8 – Time series

Make sure to install the packages “TTR” and “forecast”

```
#install.packages("TTR")
#install.packages("forecast")
```

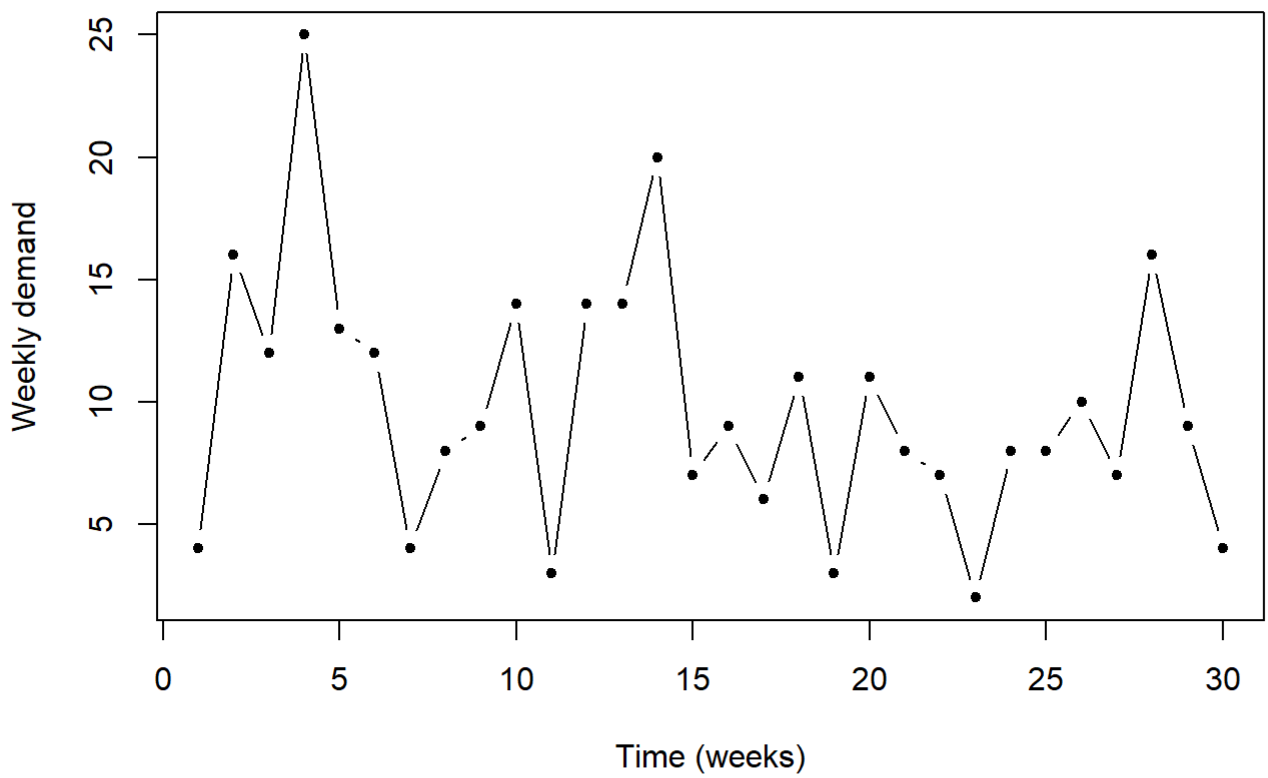
Also, don’t forget to load these libraries

```
library(TTR)
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.4.4
```

Observe the following time series, the weekly demand for some product.

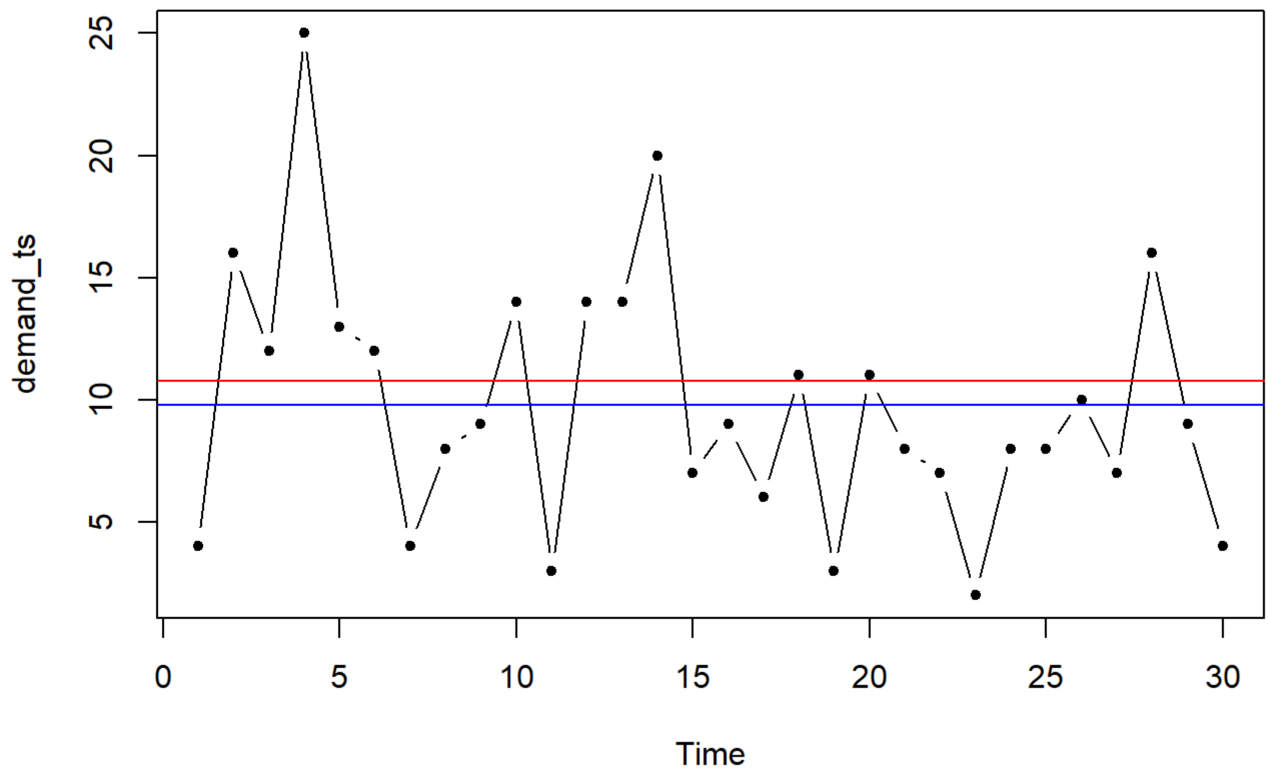
```
weekly_demand <- c(
  4, 16, 12, 25, 13, 12, 4,  8, 9, 14,
  3, 14, 14, 20,  7,  9, 6, 11, 3, 11,
  8,  7,  2,  8,  8, 10, 7, 16, 9,  4
)
demand_ts <- ts(weekly_demand)
plot.ts(demand_ts, type = 'b', pch = 20,
        xlab = "Time (weeks)",
        ylab = "Weekly demand")
```



Time series models

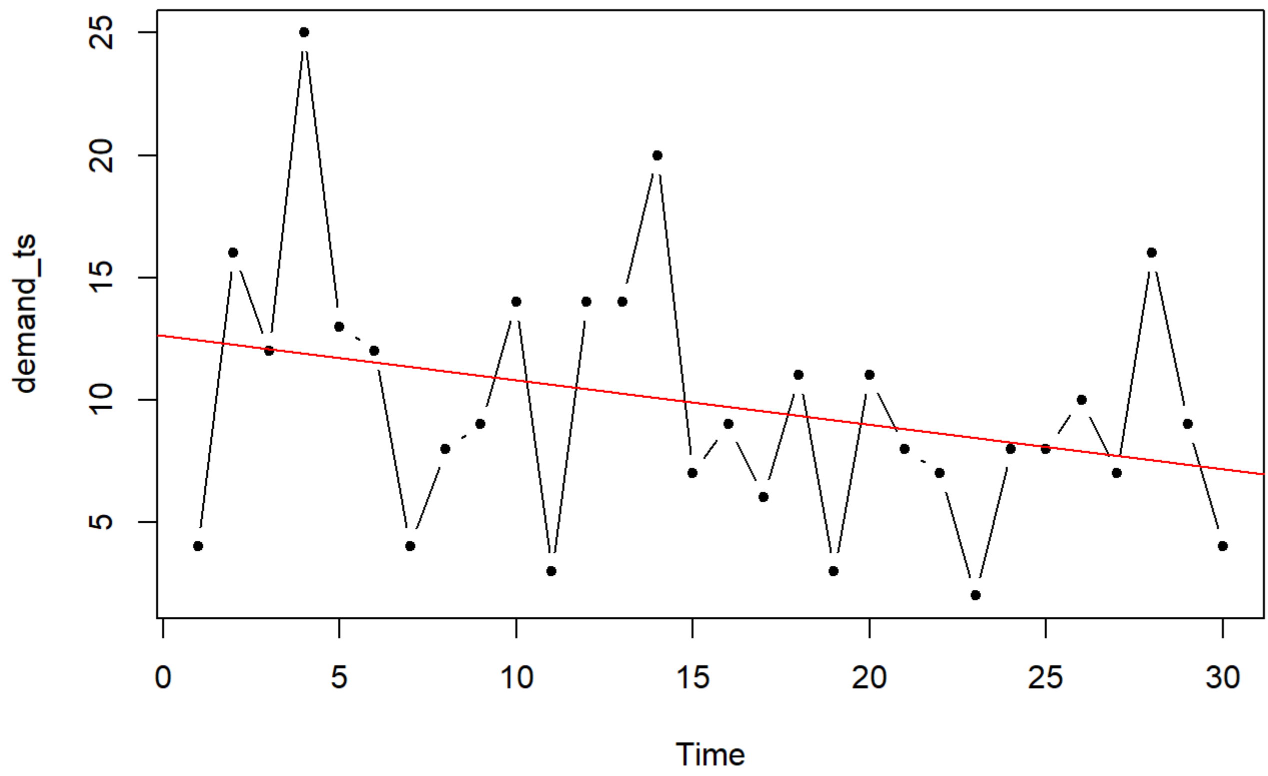
In the chart below, we attempt to model observed values with a constant function $X_t = b + \epsilon_t$. As an estimate for b, we take the average of the first observations. We can basically use our own judgement to choose how many.

```
b_est_20 <- mean(weekly_demand[1:20]) # the first 20 observations
b_est_30 <- mean(weekly_demand[1:30]) # the first 30 observations
plot.ts(demand_ts, type = 'b', pch = 20)
abline(h = b_est_20, col = 'red')
abline(h = b_est_30, col = 'blue')
```



If we want to model these values with a linear function $X_t = b_0 + b_1 t + \epsilon_t$, we can use a technique we actually already used in another context: linear regression.

```
week <- 1:length(weekly_demand)
demand_lm <- lm(weekly_demand ~ week)
plot.ts(demand_ts, type = 'b', pch = 20)
abline(demand_lm, col = 'red')
```



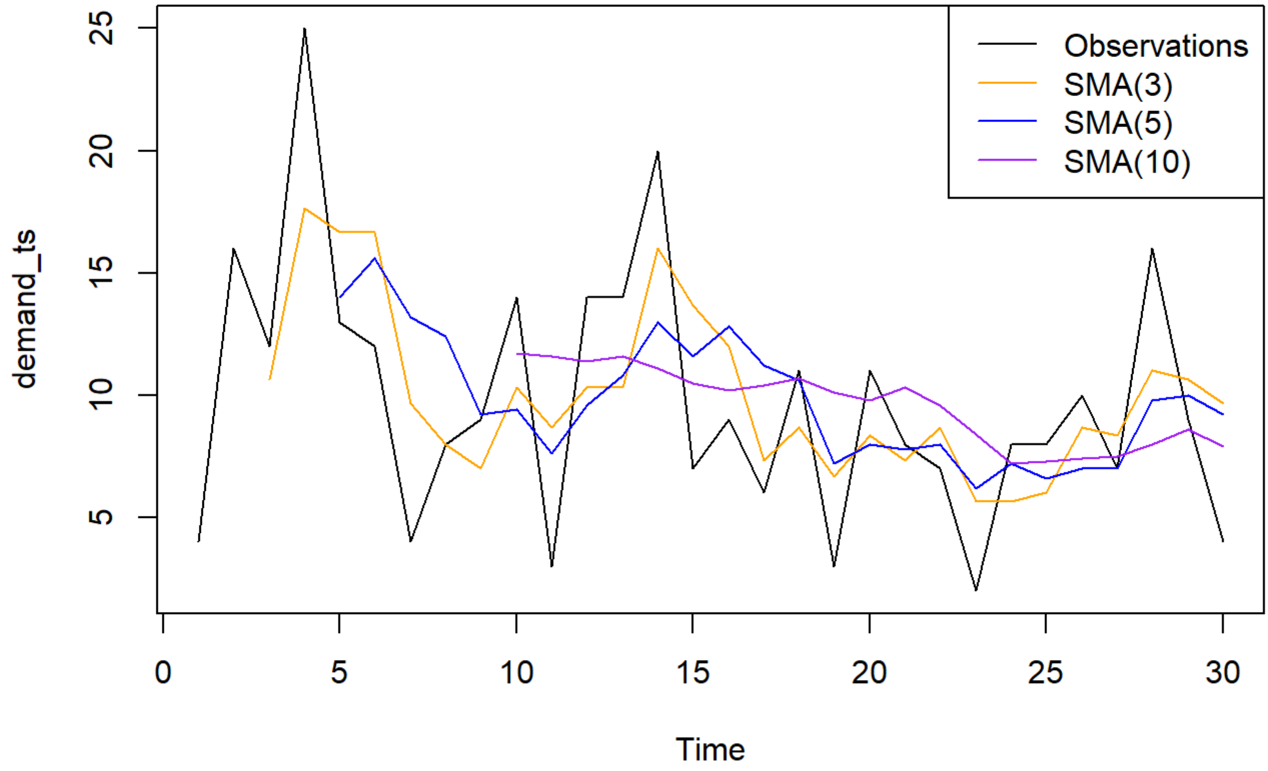
The previous estimate doesn't seem useful. If we use the regression line to make a forecast of weekly demand in the future, we would expect that the demand will soon drop to zero.

Moving average

We basically need a method that takes the last observations into account, but forgets observations after some time. Older observations may no longer be representative of the current demand. Moving averages are such methods.

Simple Moving Average

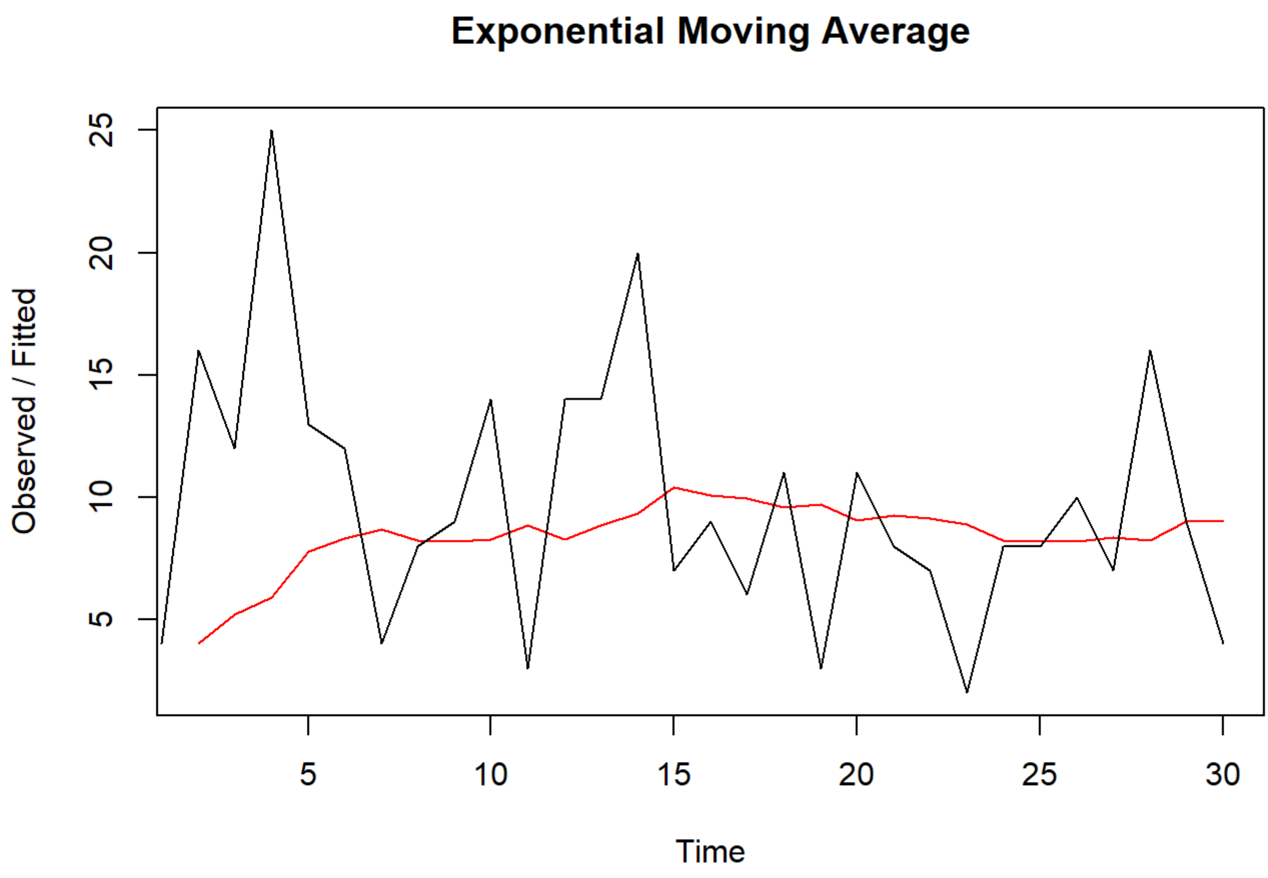
```
sma_3 <- SMA(x = demand_ts, n = 3)
sma_5 <- SMA(x = demand_ts, n = 5)
sma_10 <- SMA(x = demand_ts, n = 10)

plot.ts(demand_ts)
lines(sma_3, col = 'orange')
lines(sma_5, col = 'blue')
lines(sma_10, col = 'purple')
legend("topright", lty = 1,
       c("Observations", "SMA(3)", "SMA(5)", "SMA(10)"),
       col = c("black", "orange", "blue", "purple"))
```



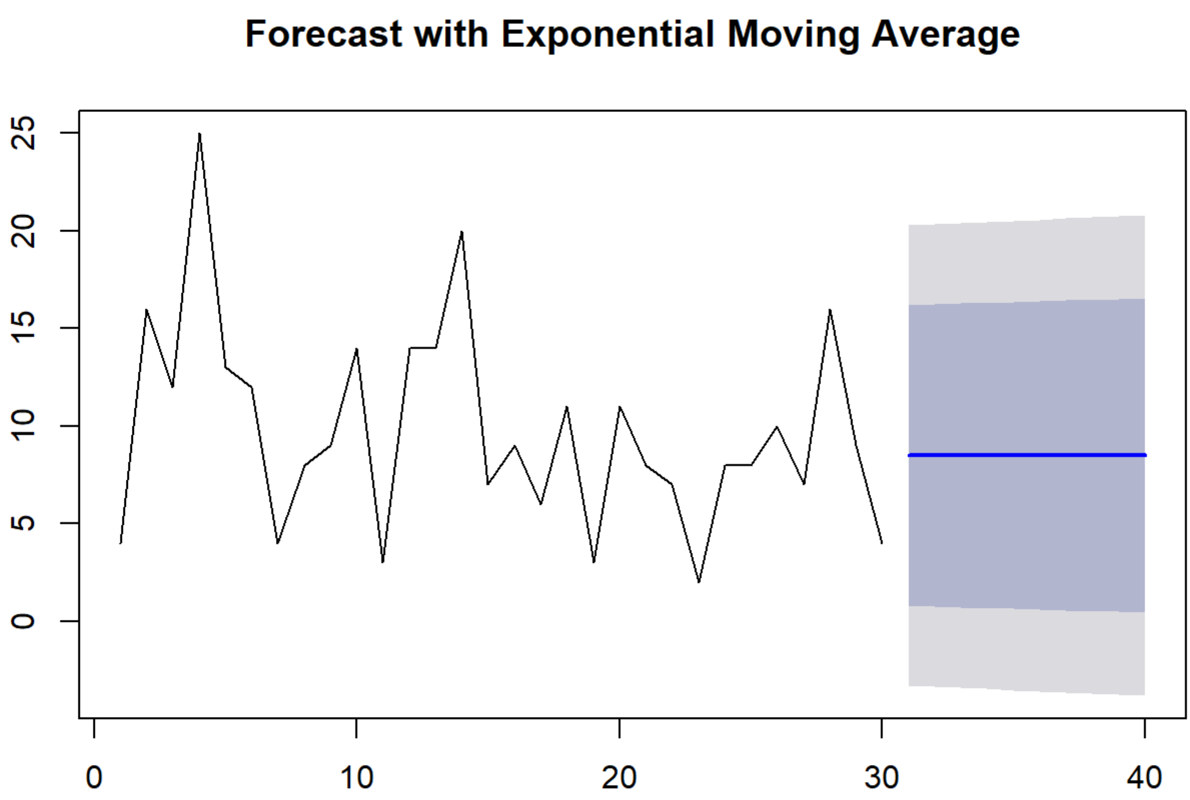
Exponential Moving Average

$X_t = \alpha x_t + (1 - \alpha) X_{t-1}$

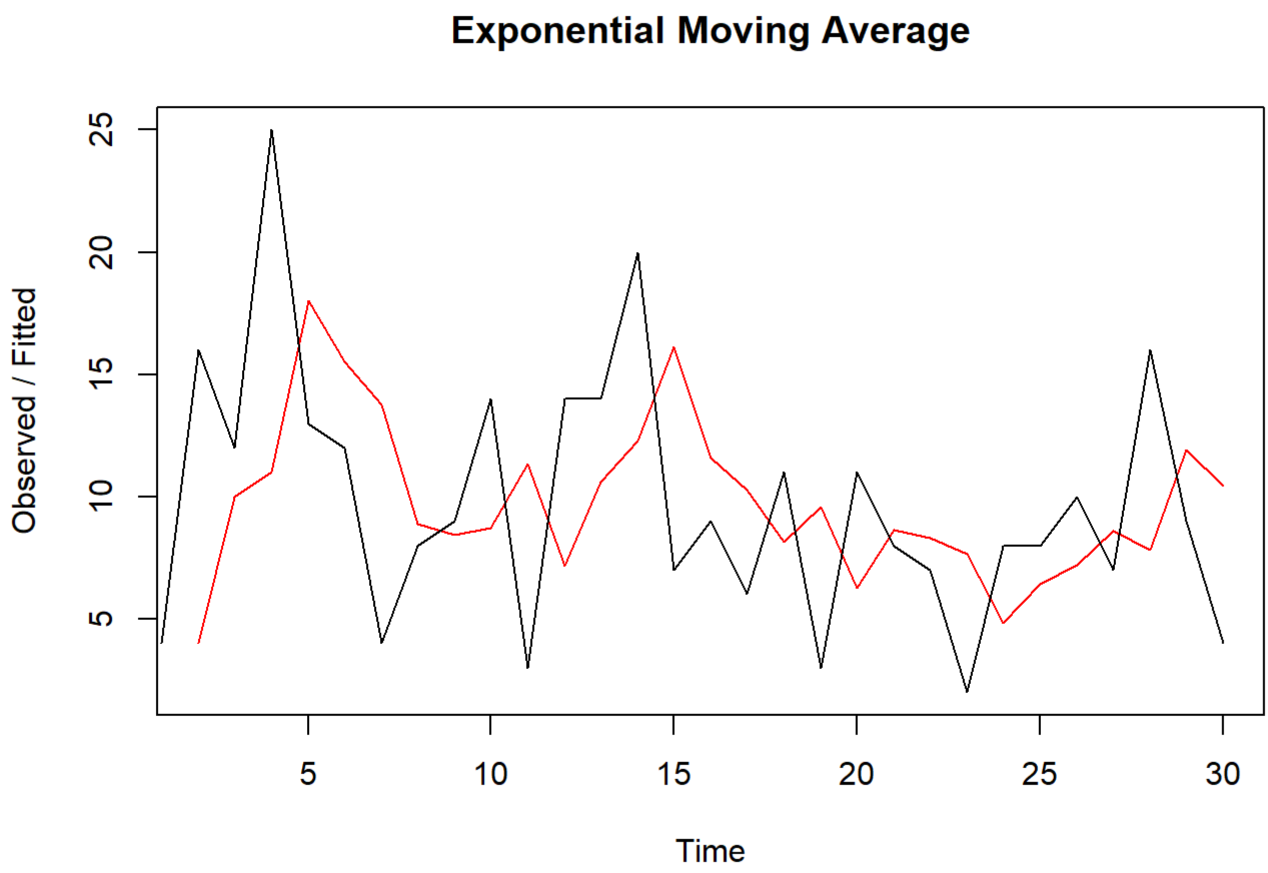
```
# Play with the value of alpha. What happens if alpha is close to 1?
ema <- HoltWinters(demand_ts, alpha = 0.1, beta = FALSE, gamma = FALSE)
plot(ema, main = "Exponential Moving Average")
```



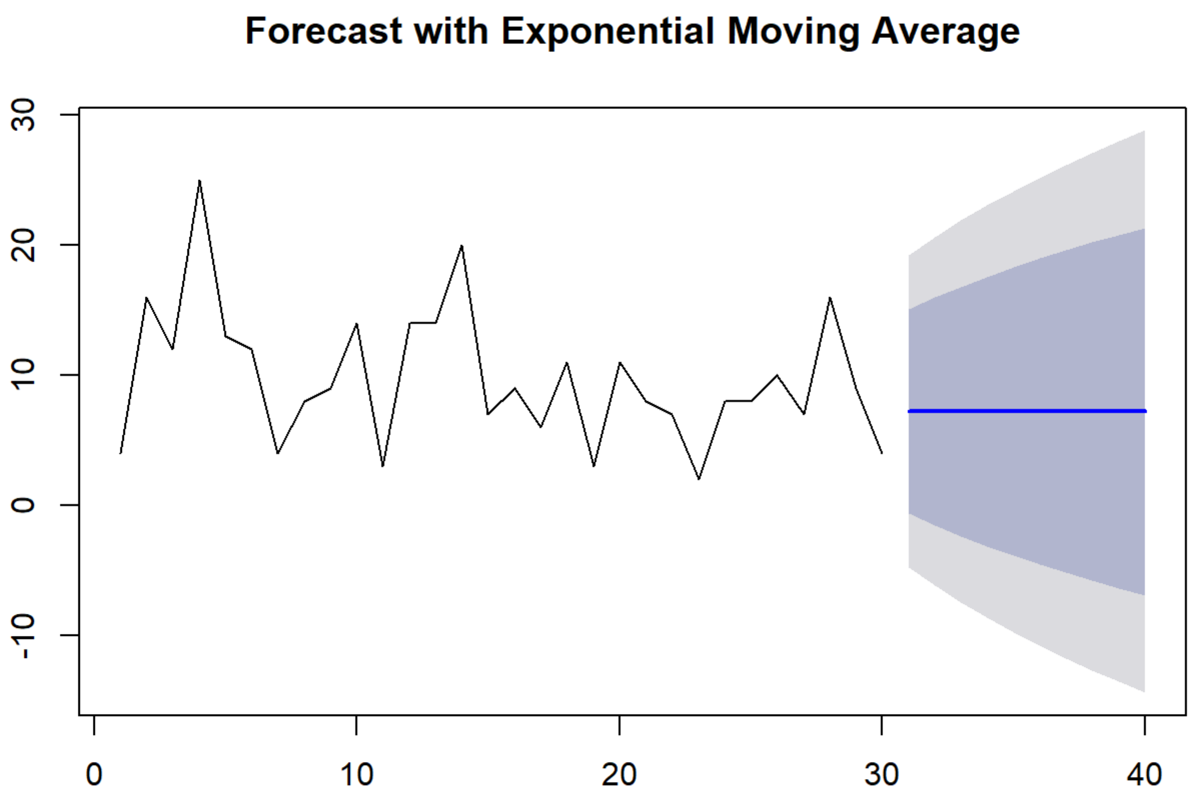

```
ema_fc <- forecast(ema, h = 10)
plot(ema_fc, main = "Forecast with Exponential Moving Average")
```



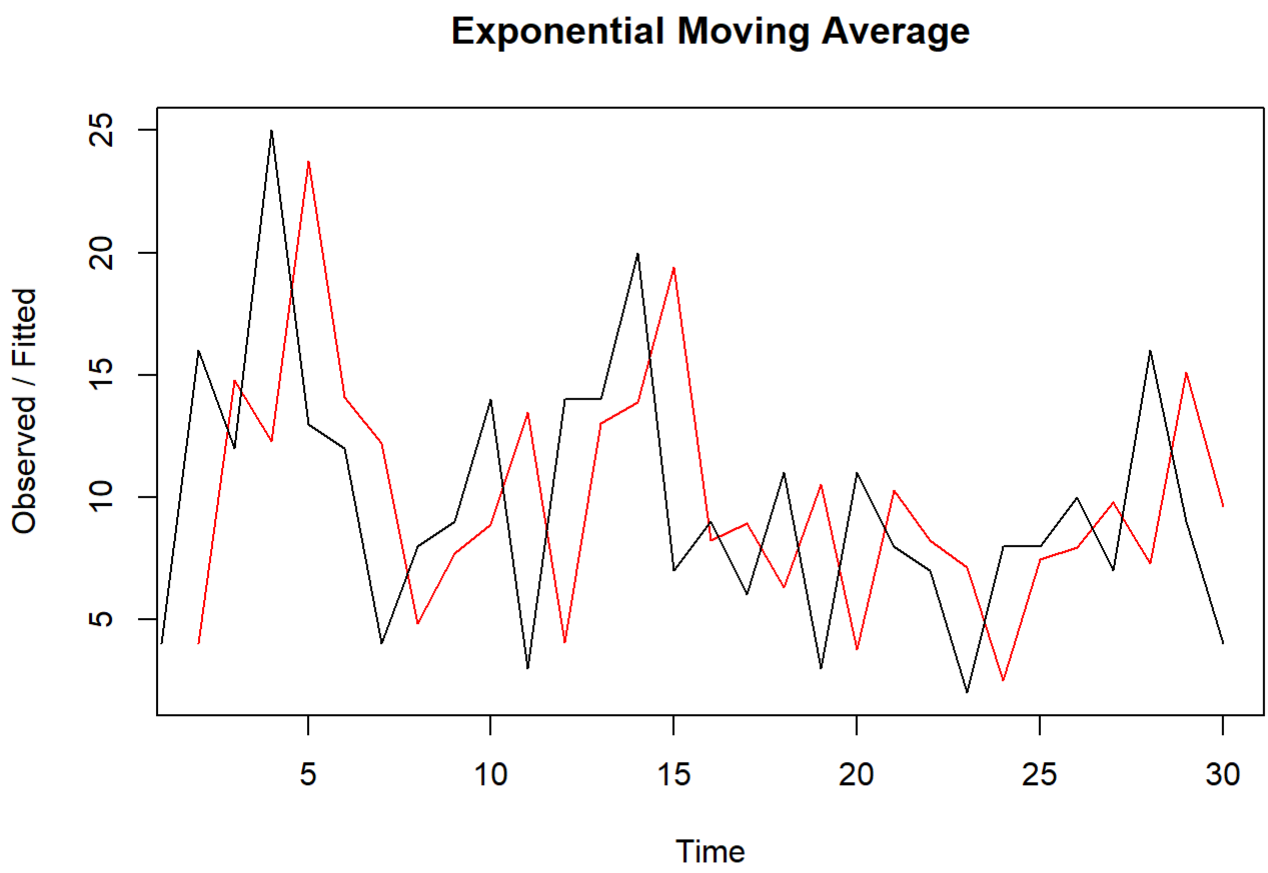
```
# Play with the value of alpha. What happens if alpha is close to 1?
ema <- HoltWinters(demand_ts, alpha = 0.5, beta = FALSE, gamma = FALSE)
plot(ema, main = "Exponential Moving Average")
```



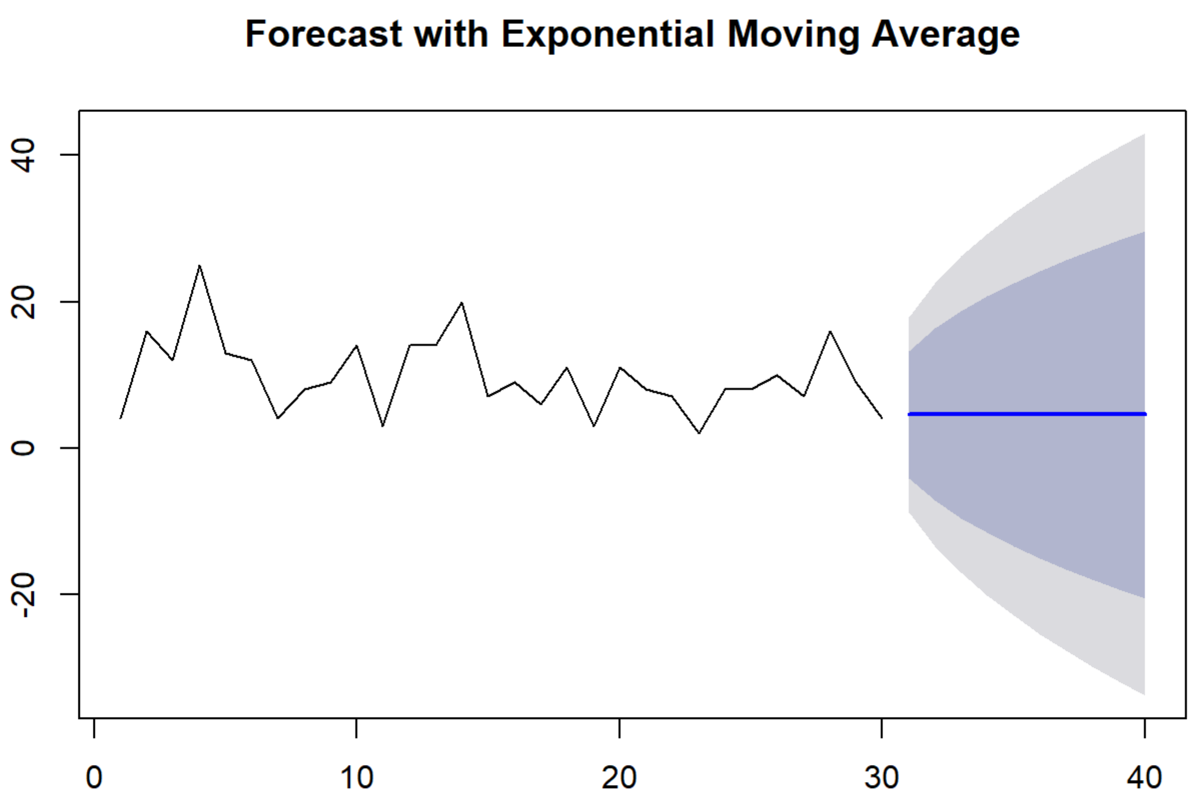

```
ema_fc <- forecast(ema, h = 10)
plot(ema_fc, main = "Forecast with Exponential Moving Average")
```




```
# Play with the value of alpha. What happens if alpha is close to 1?
ema <- HoltWinters(demand_ts, alpha = 0.9, beta = FALSE, gamma = FALSE)
plot(ema, main = "Exponential Moving Average")
```



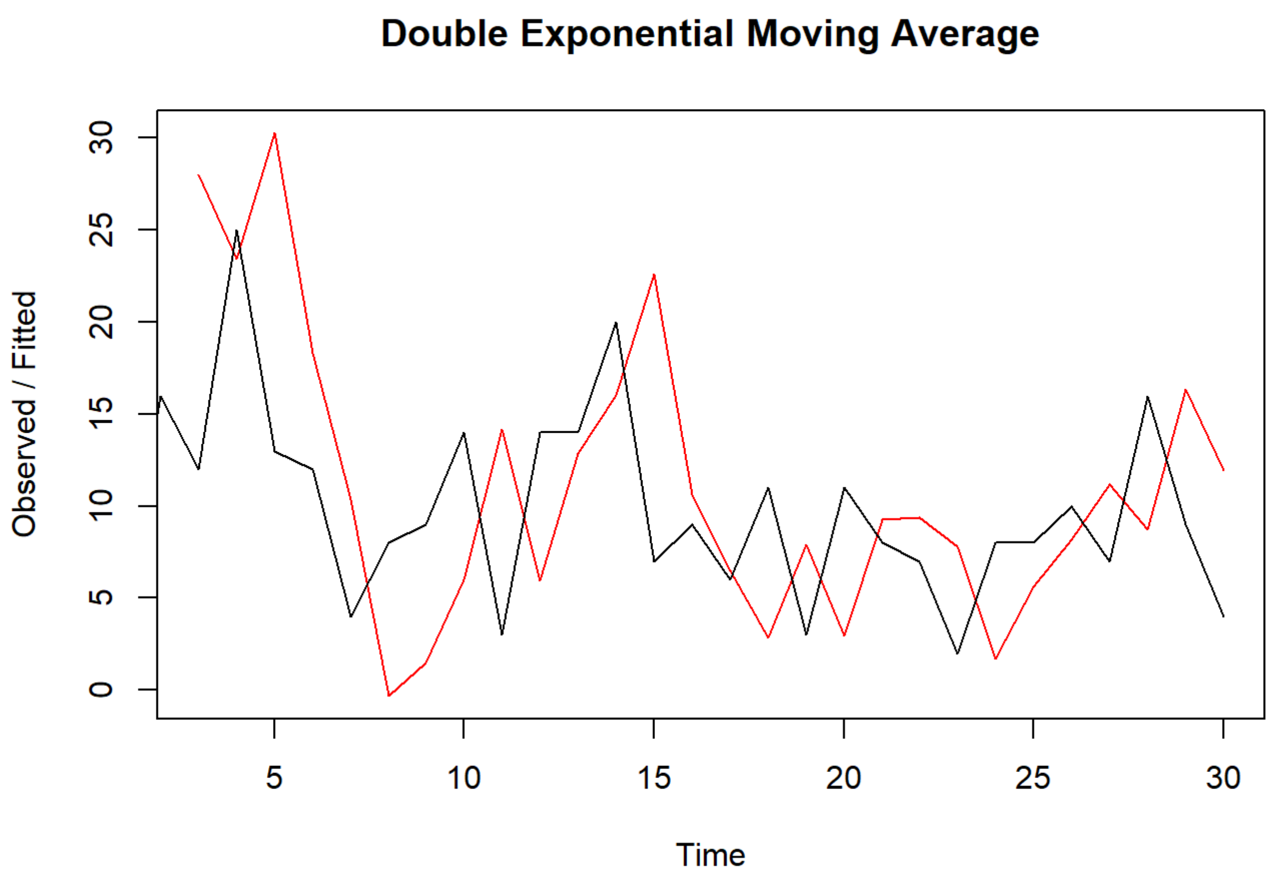

```
ema_fc <- forecast(ema, h = 10)
plot(ema_fc, main = "Forecast with Exponential Moving Average")
```



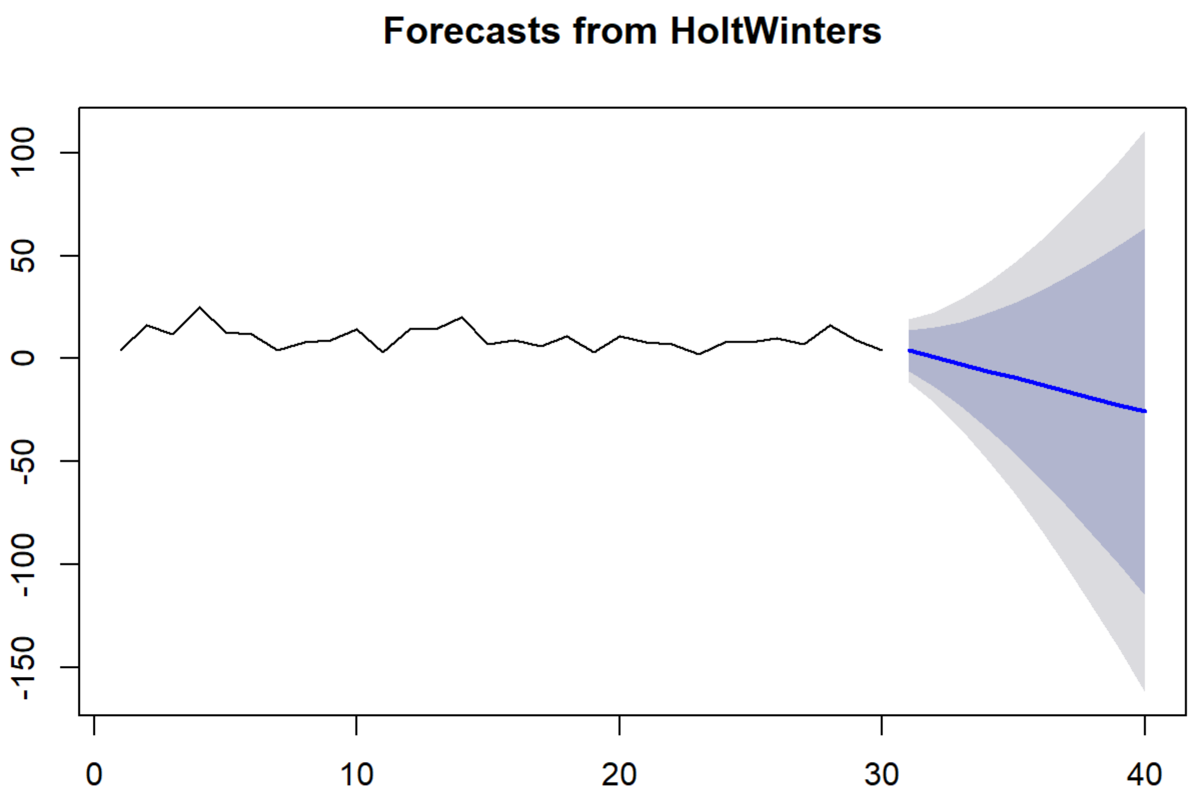
Double Exponential Moving Average

Is especially useful if you have data with a long term (linear) trend.

```
dema <- HoltWinters(demand_ts, gamma = FALSE)
plot(dema, main = "Double Exponential Moving Average")
```



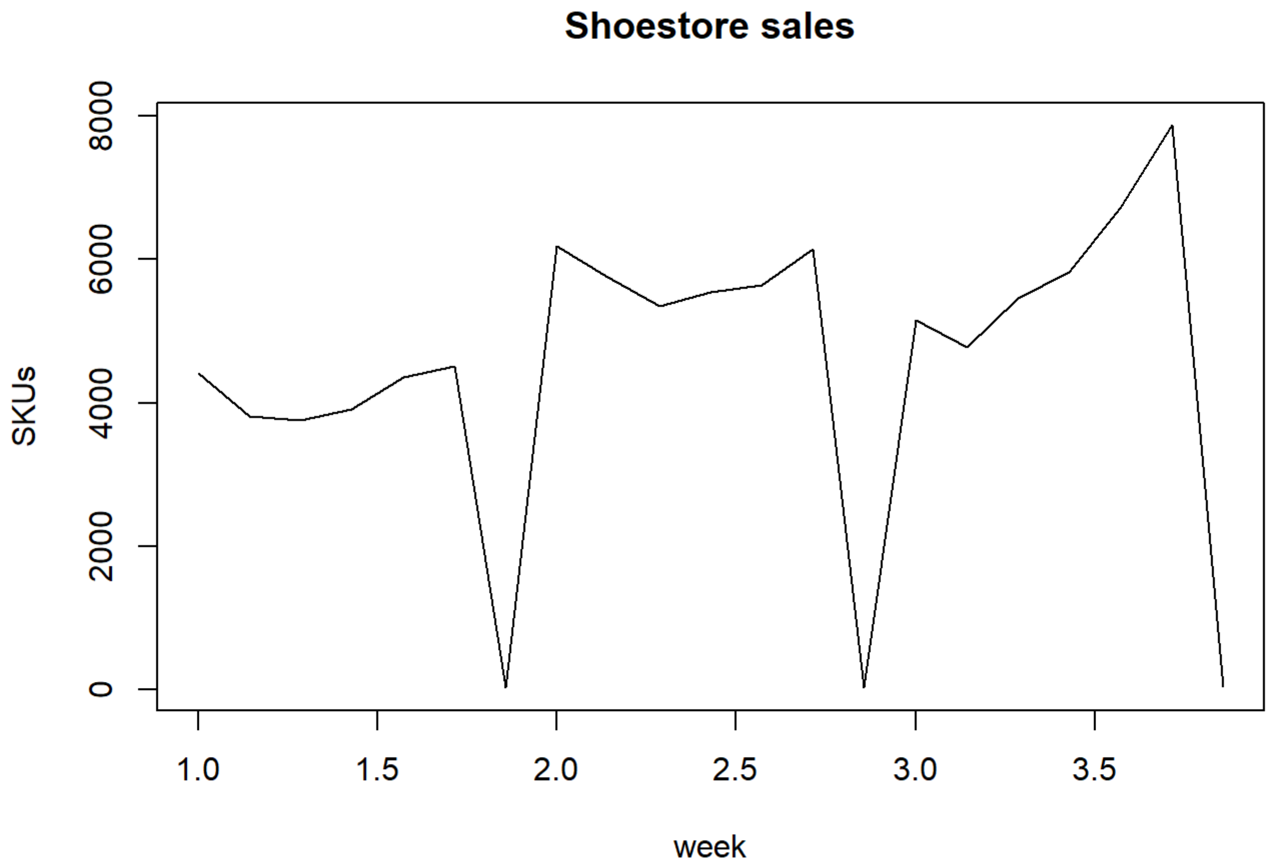

```
dema_fc <- forecast(dema, h = 10)
plot(dema_fc)
```



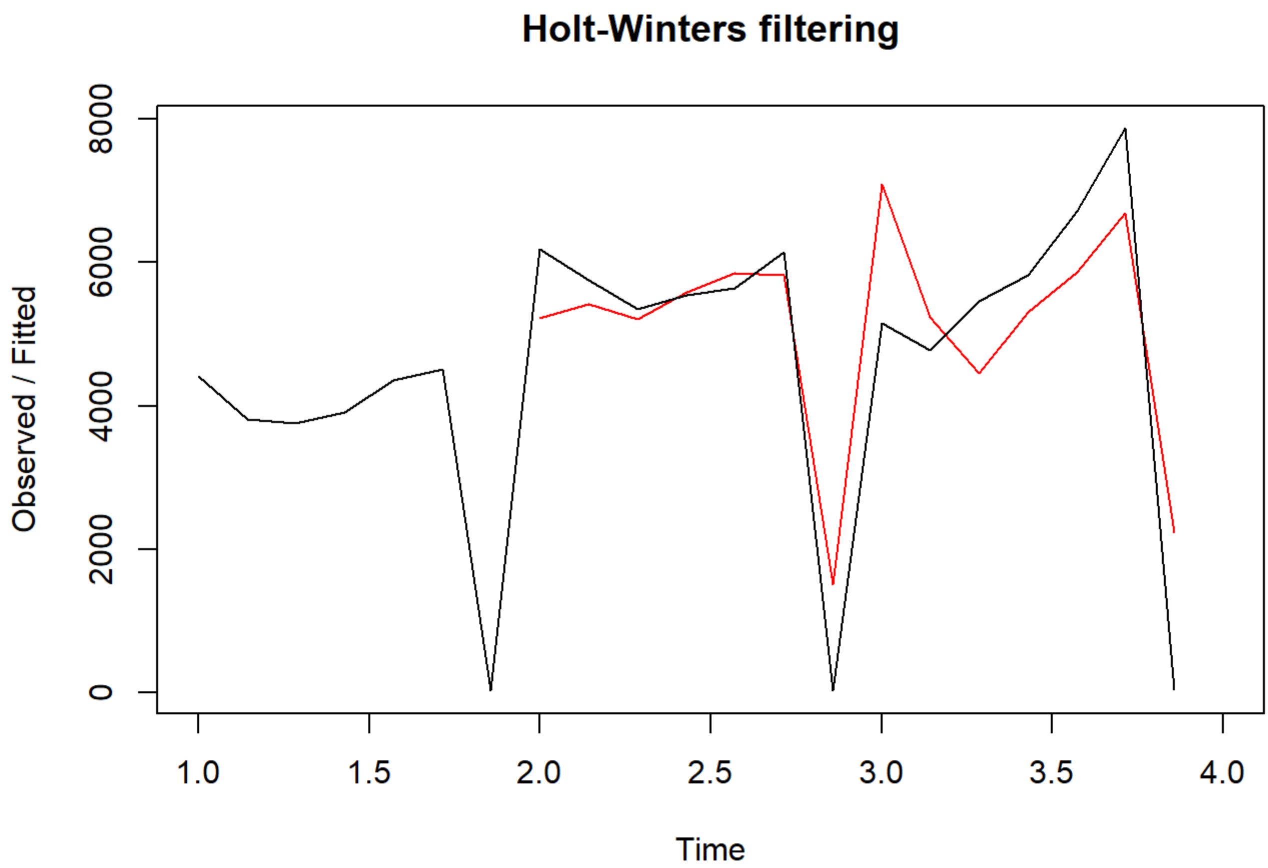
Holt-Winters Exponential Smoothing

Holt-Winter's method is also able to model seasonal trends in data.

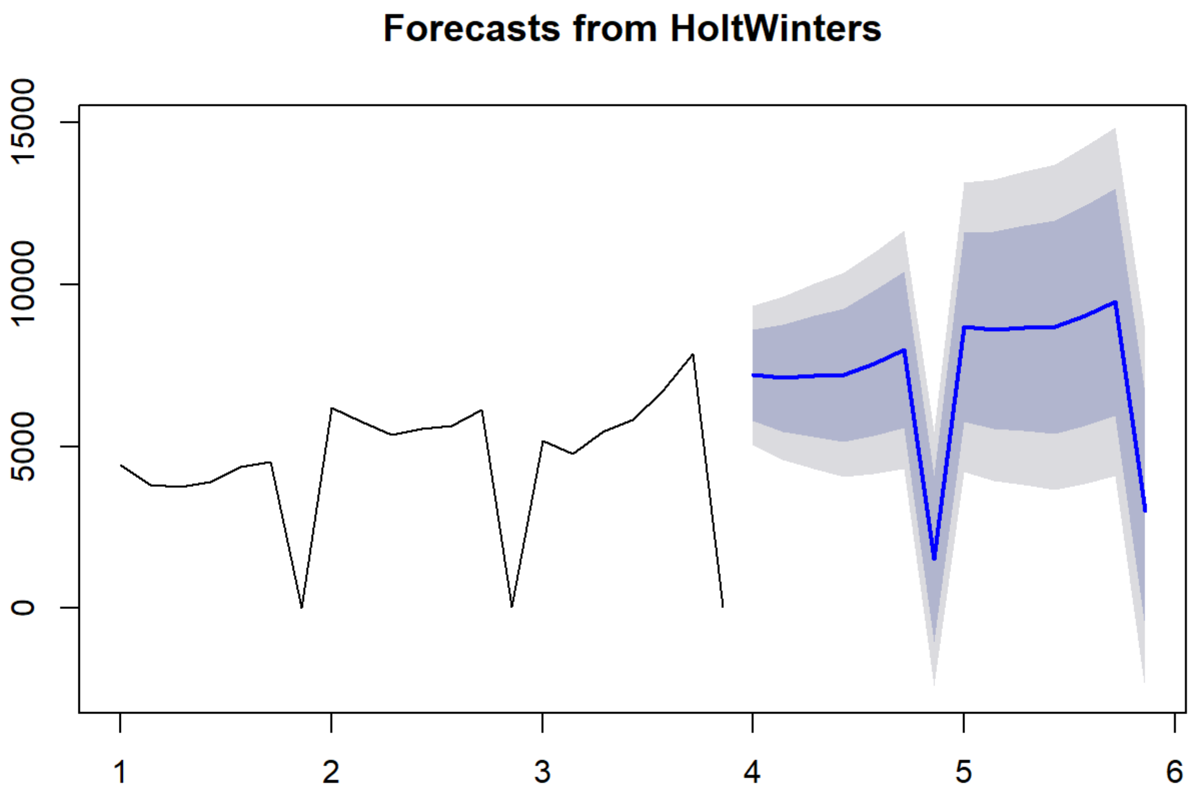
```
sales <- read.csv('shoestore-sales.csv')
sales_ts <- ts(sales$x_t, frequency = 7, start = c(1, 1))
plot.ts(sales_ts,
        main = 'Shoestore sales',
        xlab = 'week',
        ylab = 'SKUs')
```



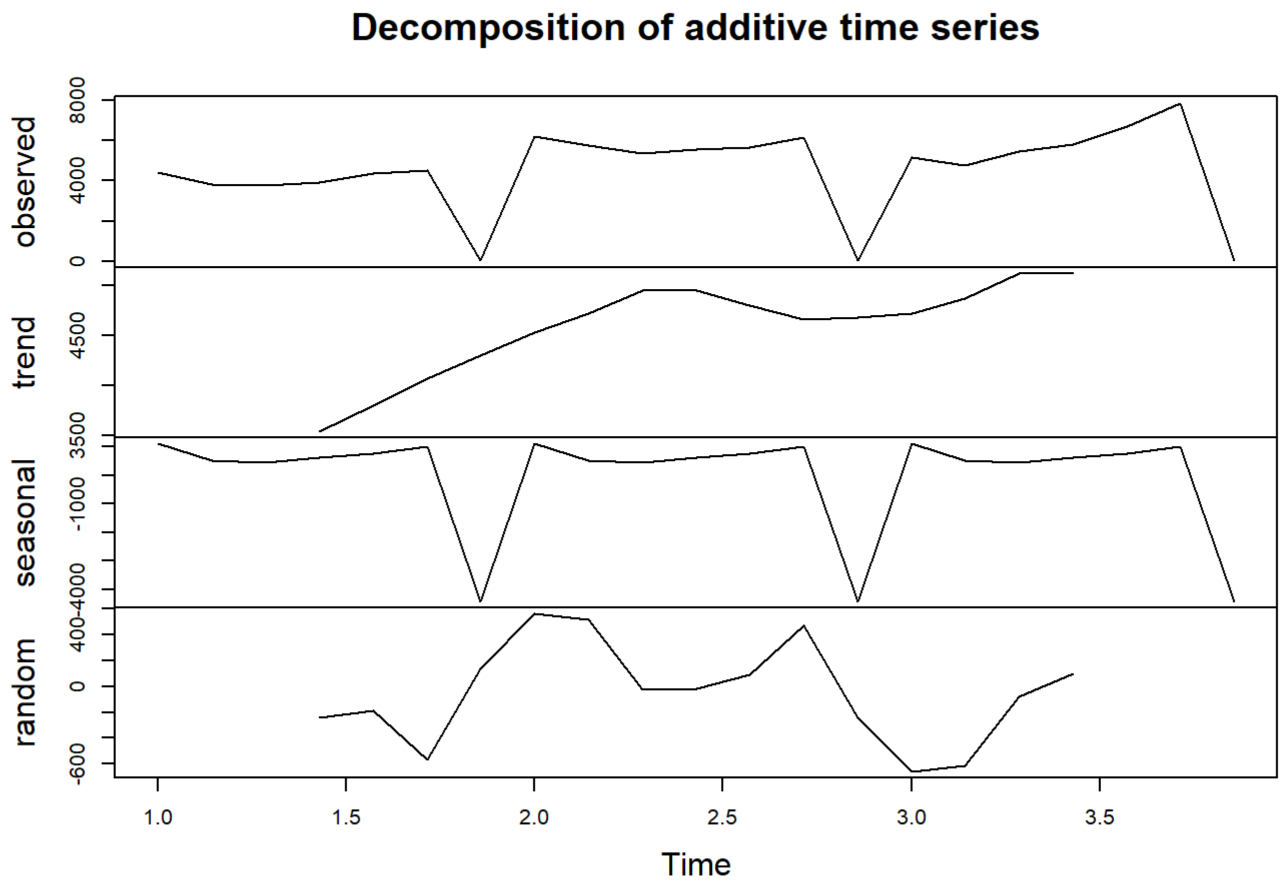

```
hw <- HoltWinters(sales_ts)
plot(hw, xlim = c(1, 4))
```



```
sales_fc <- forecast(hw, h = 14)
plot(sales_fc)
```




```
sales_decomposed <- decompose(sales_ts)
plot(sales_decomposed)
```


Decomposition of additive time series
observed
trend
seasonal
random
8000
4000
0
4500
3500
-1000
-4000
400
0
-600
1.0
1.5
2.0
2.5
3.0
3.5
Time