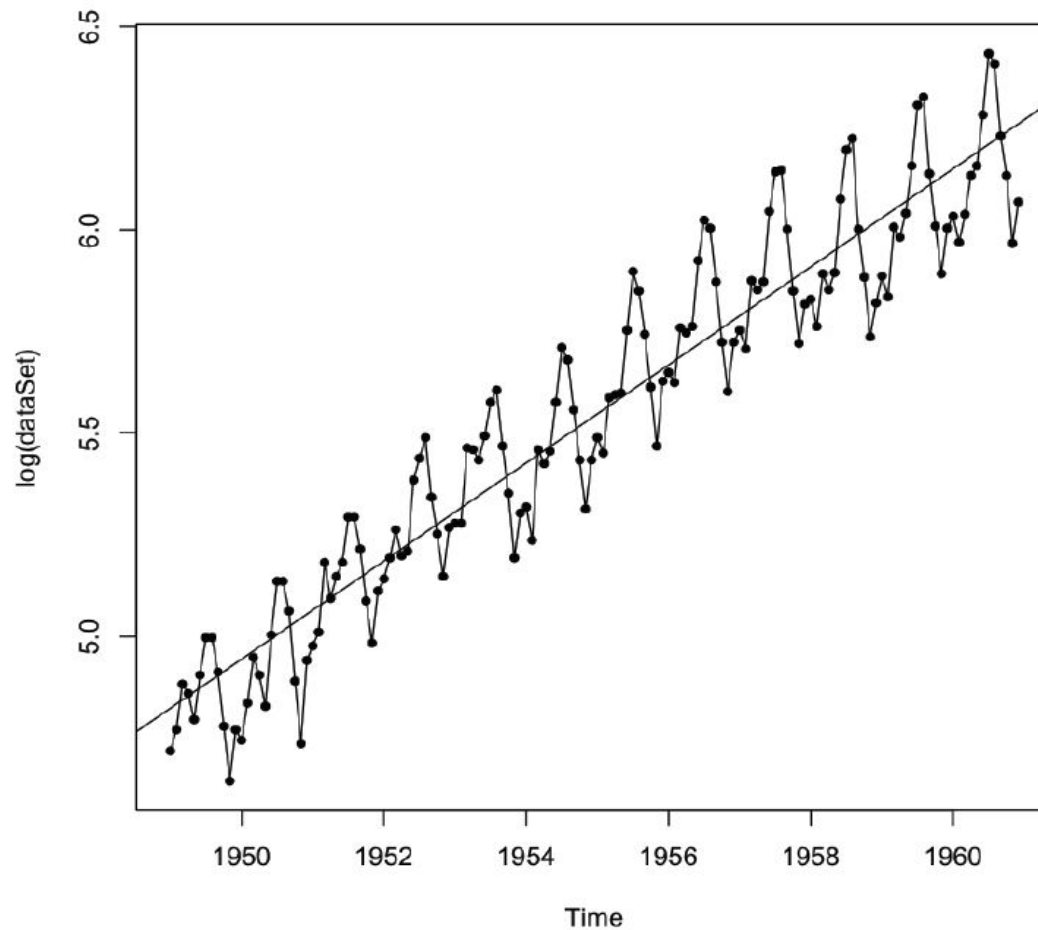
- In this plot representation, we have put a trend line i.e., connecting means of all the points.
- Here, the variance changes at every point.
- The data is not standard or stationary, so we can't apply Time Series to it.
- To make the data as stationary, the variance should be equal.
 1. So, first we need to remove unequal variances. We can do this by using log of the series.
 2. Secondly, we need to address the trend component. We do this by taking difference of the series. Now, let's test the resultant series.



```
In [15]: plot(diff(log(dataSet)))
```




```
In [16]: plot(log(dataSet),type="o",pch=20)  
         abline(reg=lm(log(dataSet)~time(dataSet)))
```



Components of TSA :

❑ **Cycle :**

- An up-and-down repetitive movement in demand.
- repeats itself over a long period of time.

❑ **Seasonal Variation :**

- An up-and-down repetitive movement within a trend occurring periodically.
- Often weather related but could be daily or weekly occurrence

❑ **Random Variations :**

Erratic movements that are not predictable because they do not follow a pattern

Interpretation:

Seasonality appears to increase with the general trend suggesting a multiplicative model rather than an additive model, i.e.

$$Y(t) = T(t) * S(t) * e(t)$$

where,

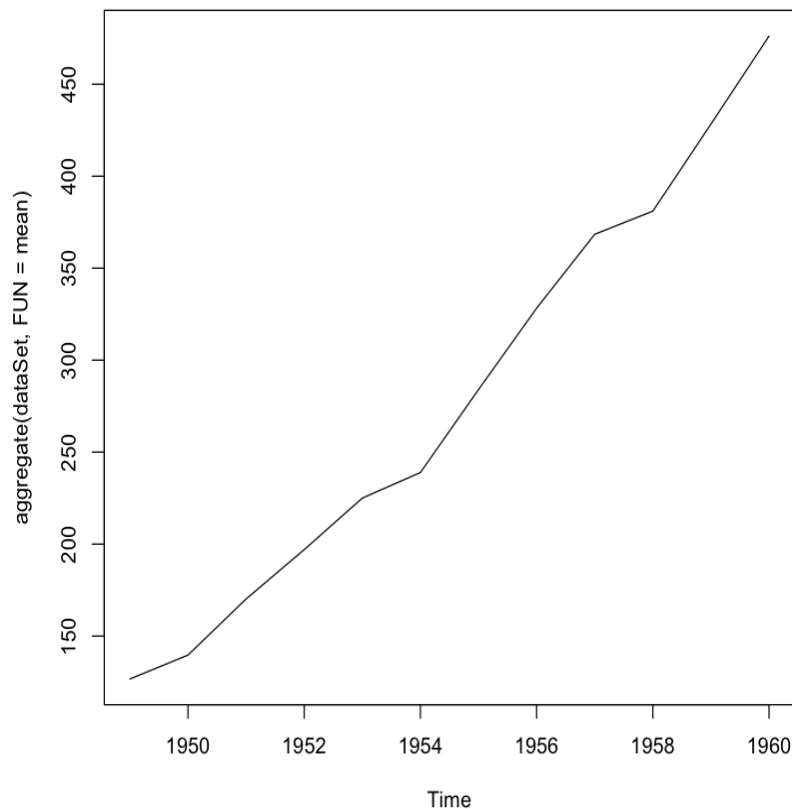
* $Y(t)$ is the number of passengers at time t

* $T(t)$ is the trend component at time t

* $S(t)$ is the seasonal component at time t

* $e(t)$ is the random variation at time t

```
In [21]: #plot an year on year general trend  
plot(aggregate(dataSet,FUN = mean))
```



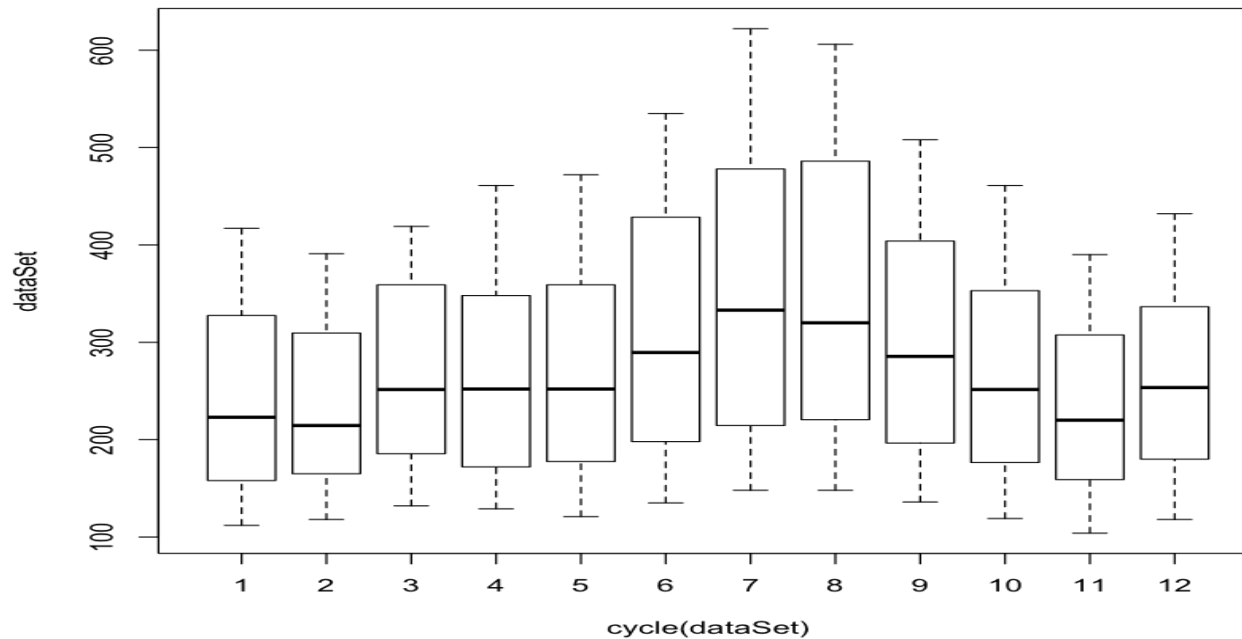
Decomposing the Data

Plot on year on year general trend

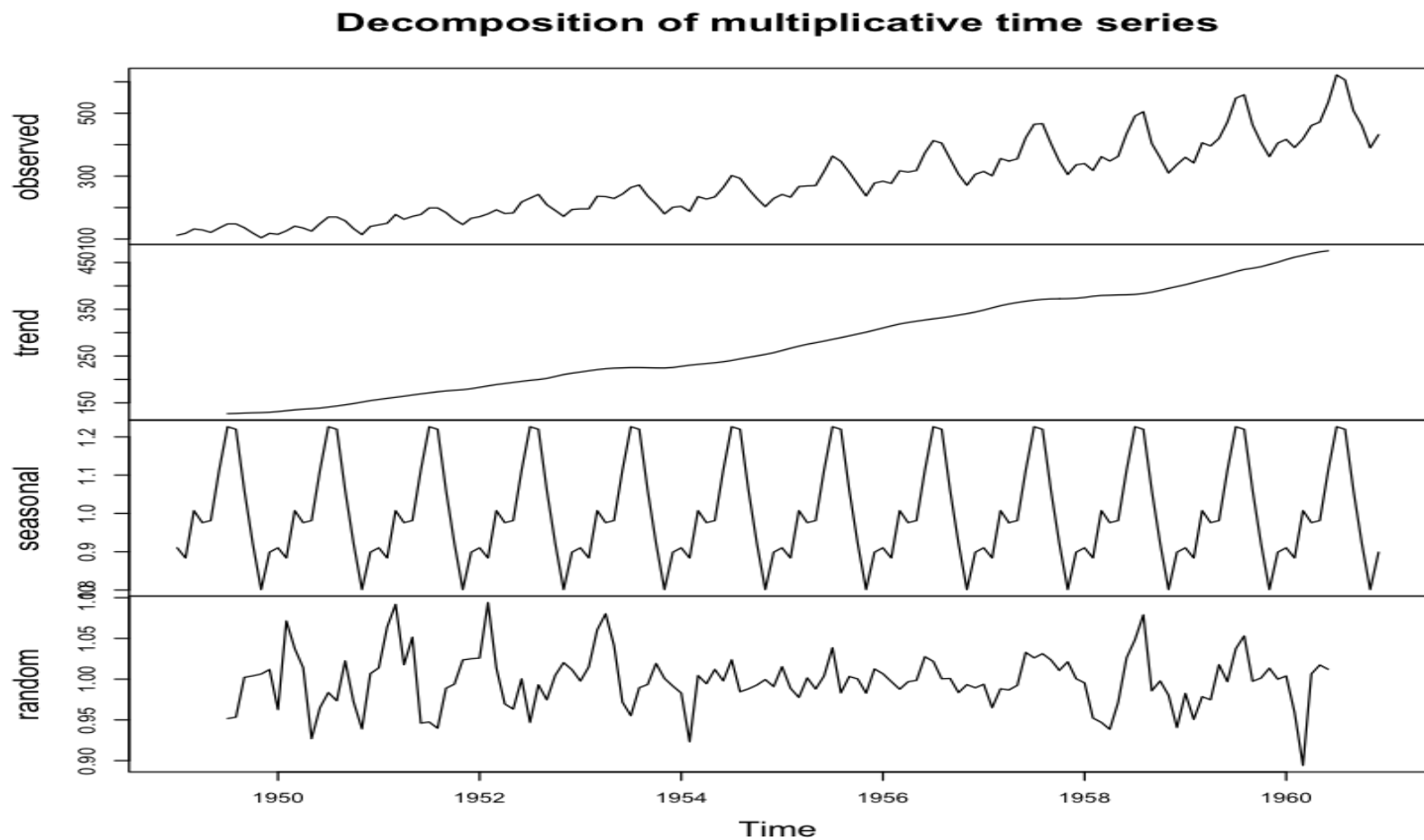
Decomposing the data into its trend, seasonal and random error components will give some idea how these components relate to the observed dataset.

Box plot by cycle

```
In [22]: boxplot(dataSet~cycle(dataSet))
```

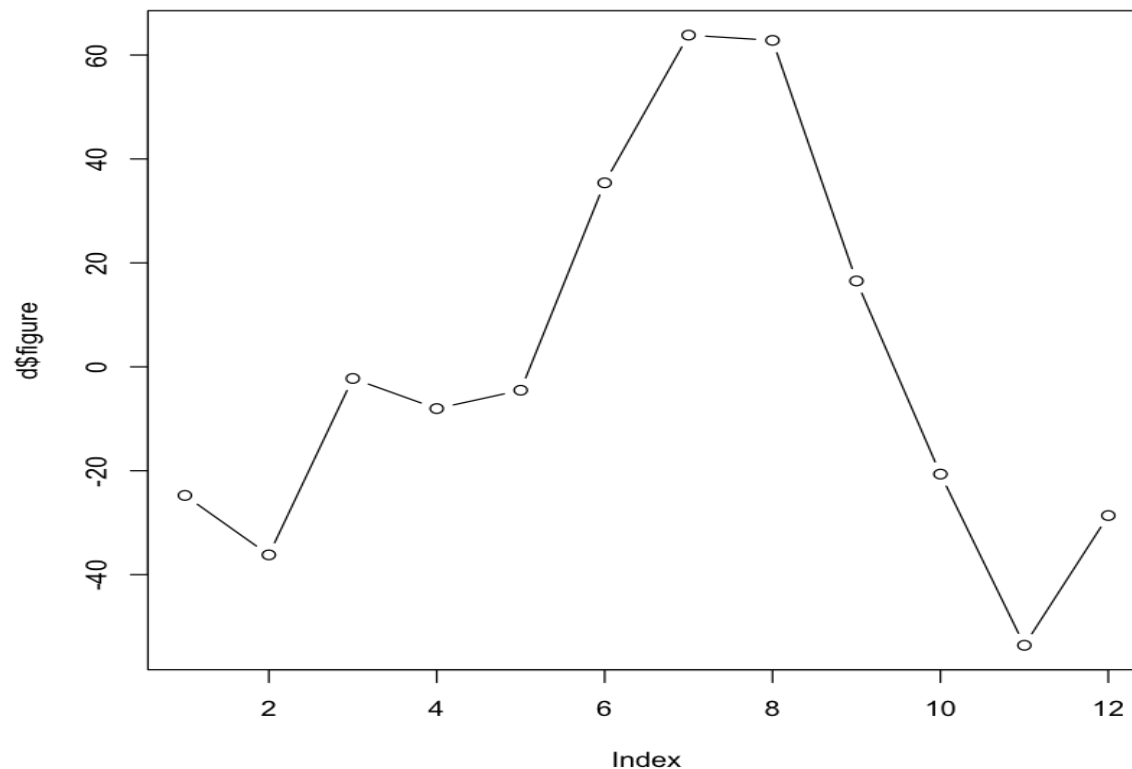


```
In [24]: plot(decompose(dataSet,type="multiplicative"))
```



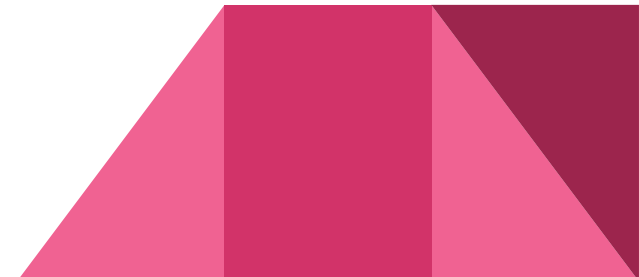
Seasonal figures

```
In [25]: a<-ts(dataSet,frequency = 12)  
d<-decompose(a)  
plot(d$figure,type="b") #seasonal figures
```



Time Series Forecasting:

- ❑ Regression Analysis
- ❑ Time Series Analysis (TSA)
 - A statistical technique that uses time- series data for explaining the past or forecasting future events.
 - The prediction is a function of time (days, months, years, etc.)
 - No causal variable; examine past behavior of a variable and and attempt to predict future behaviour



To forecast future events based on known past data
for example, to predict a price of a stock on a past performance their are two popular model:

-
- ARMA (Autoregressive moving average)
- ARIMA (Autoregressive integrated moving average)
- first we have upload a library called “tseries”
- library(tseries)

```
In [26]: library(tseries)
```

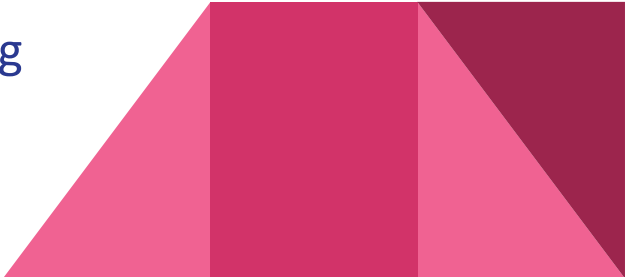


We see that the series is stationary enough to do any kind of time series modelling.

```
In [27]: adf.test(diff(log(AirPassengers)), alternative="stationary", k=0)
```

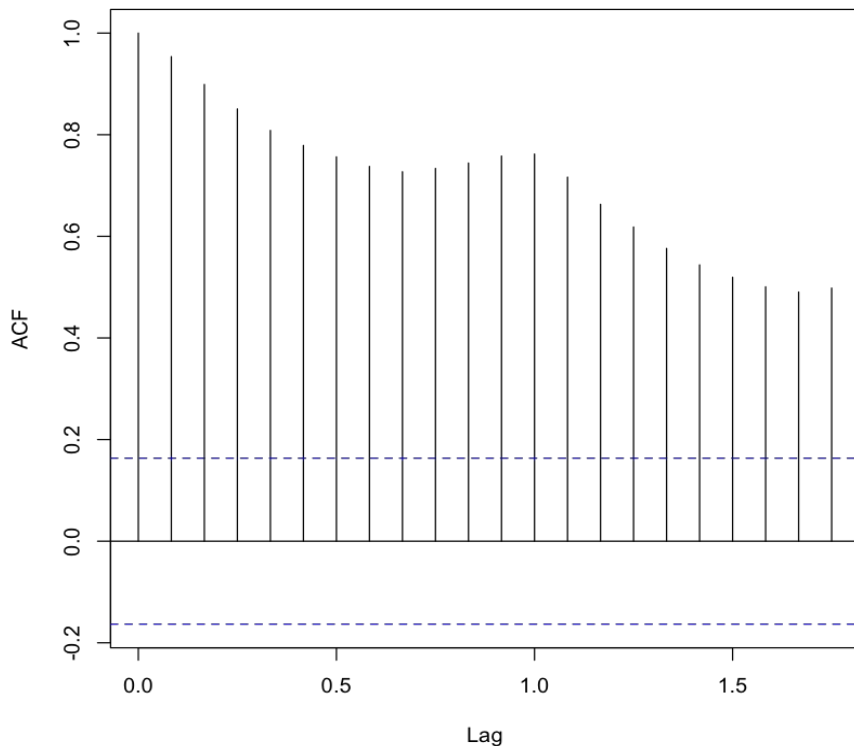
```
data: diff(log(AirPassengers)) Dickey-Fuller =  
-9.6003, Lag order = 0, p-value = 0.01  
alternative hypothesis: stationary
```

Now, to find the right parameters we use ARIMA model. We already know that the 'd' component is 1 as we need 1 difference to make the series stationary. We do this using the Correlation plots. Following are the ACF plots for the series



```
In [28]: # the function acf() compute an estimate of autocorrelation of time series.  
         acf(log(dataSet))
```

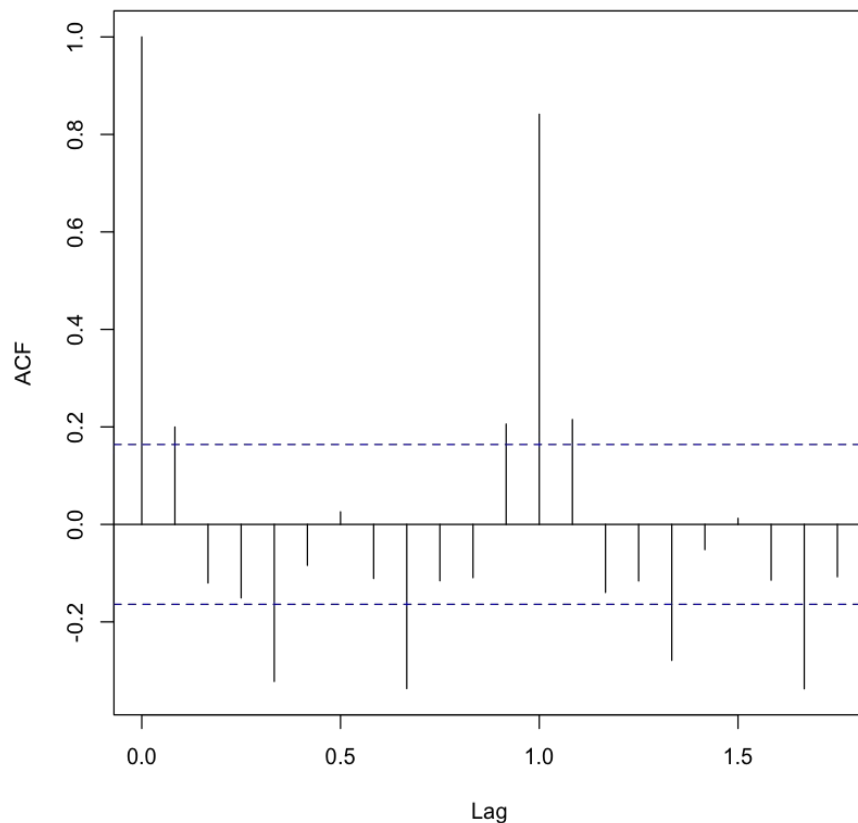
Series log(dataSet)



- By above graph we see that, the decay of ACF chart is very slow, which means that the population is not stationary.
- We have already discussed above that we now intend to regress on the difference of log rather than log directly.
- Let's see how ACF and PACF curve come out after regressing on the difference
 - AR I MA (autoregression integrated moving average)
 - p , d , q (value to be included, to autocorrelation function graph)

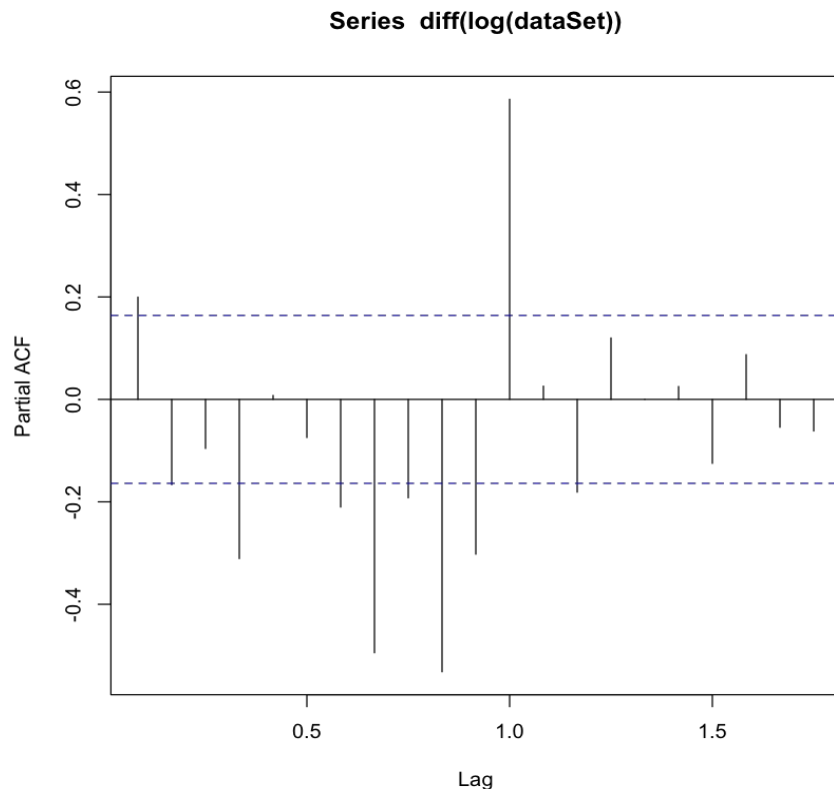
```
In [29]: acf(diff(log(dataSet))) # determine the q value i.e, q=1
```

Series diff(log(dataSet))



- The function `acf()` compute an estimate of autocorrelation of time series.
- Therefore the q value is q=1

```
In [30]: #the pacf compute an estimate of the partial autocorrelation  
#function of a (possibly multivariate) time series.  
pacf(diff(log(dataSet))) #determine the p value i.e, p=0
```



- The pacf compute an estimate of the partial autocorrelation function of a (possibly multivariate) time series.
- Therefore the p value is $p=0$

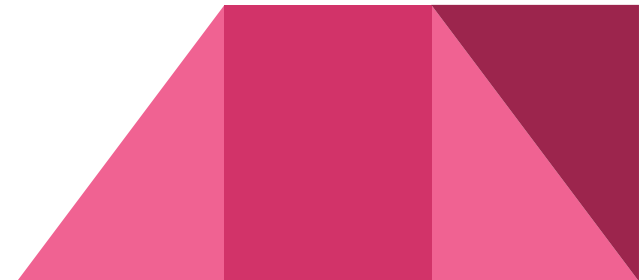
□ The d value comes from number of times you do the differentiate. In this case $d=1$.

□ From the above plot, ACF plot cuts off after the first lag. Hence, we understood that value of p should be 0 as the ACF is the curve getting a cut off.

□ While value of q should be 1 or 2.

After a few iterations, we found that $(0,1,1)$ as (p, d, q) comes out to be the combination with least AIC and BIC.

□ Let's fit an ARIMA model and predict the future 10 years. Also, we will try fitting in a seasonal component in the ARIMA formulation. Then, we will visualize the prediction along with the training data. $c(p, d, q)=c(0,1,1)$



```
In [31]: fitArima=arima(log(dataSet),c(0,1,1),seasonal = list(order=c(0,1,1),period=12))
```

```
In [32]: fitArima
```

Fit Arima:

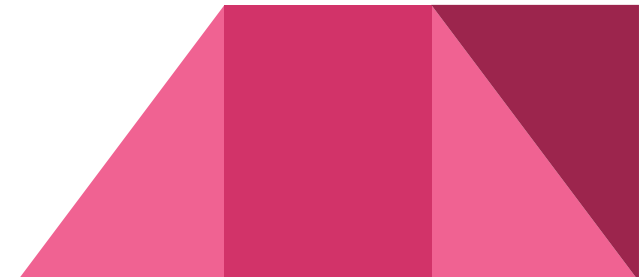
arima(x = log(dataSet), order = c(0, 1, 1), seasonal = list(order = c(0, 1, 1), period = 12))

Coefficients:

	<i>ma1</i>	<i>sma1</i>
	-0.4018	-0.5569

<i>s.e.</i>	0.0896	0.0731
-------------	--------	--------

sigma^2 estimated as 0.001348: log likelihood = 244.7 , aic = -483.4



□ The prediction values are in logarithmic form.

□ To convert it into a original form we need to transform the e value i.e. 2.718

□ After converting to original form the values will be as follow. These are the predicted values for ahead of 10 years.

```
In [35]: pred <- predict(fitArima, n.ahead = 10*12)
```

```
In [36]: # the prediction values are in logarithmic form
# to convert it into a original form we need to transform the e value i.e, 2.718

pred
```

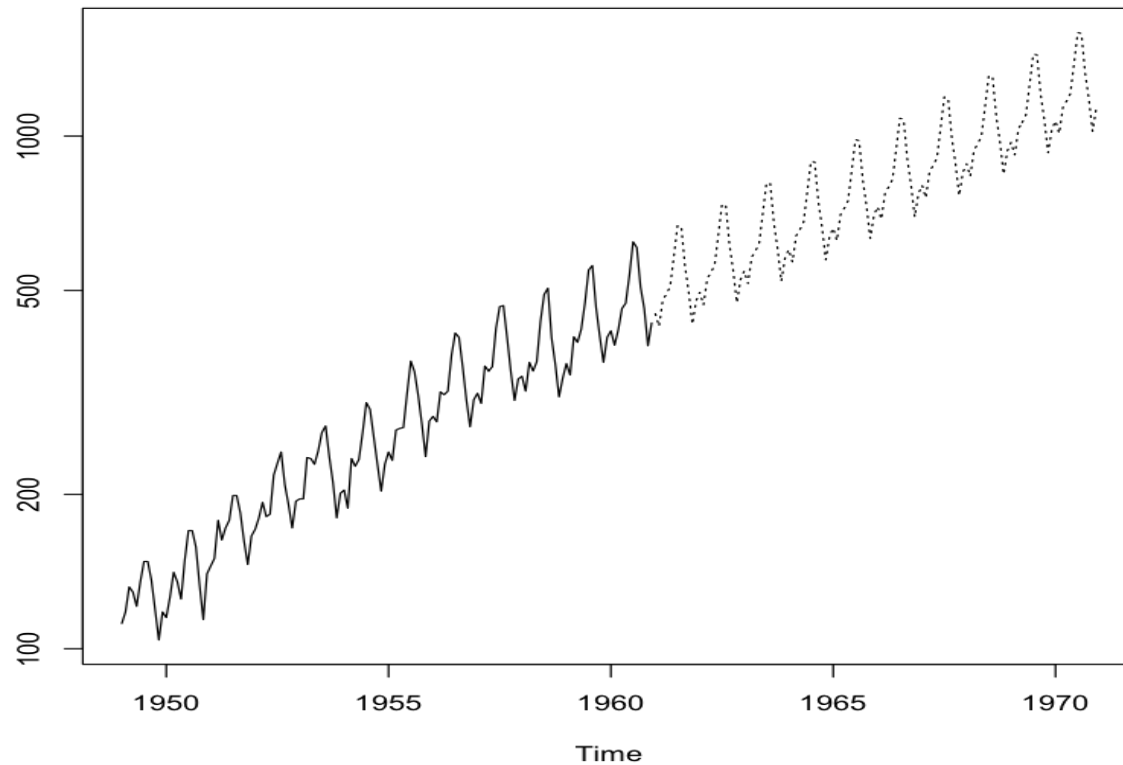
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1961	450	425	479	492	509	583	670	667	558	497	430	477
1962	496	468	527	542	560	642	737	734	614	547	473	525
1963	546	516	580	597	617	707	812	808	676	602	521	578
1964	601	568	639	657	679	778	894	890	745	663	573	637
1965	661	625	703	723	748	857	984	980	820	730	631	701
1966	728	688	775	796	823	943	1083	1079	903	804	695	772
1967	802	758	853	877	906	1039	1193	1188	994	885	765	850
1968	883	834	939	965	998	1143	1313	1308	1094	975	843	935
1969	972	919	1034	1063	1099	1259	1446	1440	1205	1073	928	1030
1970	1070	1012	1138	1170	1210	1386	1592	1585	1326	1181	1021	1134

Graph plot of original data and predicted data.

```
In [43]: a<-round(2.718^pred$pred,0)
```

```
In [44]: a
```

```
In [45]: ts.plot(AirPassengers,2.718^pred$pred, log = "y", lty = c(1,3))
```



Testing of model:

```
In [46]: #testing of model  
datawide<-ts(dataSet,frequency = 12,start = c(1949,1),end=c(1959,12))
```

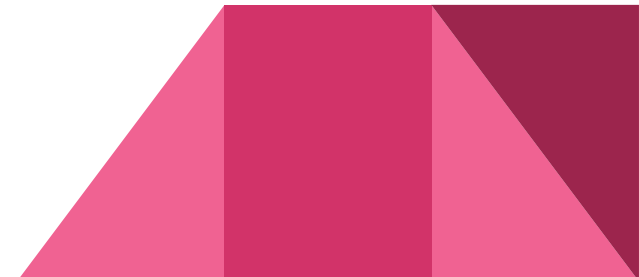
```
In [47]: fitArima<-arima(log(datawide),c(0,1,1),seasonal = list(order=c(0,1,1),period=12))  
fitArima
```

arima(x = log(datawide), order = c(0, 1, 1), seasonal = list(order = c(0, 1, 1),
period = 12))

Coefficients:

	ma1	sma1
	-0.3484	-0.5623
s.e.	0.0943	0.0774

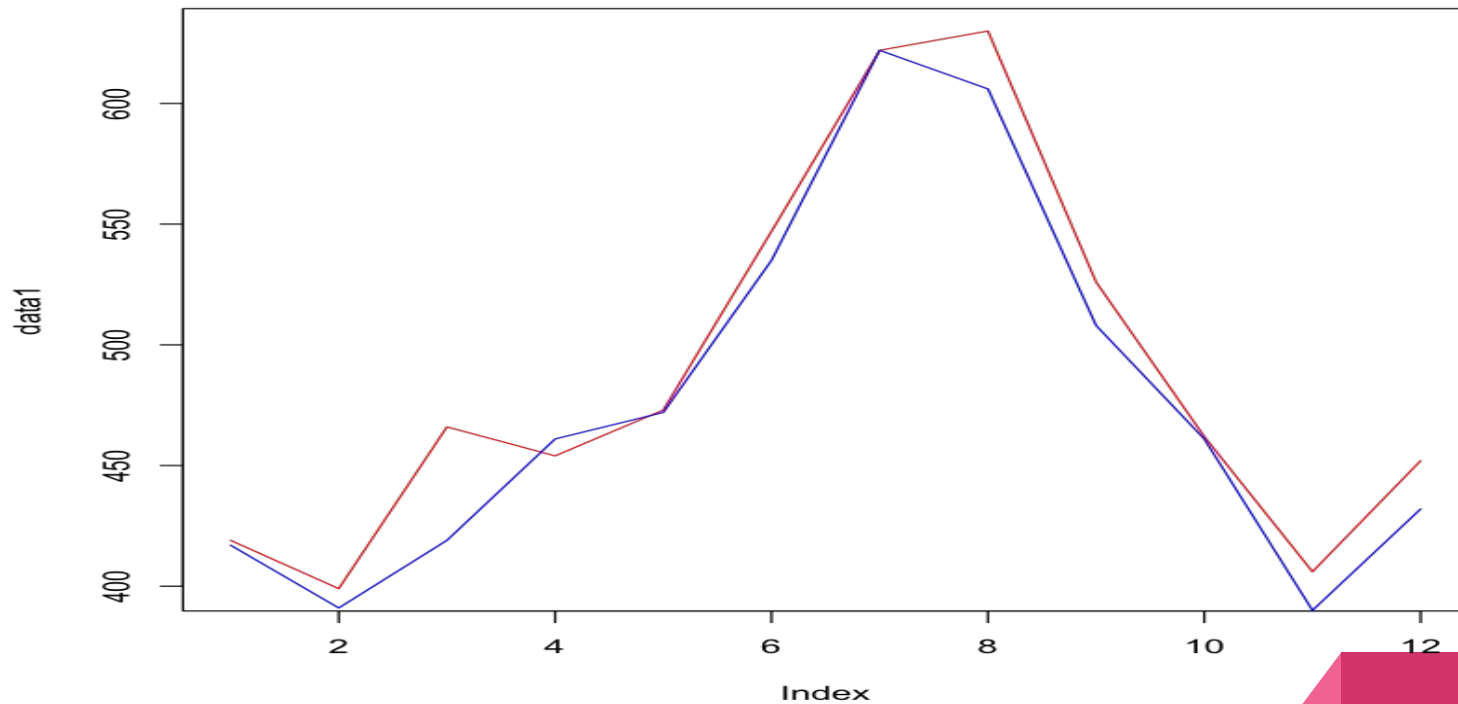
sigma^2 estimated as 0.001313: log likelihood = 223.63 , aic = -441.26



```
In [54]: pred<-predict(fitArima,n.ahead = 10*12)
pred1<-round(2.718^pred$pred,0)
data1=round(head(pred1,12),0)# prediction data
data2=round(tail(dataSet,12),0)# original data
pred1
data1
data2
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1960	419	399	466	454	473	547	622	630	526	462	406	452
1961	470	447	522	509	530	613	697	706	590	518	455	506
1962	526	501	585	570	594	687	781	791	661	580	510	568
1963	590	561	656	639	665	769	875	886	741	650	572	636
1964	661	629	735	716	746	862	980	993	830	728	641	713
1965	740	704	824	802	836	966	1098	1112	930	816	718	798
1966	830	789	923	899	936	1082	1231	1247	1042	915	804	895
1967	930	884	1034	1007	1049	1213	1379	1397	1168	1025	901	1003
1968	1042	991	1159	1129	1176	1359	1545	1565	1308	1148	1010	1123
1969	1167	1110	1299	1265	1317	1523	1732	1754	1466	1287	1132	1259
	419	399	466	454	473	547	622	630	526	462	406	452
	417	391	419	461	472	535	622	606	508	461	390	432

```
In [55]: plot(data1,col = "red",type="l")  
         lines(data2,col="blue")
```



Here,

Blue line = Original data

Red line = Predicted data

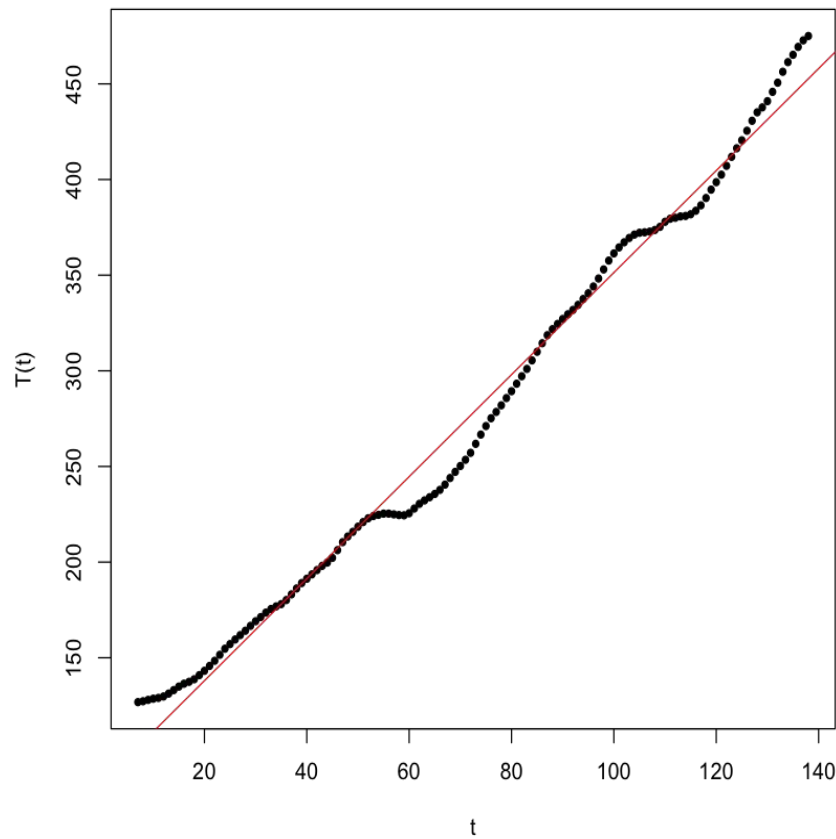
```
In [40]: dataSet.decompM$seasonal
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1949	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1950	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1951	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1952	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1953	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1954	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1955	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1956	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1957	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1958	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1959	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244
1960	0.9102304	0.8836253	1.0073663	0.9759060	0.9813780	1.1127758	1.2265555	1.2199110	1.0604919	0.9217572	0.8011781	0.8988244

```
In [56]: t <- seq(1, 144, 1)
modelTrend <- lm(formula = dataSet.decompM$trend ~ t)
predT <- predict.lm(modelTrend, newdata = data.frame(t))

plot(dataSet.decompM$trend[7:138] ~ t[7:138], ylab="T(t)", xlab="t",
     type="p", pch=20, main = "Trend Component: Modelled vs Observed")
lines(predT, col="red")
```

Trend Component: Modelled vs Observed



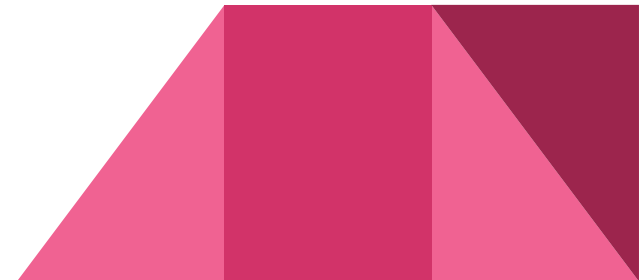
Model Fitting:

Trend Component

Inspecting the trend component in the decomposition plot suggests that the relationship is linear, thus fitting a linear model.

➤ Conclusion

The realistic estimations for 1961 are less than 1960 data. This is because the linear regression modelling the trend component under-estimates the trend component towards the end of the observable dataset (see figure above). A piecewise regression or switching to an ARMA/ARIMA model would give better predictions.





THANK YOU

DONE BY:

TEAM 6:

KAVEEN GANDHI

HARSHITHA KC

KEERTHANA L

KISHAN RAO