Pictorial Week 10: Time Series Week 1

How to: Install a Package

Use the install.packages function:

```
install.packages('fpp2')
install.packages('forecast')
install.packages('ggplot2')
```

You can also do this in one line of code if you also use the c() function:

```
install.packages(c('fpp2','forecast','ggplot2'))
```

How to: Load a Package

Loading the packages "forecast", "fpp2", and "ggplot2" using the library function:

How to: Assign a dataset in a package to an object

The gas dataset in the forecast package is assigned to an object called data

How to: Convert a dataset to a time series object/check that an object is a time series using the as.ts function:

```
data <- as.ts(data)
```

How to: Print the values of an R object to the console:

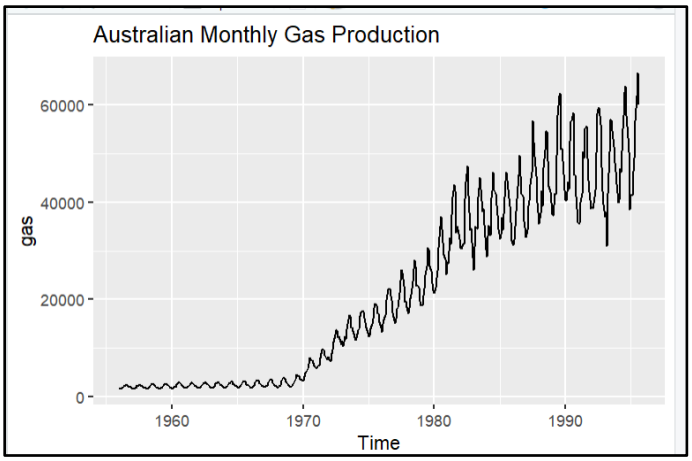
```
print(data)
```

Depending on the type of R object, you may need to use the argument 'n' within the print function to see all of the observations e.g. print(df, n = 110) if you had a dataset with 110 observations.

How to: Apply the autoplot function

The autoplot function is used to plot the dataset with the title "Australian monthly Gas production" with a y-axis label "gas"

```
6
7 autoplot(data) + ggtitle('Australian Monthly Gas Production') + ylab('gas')
8
```



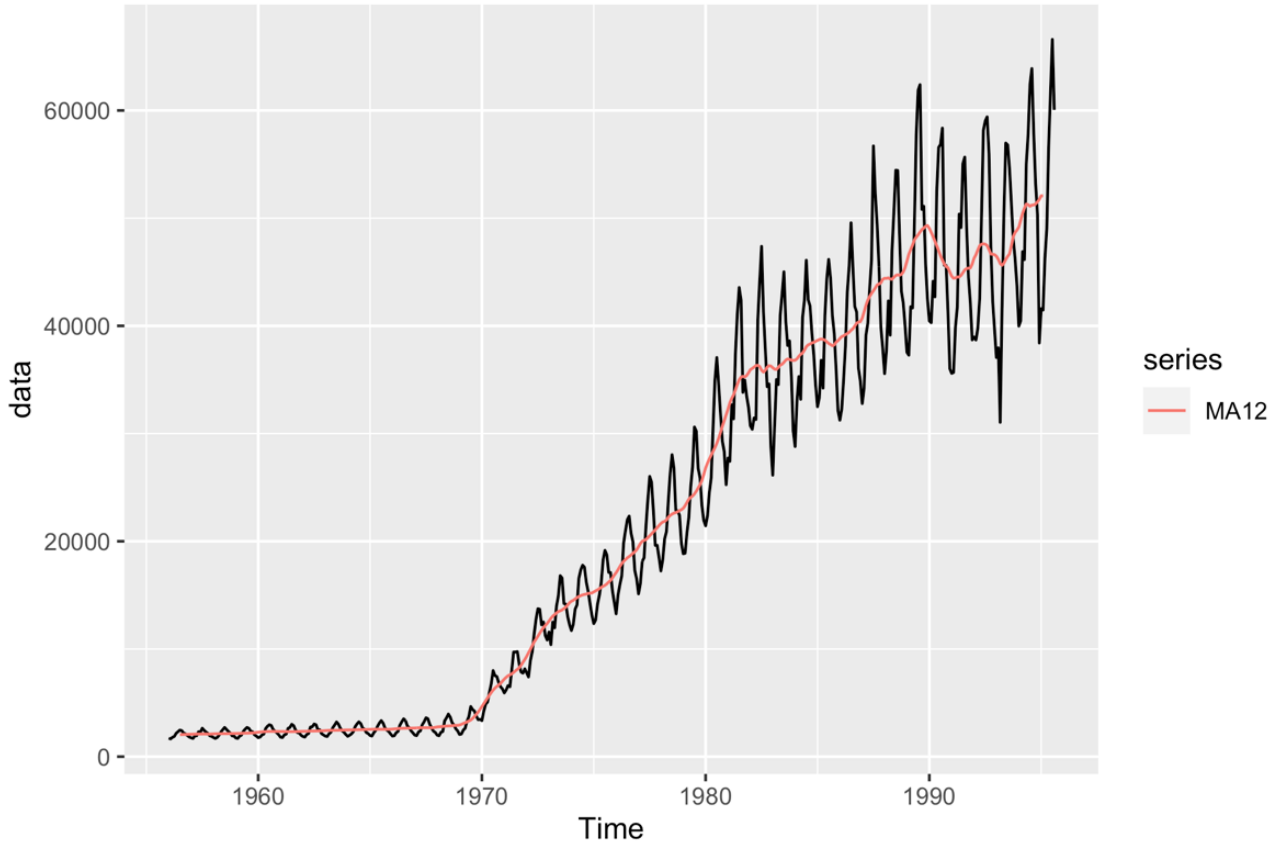
How to: Apply the ma() and autolayer() functions unto a plot of your data in order to get a CMA(k) series.

With k = 12 , we can calculate the CMA(12) series of our data and print it to the console with:

```
ma(data, 12)
```

And we can add it to our plot using the autolayer function:

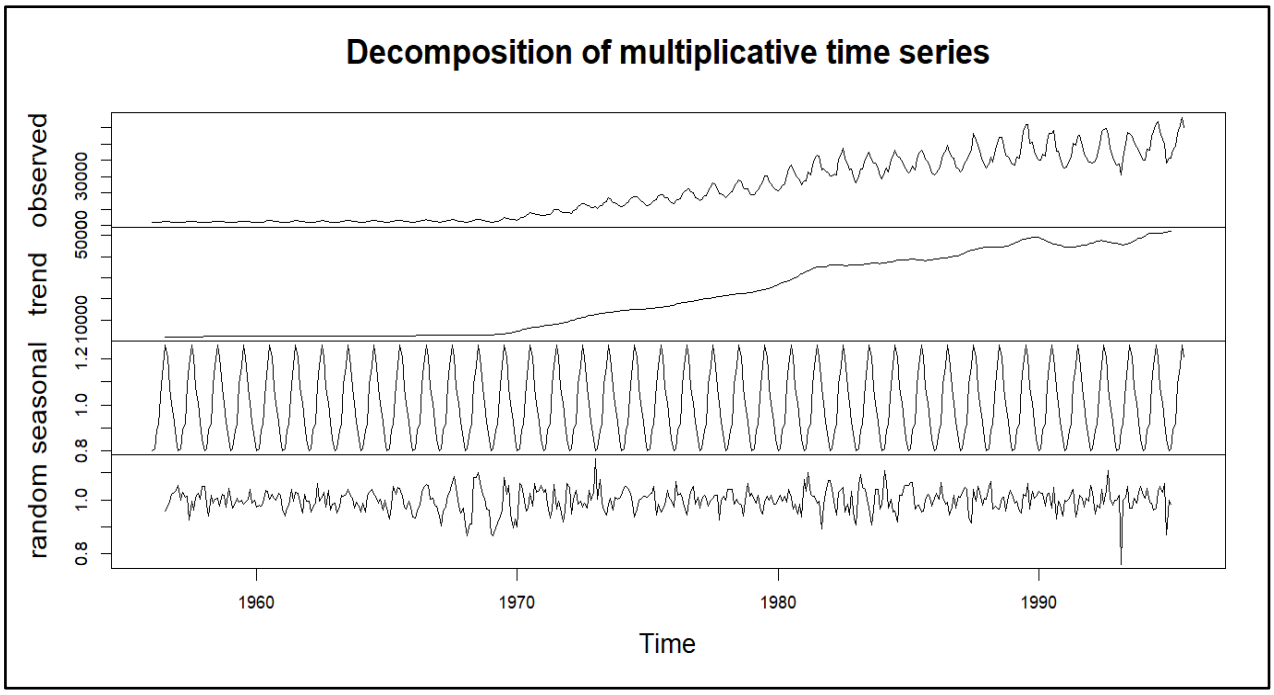
```
autoplot(data) + autolayer(ma(data, 12), series = "MA12")
```



How to: Apply the decompose() function

The decompose function is used on our data assuming a multiplicative model and the result is stored in an object called Decomp_model. The Decomp_model is then plotted.

```
11  Decomp_model <- decompose(data, type = 'multiplicative')
12  plot(Decomp_model)
```



How to: See the values of each of the components in the Decomp_model:

```
Decomp_model
```

OR

```
print(Decomp_model)
```

This will then print out the observed values, seasonal component values, trend values and random component values for each time period in the console. Try it and take a look at the values.