TimeSeries

Michael McCormack

March 12, 2018

```
suppressMessages(require(tidyverse))
```

```
## Warning: package 'tidyverse' was built under R version 3.4.2
## Warning: package 'tibble' was built under R version 3.4.3
## Warning: package 'tidyr' was built under R version 3.4.2
## Warning: package 'purrr' was built under R version 3.4.2
## Warning: package 'stringr' was built under R version 3.4.3
```

```
require(ggfortify)
```

```
## Loading required package: ggfortify
## Warning: package 'ggfortify' was built under R version 3.4.3
```

```
require(forecast)
```

```
## Loading required package: forecast
## Warning: package 'forecast' was built under R version 3.4.2
```

Step 1: Get the data

Download from website https://cdn.rawgit.com/mikejt33/DataViz/246c2026/data/flights.csv.gz Easiest to unzip locally then read in the data as a csv file.

```
flights <- read.csv('flights.csv')
```

Step 2: Initial EDA

Are there any null values?

Isolate the data you want to use. We recommend using arrival delay and departure delay times. Or Origin/Destination cities. Or Aggregate by Carrier.

```
arr_delay <- flights %>% arrange(FL_DATE) %>% group_by(FL_DATE) %>% summarise(mean(ARR_DELAY, na.rm = T
names(arr_delay) <- c('FL_DATE', 'AVE_DELAY')

dep_delay <- flights %>% arrange(FL_DATE) %>% group_by(FL_DATE) %>% summarise(mean(DEP_DELAY, na.rm = T
names(dep_delay) <- c('FL_DATE', 'AVE_DELAY')
```

Step 3: Fit the data to a TimeSeries object and plot

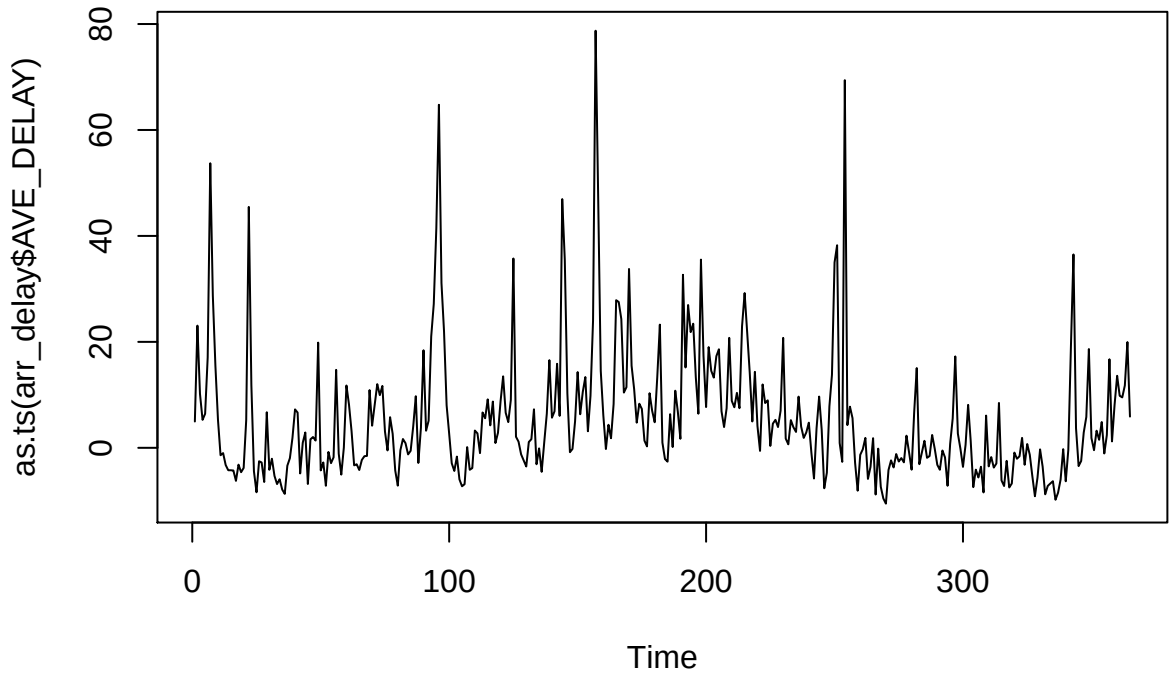
Refer to the slides for tips on how to do this.

```
arr_delay_ts <- as.ts(arr_delay)
dep_delay_ts <- as.ts(dep_delay)
```

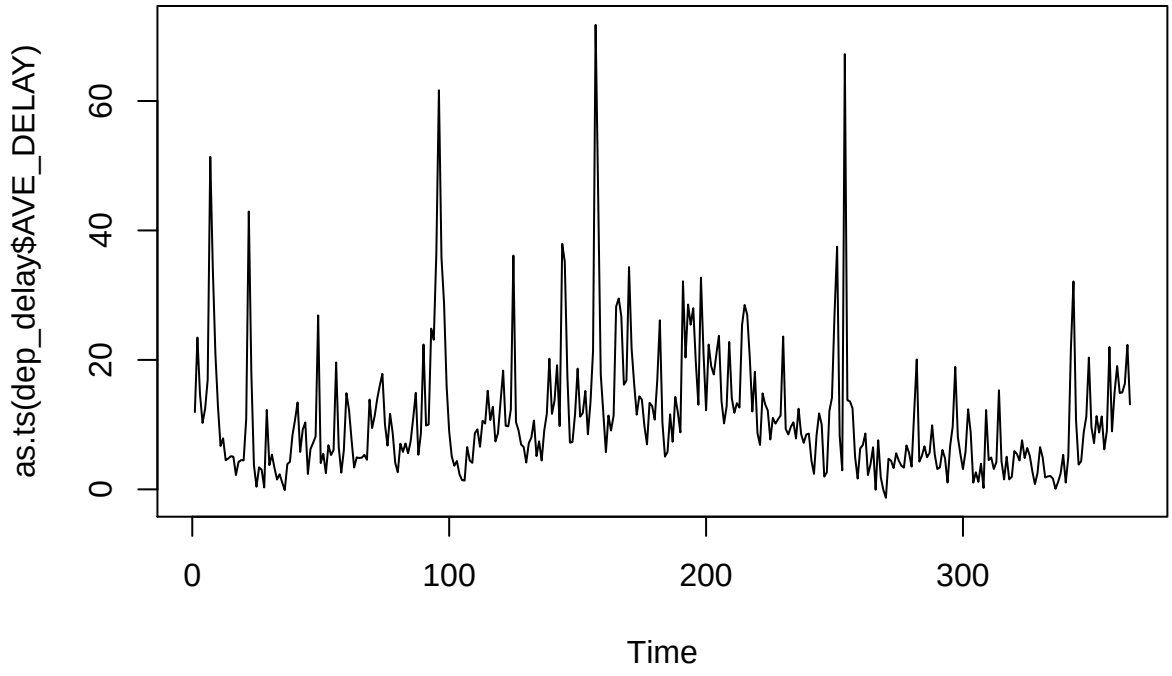
Step 4: Use TimeSeries object data to make an initial Visualization

Create a basic Visualization of the TimeSeries object data

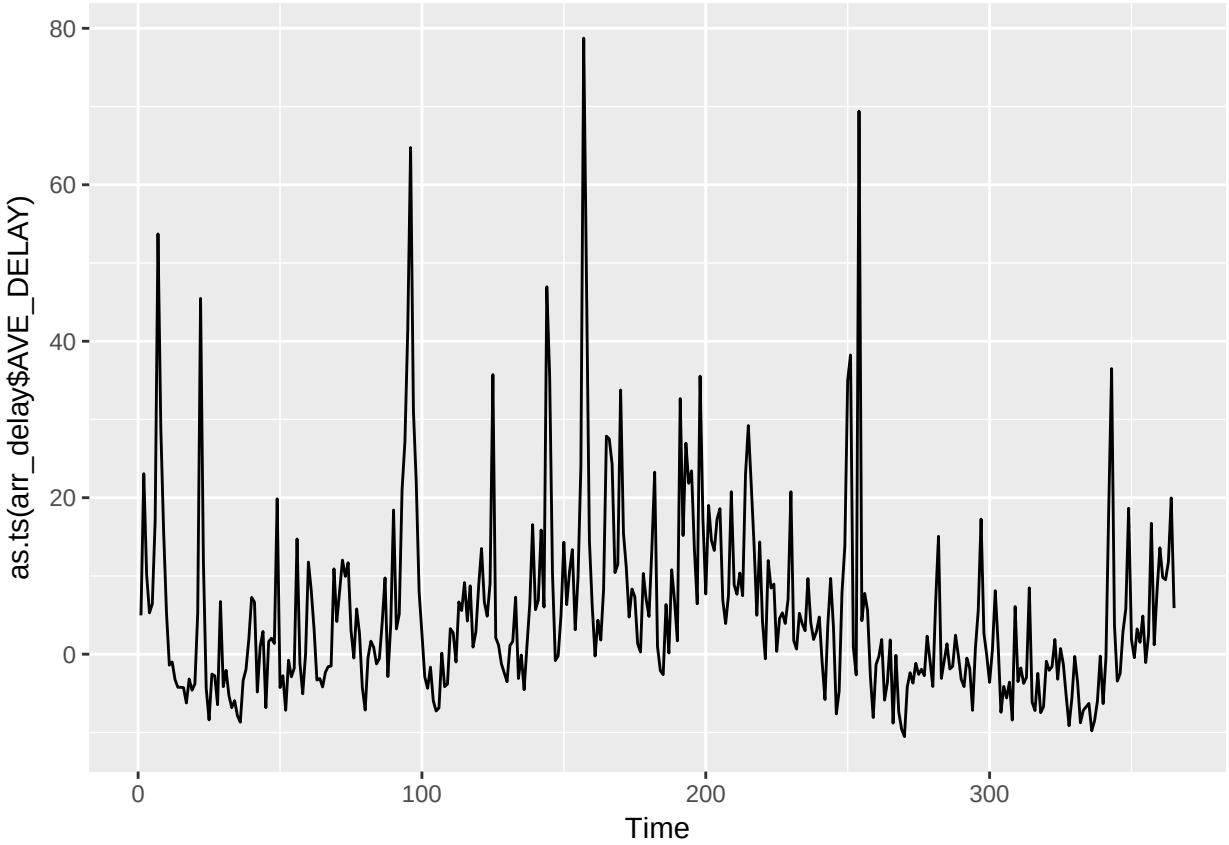
```
plot(as.ts(arr_delay$AVE_DELAY))
```




```
plot(as.ts(dep_delay$AVE_DELAY))
```




```
### Advanced Portion
autoplot(as.ts(arr_delay$AVE_DELAY))# + geom_line()
```

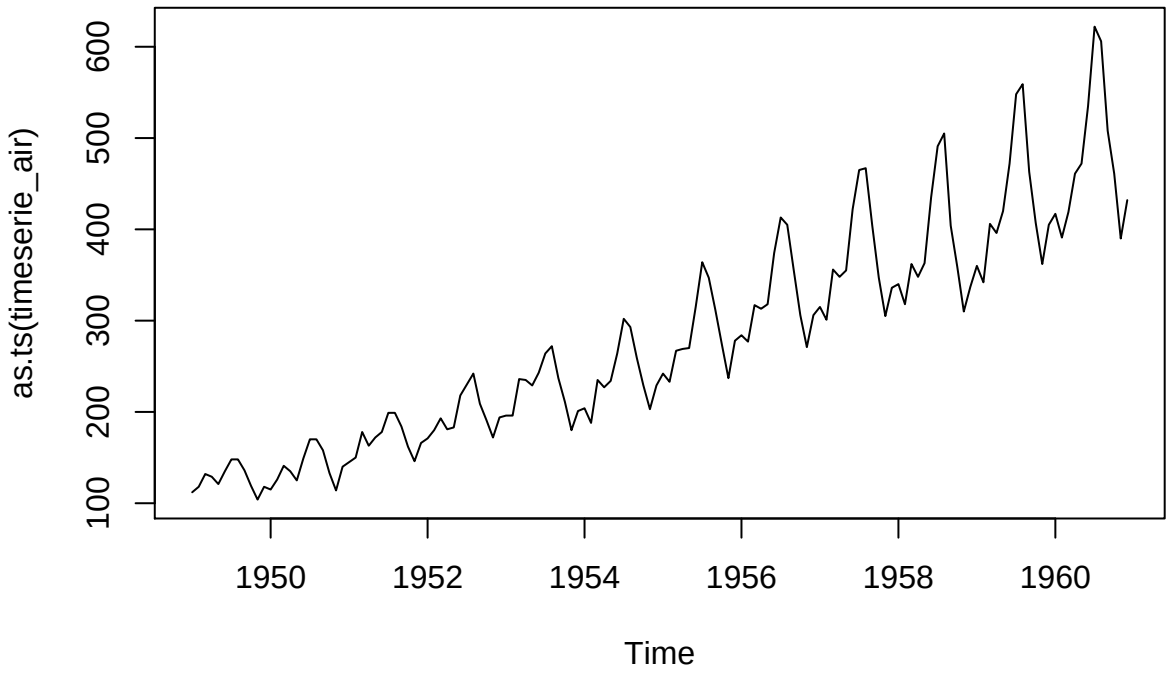


Dive in Deeper to TimeSeries For this portion of our lab we will be using data from the AirPassengers Dataset

```
data(AirPassengers)
```

Step 5: Make an inital TimeSeries Visual of the data

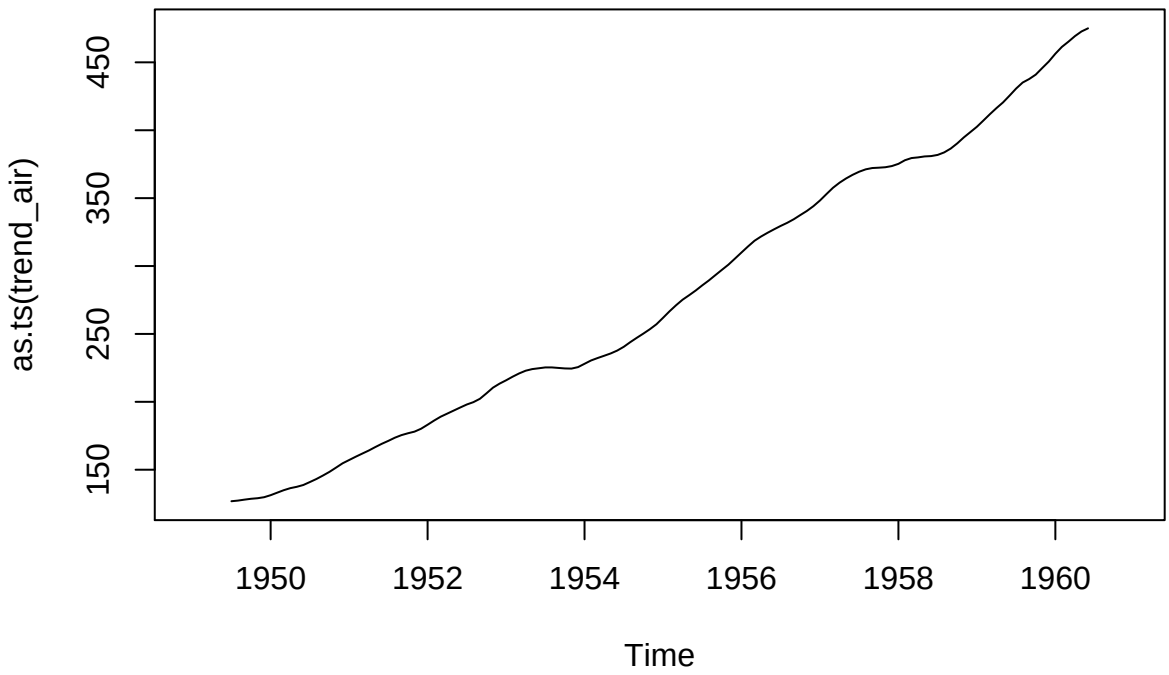
```
timeserie_air = AirPassengers
plot(as.ts(timeserie_air))
```



##

Step 6: Compute the Moving Average of this data and vizualize this

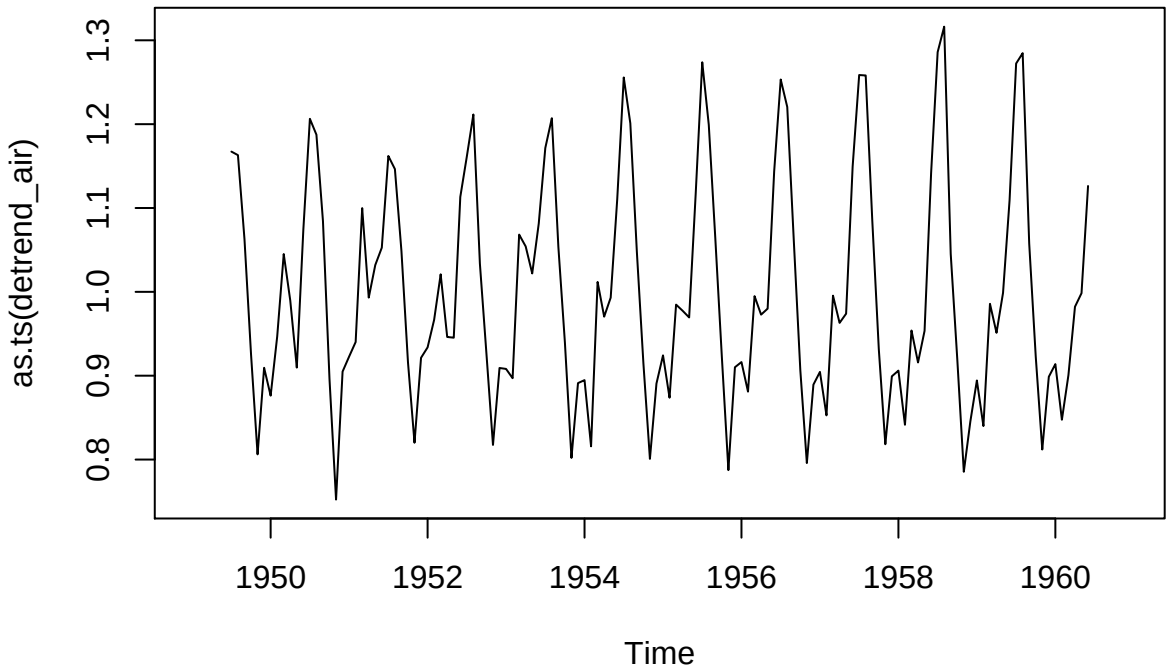
```
trend_air = ma(timeserie_air, order = 12, centre = T)
#lines(trend_air)
plot(as.ts(trend_air))
```



##

Step 7: Remove the Trend from the data and Visualize this

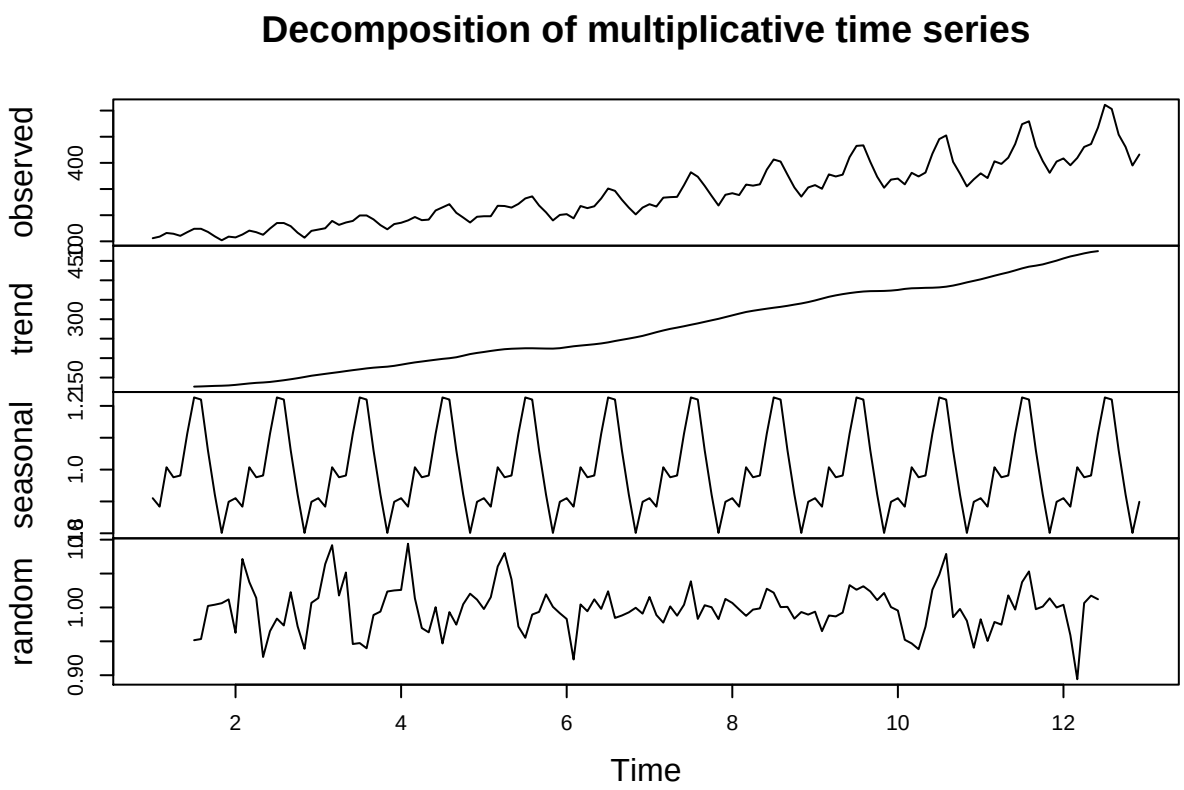
```
detrend_air = timeserie_air / trend_air
plot(as.ts(detrend_air))
```



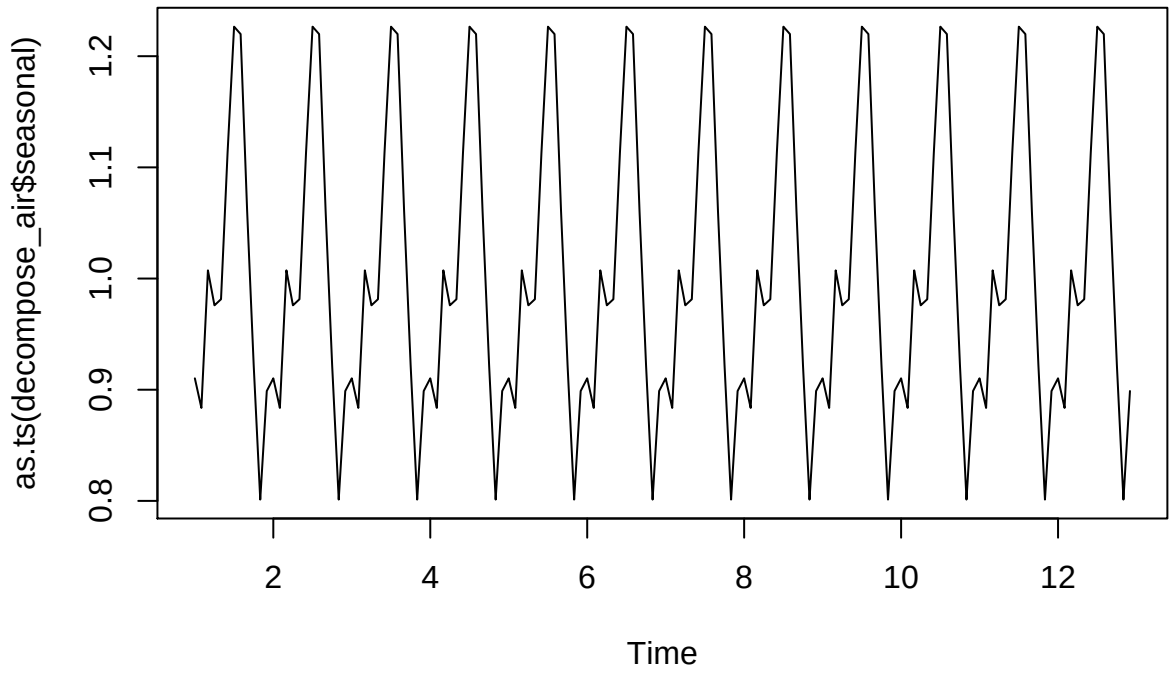
##

Step 8: Create a decomposition of the data by month – Hint (Frequency = 12)

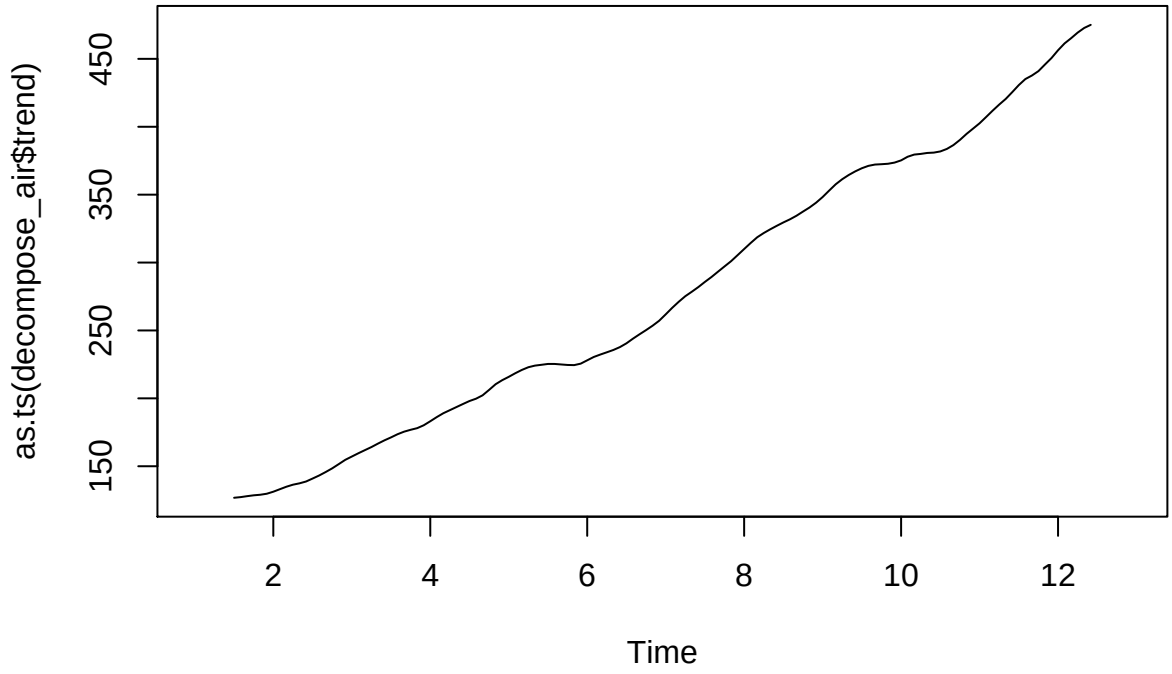
```
ts_air = ts(timeserie_air, frequency = 12)
decompose_air = decompose(ts_air, "multiplicative")

plot(decompose_air)
```



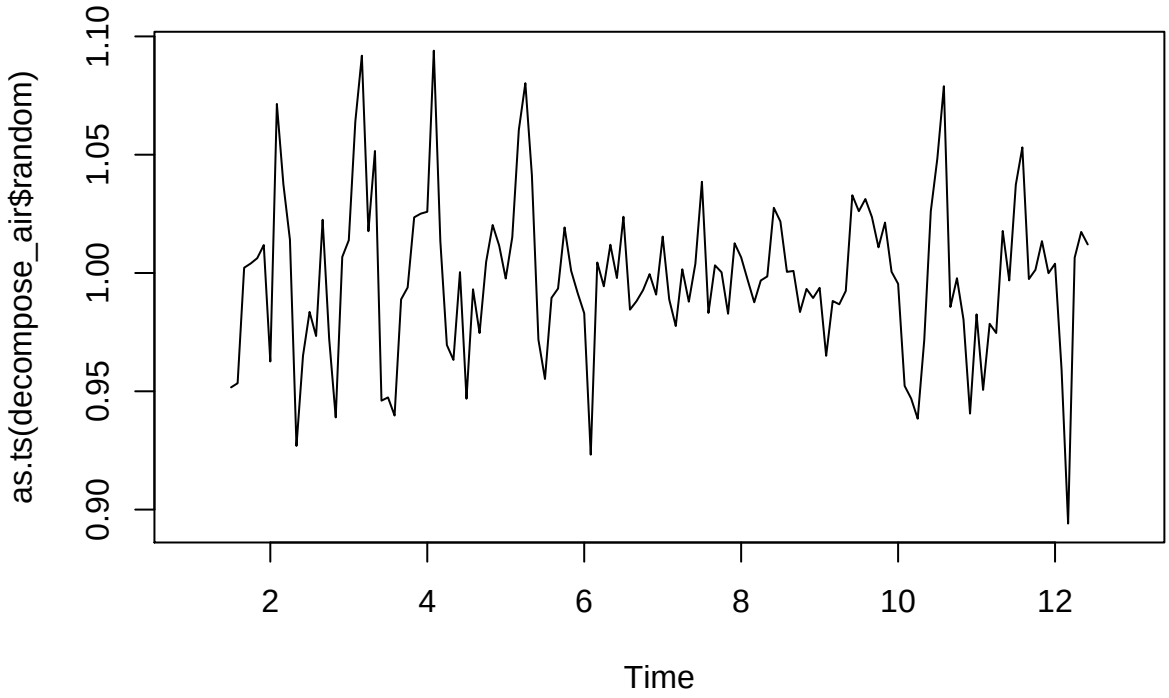

```
### Advanced answers
plot(as.ts(decompose_air$seasonal))
```




```
plot(as.ts(decompose_air$trend))
```



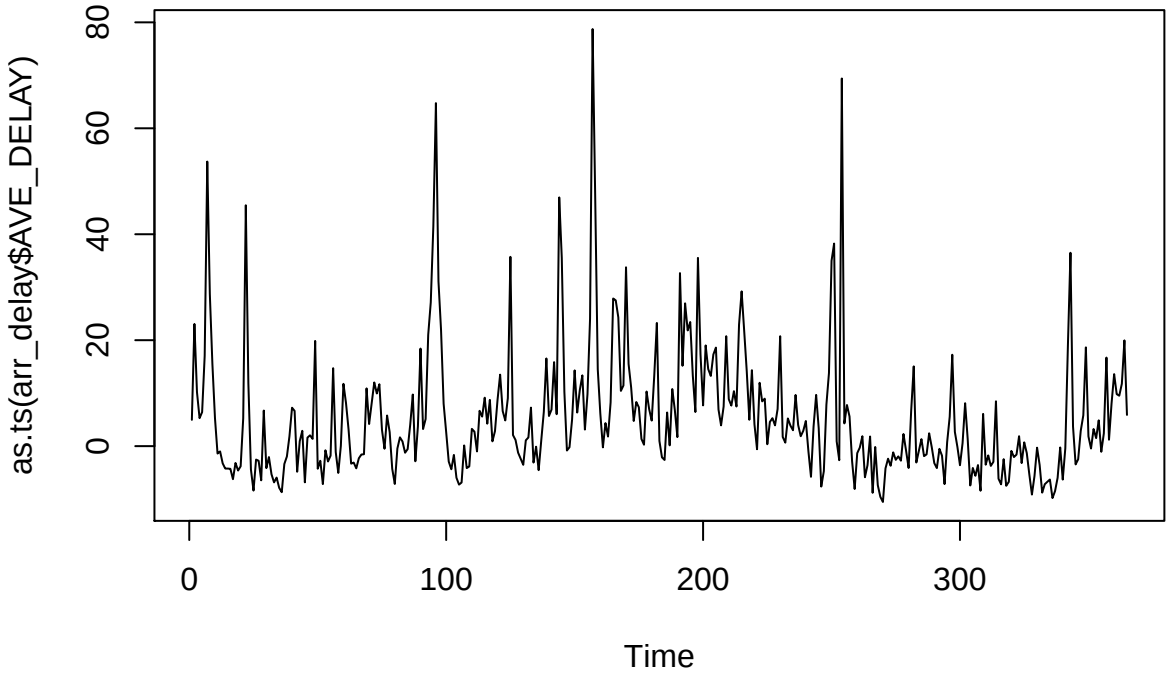

```
plot(as.ts(decompose_air$random))
```



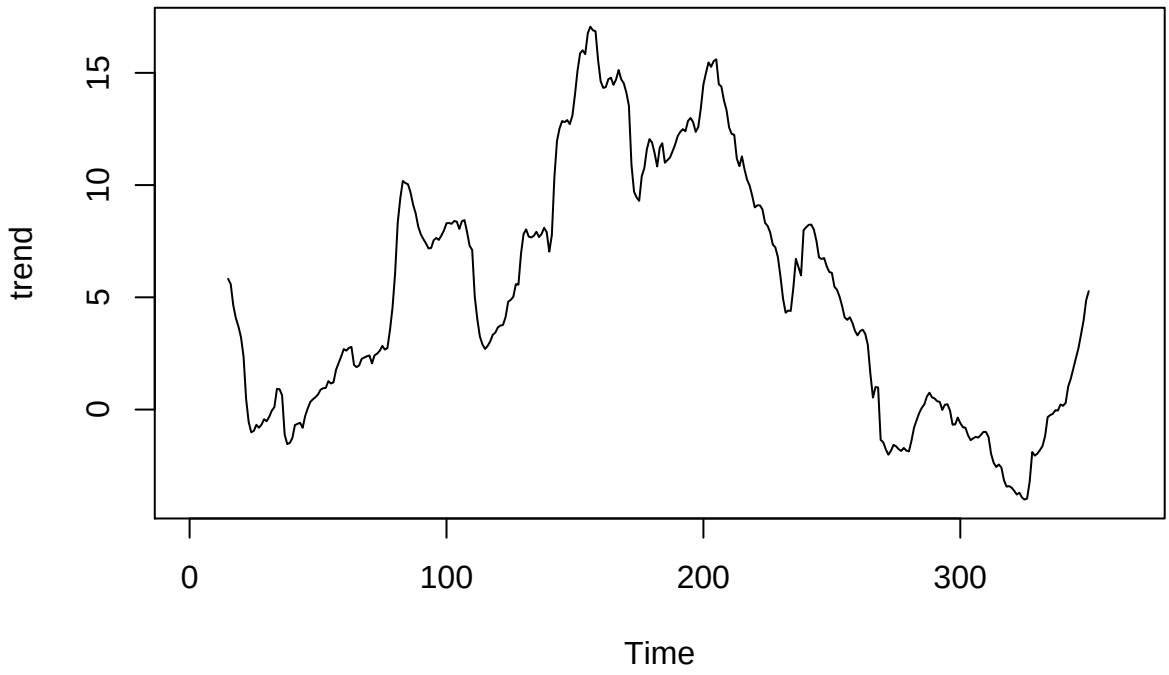
##

Step 9: Create your own simple moving average for monthly data

```
moving_ave <- function(x,n=5){stats::filter(x,rep(1/n,n), sides=2)}

trend <- moving_ave(as.ts(arr_delay$AVE_DELAY),30)
plot(as.ts(arr_delay$AVE_DELAY))
```




```
plot(trend)
```


trend
0
5
10
15
0
100
200
300
Time