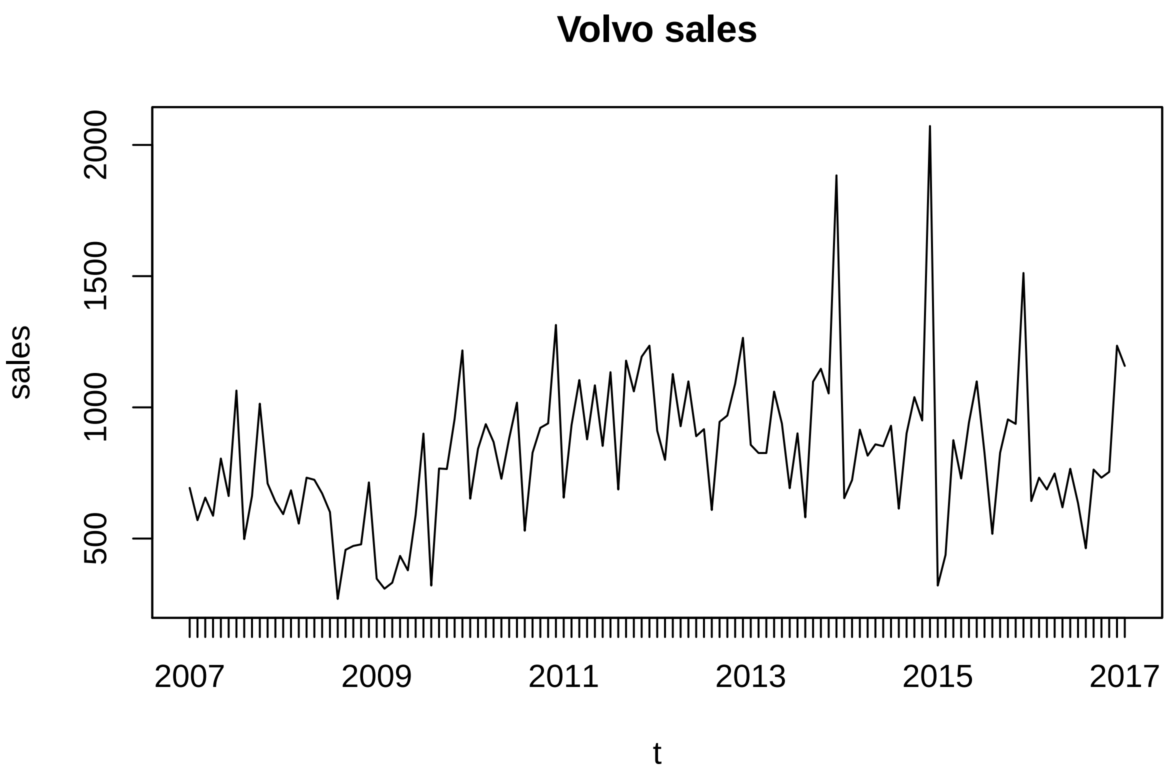
Volvo sales

Let's plot the time series of sales of Volvo. The sales are of Norway.

```
suppressMessages(library(zoo))
suppressMessages(library(forecast))
df = read.csv('datasets_830_1554_norway_new_car_sales_by_make.csv')
df = subset(df, Make == 'Volvo')
sales = zooreg(data = df$Quantity, as.yearmon("2007-01-01"), freq = 12)
N = length(sales)
t = c(1:N)
Q = factor(c(rep(c(1:12), N/12), c(1:(N%%12))))
plot(sales, main = "Volvo sales", xlab = "t")
```



The rest of our work will consist in fitting some models in a window of two years and trying to predict the next month. The plots will show the original and the predicted time series.

Regression

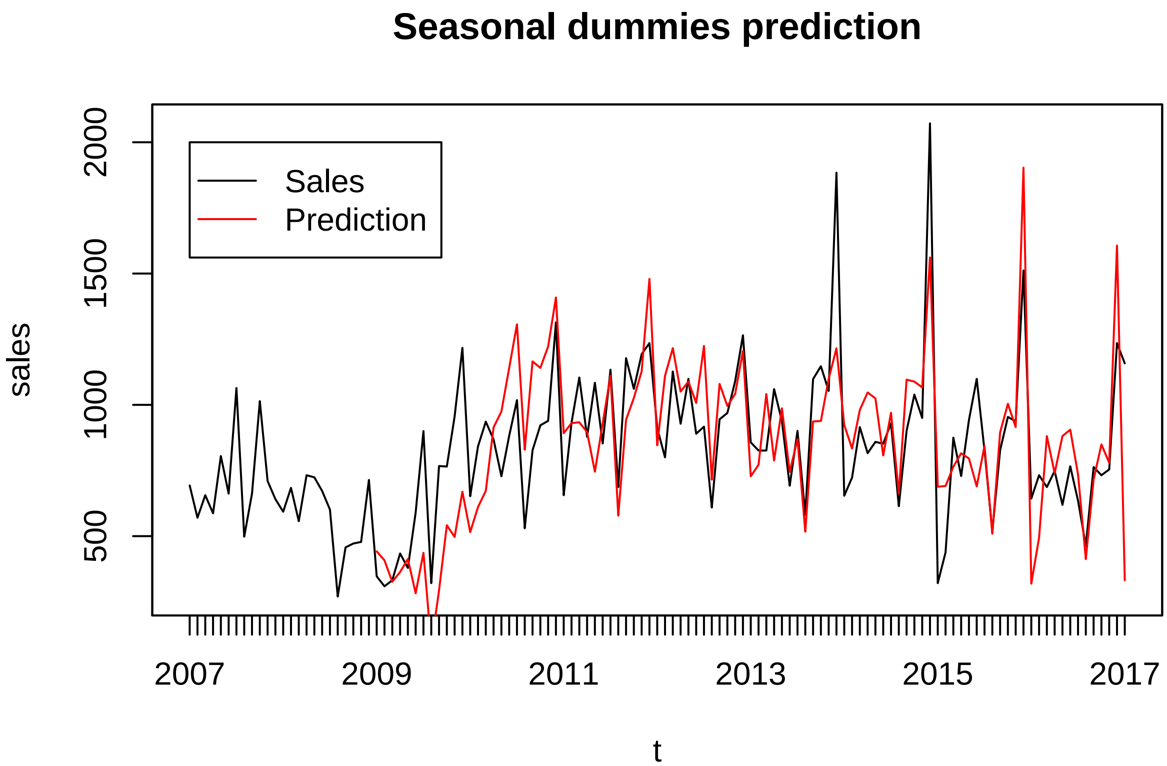
It seems reasonable to have seasonality of 12 months, so we will use it.

Seasonal dummies

We will now fit a regression model that consist of trend and seasonality coefficients, using seasonal summies too. Note how we apply the two years window.

```
next_step = function(t) {
  train = data.frame(
    t = t[1:24],
    sales = sales[t[1:24]],
```

```
    Q = Q[t[1:24]]
  )
  mod = lm(sales~t+Q, data = train)
  prediction = predict(mod, data.frame(t=t[25], Q=Q[t[25]]))
  return(prediction)
}
prediction = rollapply(t, 25, next_step)
plot(sales, main = "Seasonal dummies prediction", xlab = "t")
lines(time(sales)[25:length(t)], prediction, col = "red")
legend(
  x = time(sales)[1],
  y = 2000,
  legend = c('Sales', 'Prediction'),
  col = c('black', 'red'),
  pch = c('', ''),
  lty = c(1, 1)
)
```




```
mape = mean(abs((prediction - sales[25:length(sales)])/sales[25:length(sales)]))
paste("MAPE =", mape)
```

```
## [1] "MAPE = 0.214958623085887"
```

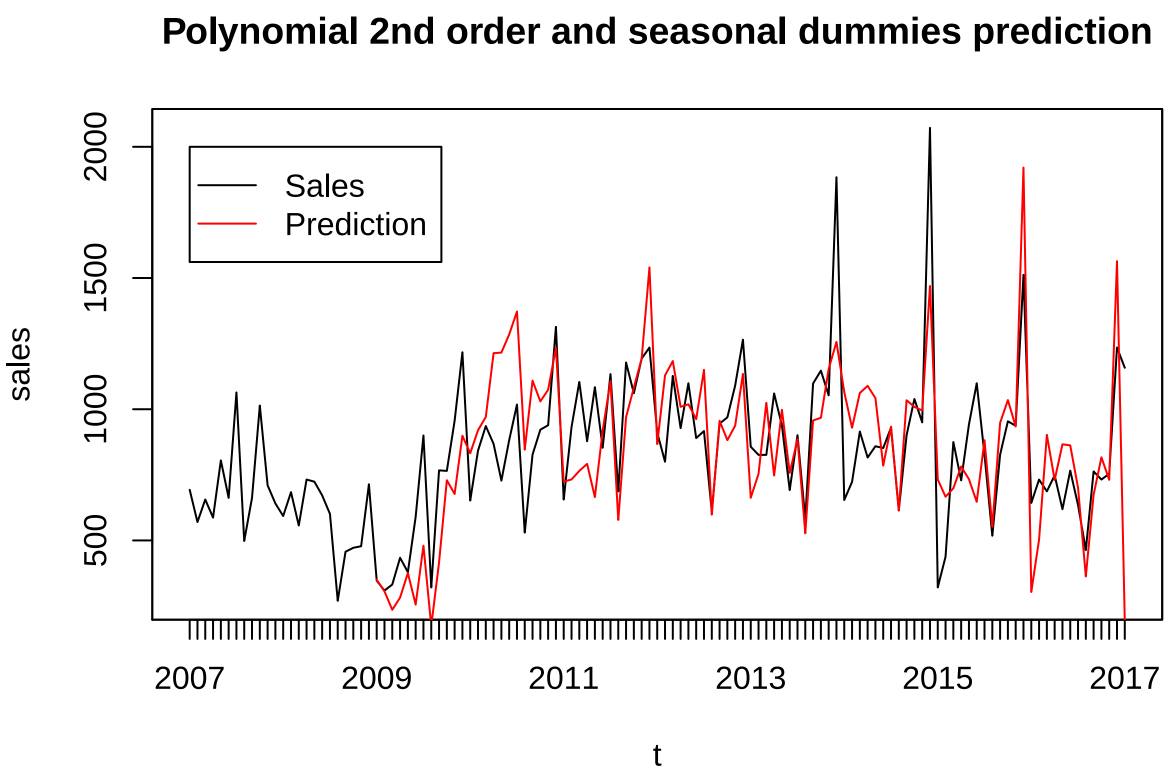
We see a reasonable MAPE and plot.

Polynomial

Now we add a two degree polynomial to the last model.

```
library(zoo)
next_step = function(t) {
```

```r
  train = data.frame(
    t = t[1:24],
    sales = sales[t[1:24]],
    Q = Q[t[1:24]]
  )
  mod = lm(sales~poly(t, 2)+Q, data = train)
  prediction = predict(mod, data.frame(t=t[25], Q=Q[t[25]]))
  return(prediction)
}
prediction = rollapply(t, 25, next_step)
plot(sales, main = "Polynomial 2nd order and seasonal dummies prediction", xlab = "t")
lines(time(sales)[25:length(t)], prediction, col = "red")
legend(
  x = time(sales)[1],
  y = 2000,
  legend = c('Sales', 'Prediction'),
  col = c('black', 'red'),
  pch = c('', ''),
  lty = c(1, 1)
)
```




```r
mape = mean(abs((prediction - sales[25:length(sales)])/sales[25:length(sales)]))
paste("MAPE =", mape)
```

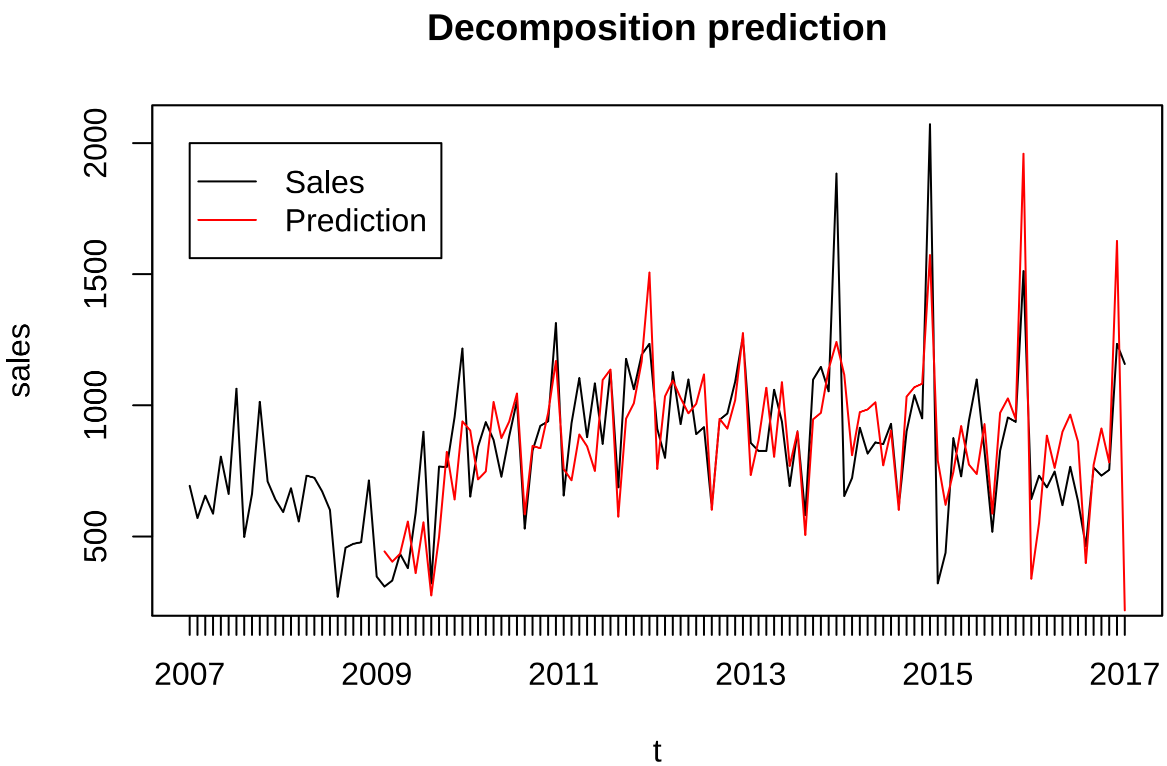
```
## [1] "MAPE = 0.213598454013614"
```

The results are close to the last one.

Decomposition

We will now use a function to decompose our series.

```
next_step = function(t) {
  mod = stl(sales[t[1:25]], s.window = "periodic")
  prediction = forecast(mod, h=1)
  return(as.numeric(prediction$mean))
}
prediction = rollapply(t, 26, next_step)
plot(sales, main = "Decomposition prediction", xlab = "t")
lines(time(sales)[26:length(t)], prediction, col = "red")
legend(
  x = time(sales)[1],
  y = 2000,
  legend = c('Sales', 'Prediction'),
  col = c('black', 'red'),
  pch = c('', ''),
  lty = c(1, 1)
)
```




```
mape = mean(abs((prediction - sales[26:length(sales)])/sales[26:length(sales)]))
paste("MAPE =", mape)
```

```
## [1] "MAPE = 0.195302561502996"
```

Interesting results.

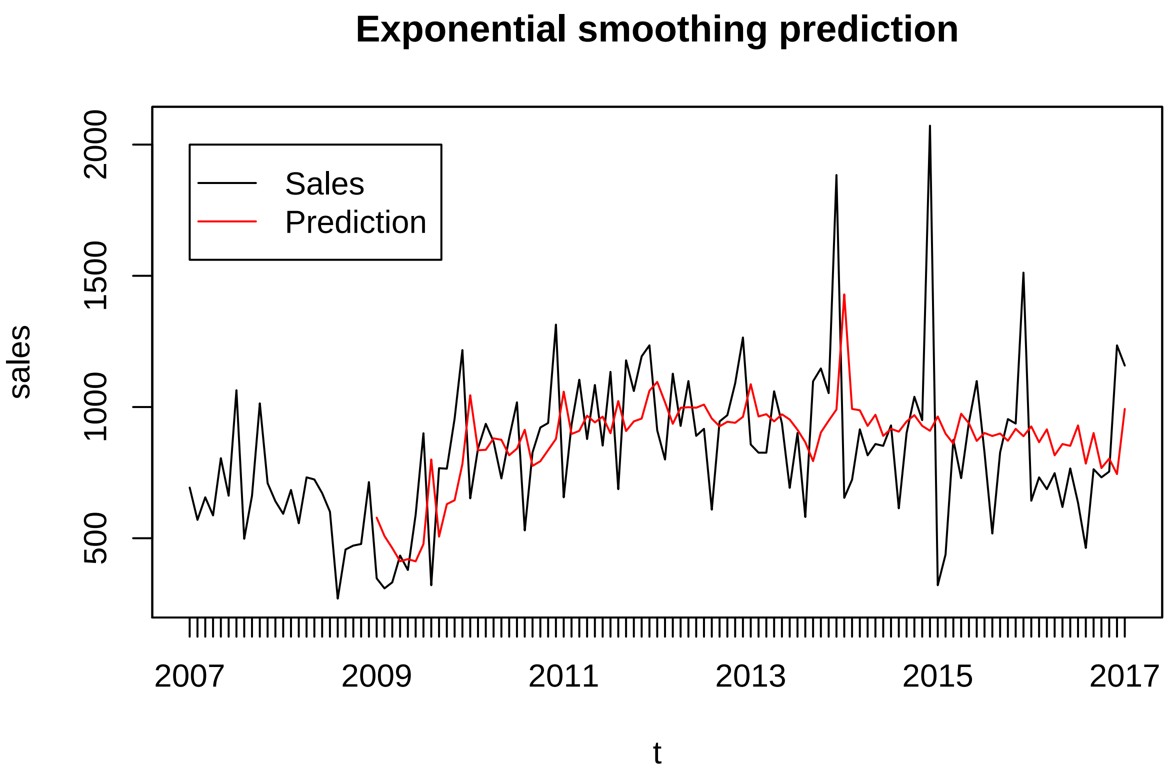
Exponential smoothing

Now we will fit some models based on exponential smoothing.

Exponential smoothing

This is the basic exponential smoothing model. It only considerates constant mean.

```
next_step = function(t) {
  train = data.frame(
    t = t[1:24],
    sales = sales[t[1:24]],
    Q = Q[t[1:24]]
  )
  mod = HoltWinters(train$sales, beta = F, gamma = F)
  prediction = forecast(mod, 1)
  return(prediction$mean)
}
prediction = rollapply(t, 25, next_step)
plot(sales, main = "Exponential smoothing prediction", xlab = "t")
lines(time(sales)[25:length(t)], prediction, col = "red")
legend(
  x = time(sales)[1],
  y = 2000,
  legend = c('Sales', 'Prediction'),
  col = c('black', 'red'),
  pch = c('', ''),
  lty = c(1, 1)
)
```




```
mape = mean(abs((prediction - sales[25:length(sales)])/sales[25:length(sales)]))
paste("MAPE =", mape)
```

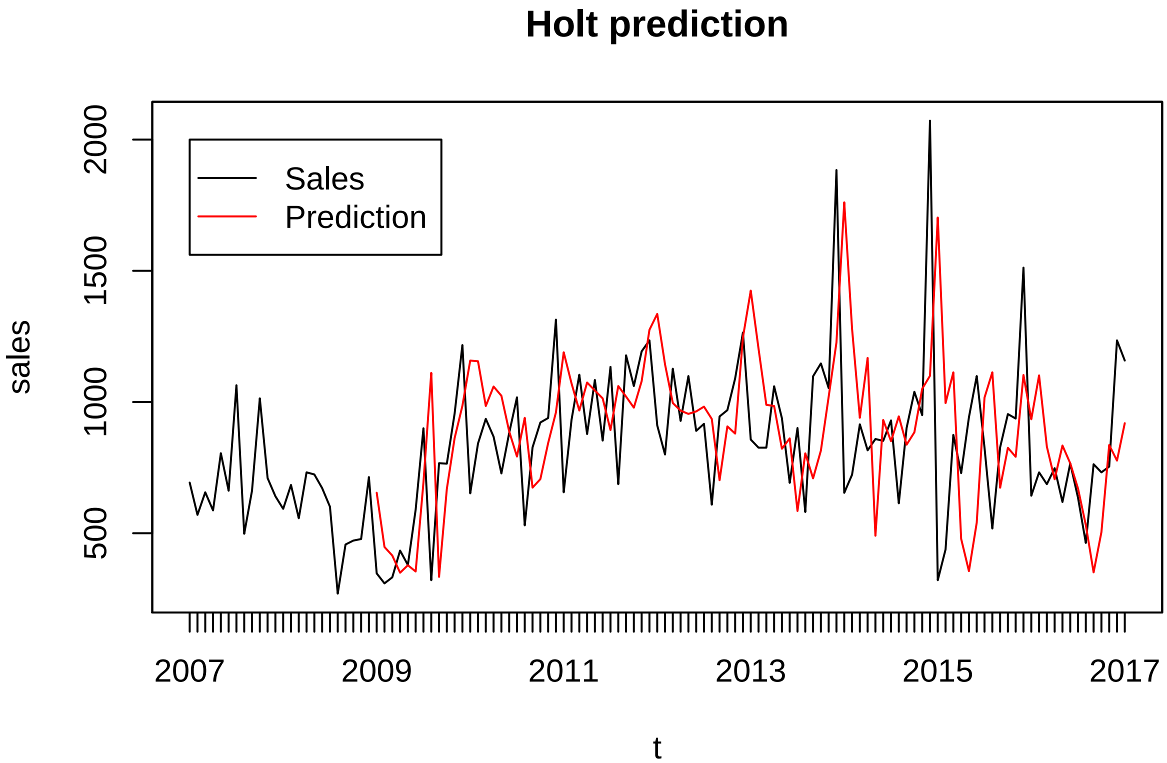
```
## [1] "MAPE = 0.277045393426693"
```

The MAPE is worse than before.

Holt

Now we have trend in our model.

```
next_step = function(t) {
  train = data.frame(
    t = t[1:24],
    sales = sales[t[1:24]],
    Q = Q[t[1:24]]
  )
  mod = HoltWinters(train$sales, beta = T, gamma = F)
  prediction = forecast(mod, 1)
  return(prediction$mean)
}
prediction = rollapply(t, 25, next_step)
plot(sales, main = "Holt prediction", xlab = "t")
lines(time(sales)[25:length(t)], prediction, col = "red")
legend(
  x = time(sales)[1],
  y = 2000,
  legend = c('Sales', 'Prediction'),
  col = c('black', 'red'),
  pch = c('', ''),
  lty = c(1, 1)
)
```




```
mape = mean(abs((prediction - sales[25:length(sales)])/sales[25:length(sales)]))
paste("MAPE =", mape)
```

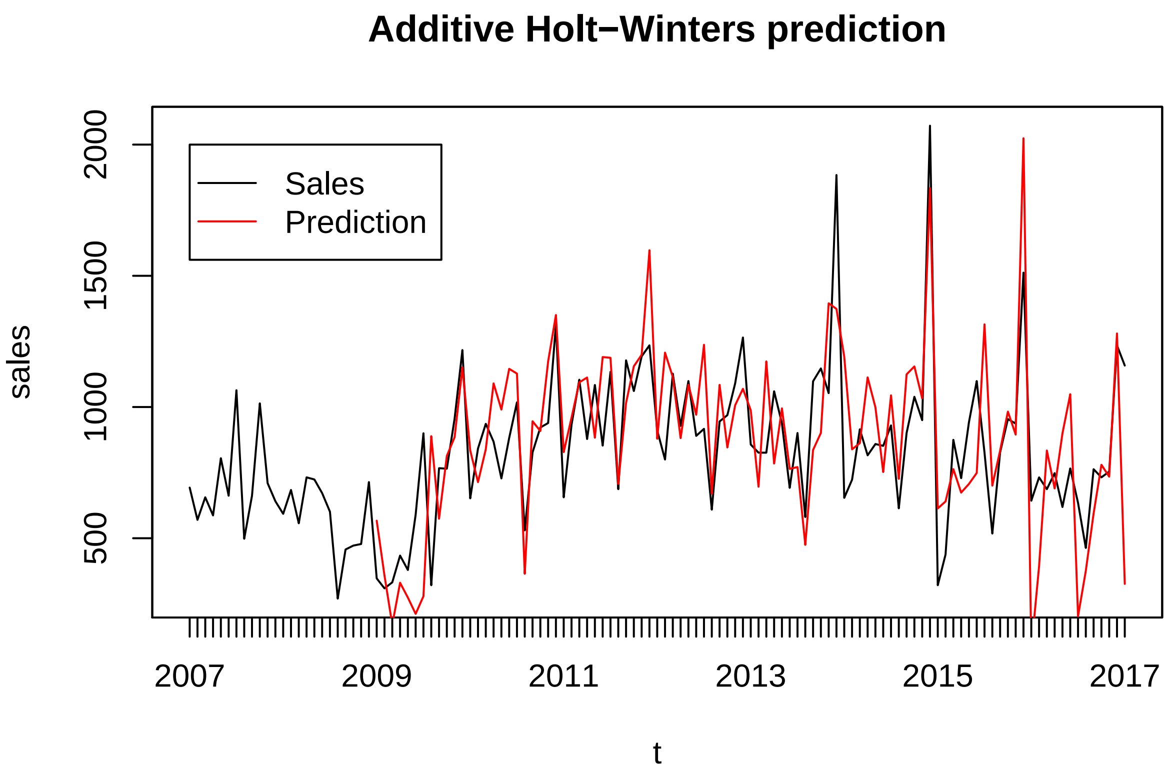
```
## [1] "MAPE = 0.367475797650731"
```

It's even worse.

Additive Holt-Winters

We will now consider the complete Holt-Winters, with seasonality. This time it's additive.

```
next_step = function(t) {
  train = data.frame(
    t = t[1:24],
    sales = sales[t[1:24]],
    Q = Q[t[1:24]]
  )
  mod = HoltWinters(ts(train$sales, frequency = 12), beta = T, gamma = T, seasonal = "additive")
  prediction = forecast(mod, 1)
  return(prediction$mean)
}
prediction = rollapply(t, 25, next_step)
plot(sales, main = "Additive Holt-Winters prediction", xlab = "t")
lines(time(sales)[25:length(t)], prediction, col = "red")
legend(
  x = time(sales)[1],
  y = 2000,
  legend = c('Sales', 'Prediction'),
  col = c('black', 'red'),
  pch = c('', ''),
  lty = c(1, 1)
)
```




```
mape = mean(abs((prediction - sales[25:length(sales)])/sales[25:length(sales)]))
paste("MAPE =", mape)
```

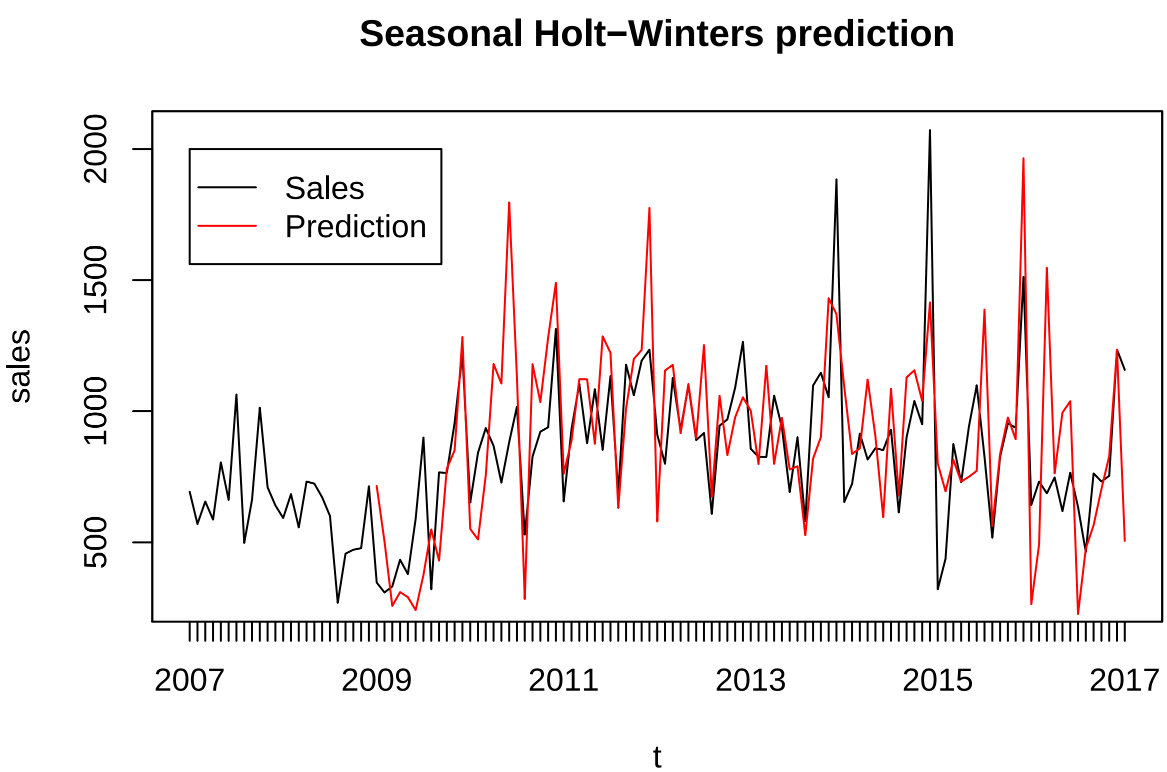
```
## [1] "MAPE = 0.247905784030637"
```

We see improvements.

Multiplicative Holt-Winters

Now the model is multiplicative.

```
next_step = function(t) {
  train = data.frame(
    t = t[1:24],
    sales = sales[t[1:24]],
    Q = Q[t[1:24]]
  )
  mod = HoltWinters(ts(train$sales, frequency = 12), beta = T, gamma = T, seasonal = "multiplicative")
  prediction = forecast(mod, 1)
  return(prediction$mean)
}
prediction = rollapply(t, 25, next_step)
plot(sales, main = "Seasonal Holt-Winters prediction", xlab = "t")
lines(time(sales)[25:length(t)], prediction, col = "red")
legend(
  x = time(sales)[1],
  y = 2000,
  legend = c('Sales', 'Prediction'),
  col = c('black', 'red'),
  pch = c('', ''),
  lty = c(1, 1)
)
```




```
mape = mean(abs((prediction - sales[25:length(sales)])/sales[25:length(sales)]))
paste("MAPE =", mape)
```

```
## [1] "MAPE = 0.276258405722419"
```

It's algo good.