# Kalman filter(Partial Pooling)

*LinyiGuo*

*2019/8/11*

```
rm(list = ls())
set.seed(9483)
```

Data is from NIKE and ADIDAS.

## Data Understanding & Arima Analysis

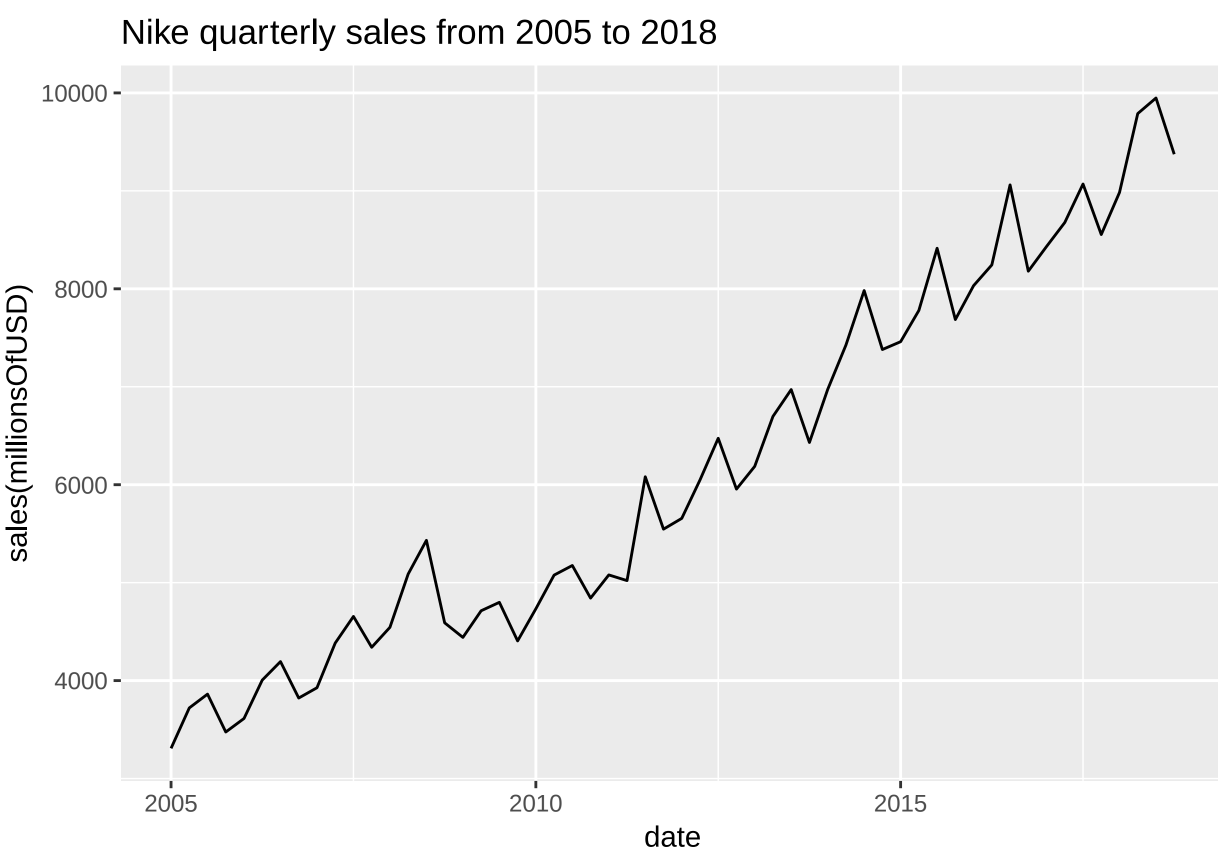
```
library(ggfortify)
```

```
## Loading required package: ggplot2

## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

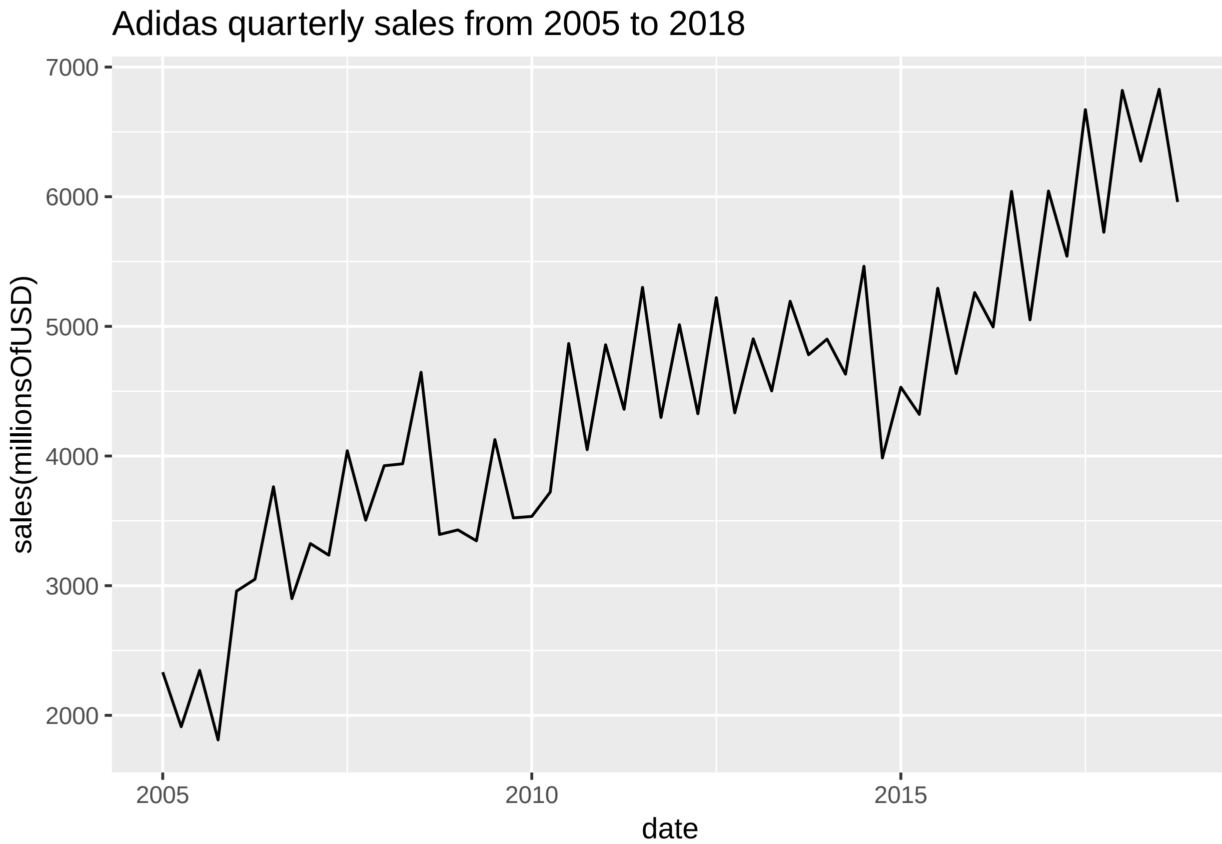
```
library(forecast)
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Registered S3 methods overwritten by 'forecast':
##   method                 from
##   autoplot.Arima         ggfortify
##   autoplot.acf           ggfortify
##   autoplot.ar            ggfortify
##   autoplot.bats          ggfortify
##   autoplot.decomposed.ts ggfortify
##   autoplot.ets           ggfortify
##   autoplot.forecast      ggfortify
##   autoplot.stl           ggfortify
##   autoplot.ts            ggfortify
##   fitted.ar              ggfortify
##   fitted.fracdiff        fracdiff
##   fortify.ts             ggfortify
##   residuals.ar           ggfortify
##   residuals.fracdiff     fracdiff
```

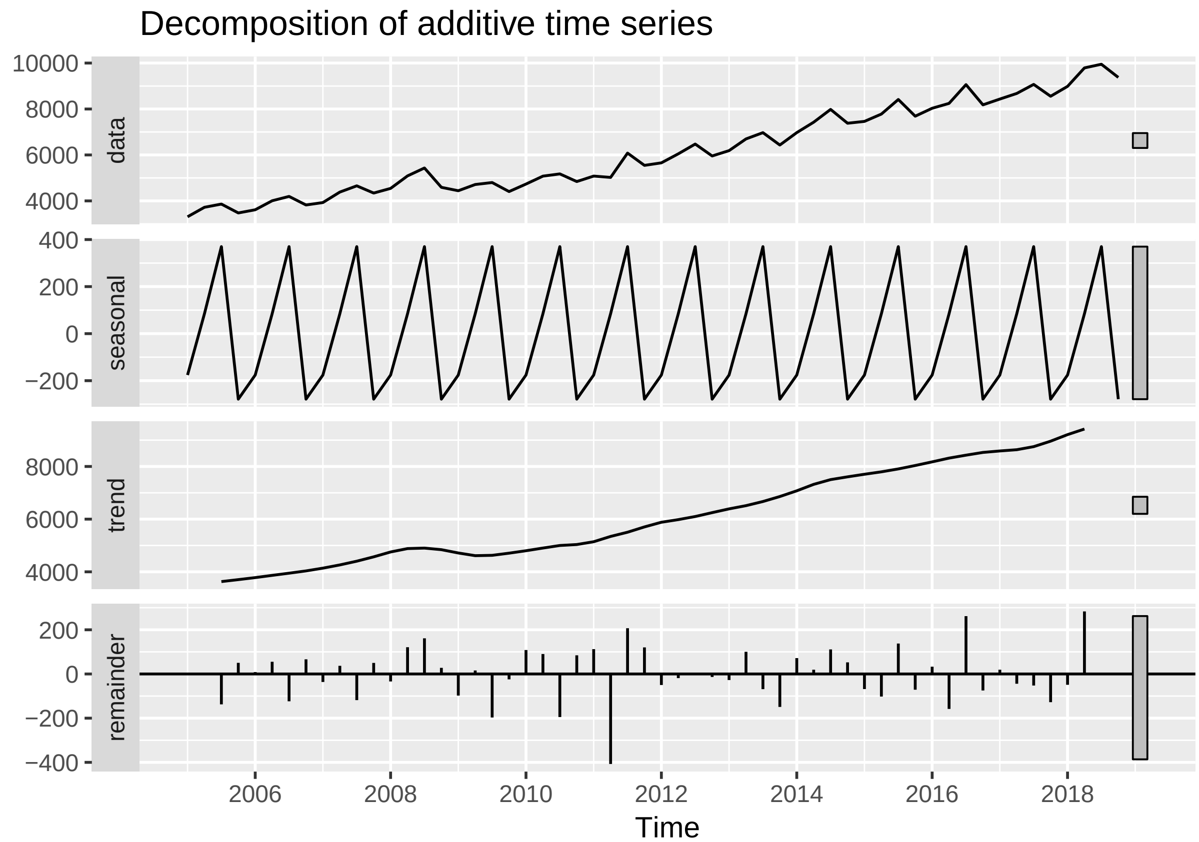
```r
library(KFAS)
library(tseries)

# Load Data
data_nike <- read.csv('C:\\Users\\GuoLY\\Desktop\\markdown\\nike.csv',header = TRUE)
data_adi <- read.csv('C:\\Users\\GuoLY\\Desktop\\markdown\\adidas.csv',header = TRUE)

data_nike <- ts(rev(data_nike[3:58,2]), frequency=4, start=c(2005,1))
data_adi <- ts(rev(data_adi[2:57,2]), frequency=4, start=c(2005,1))

# Visualization
autoplot(data_nike) + labs(x='date', y='sales(millionsOfUSD)',
                           title = 'Nike quarterly sales from 2005 to 2018')
```



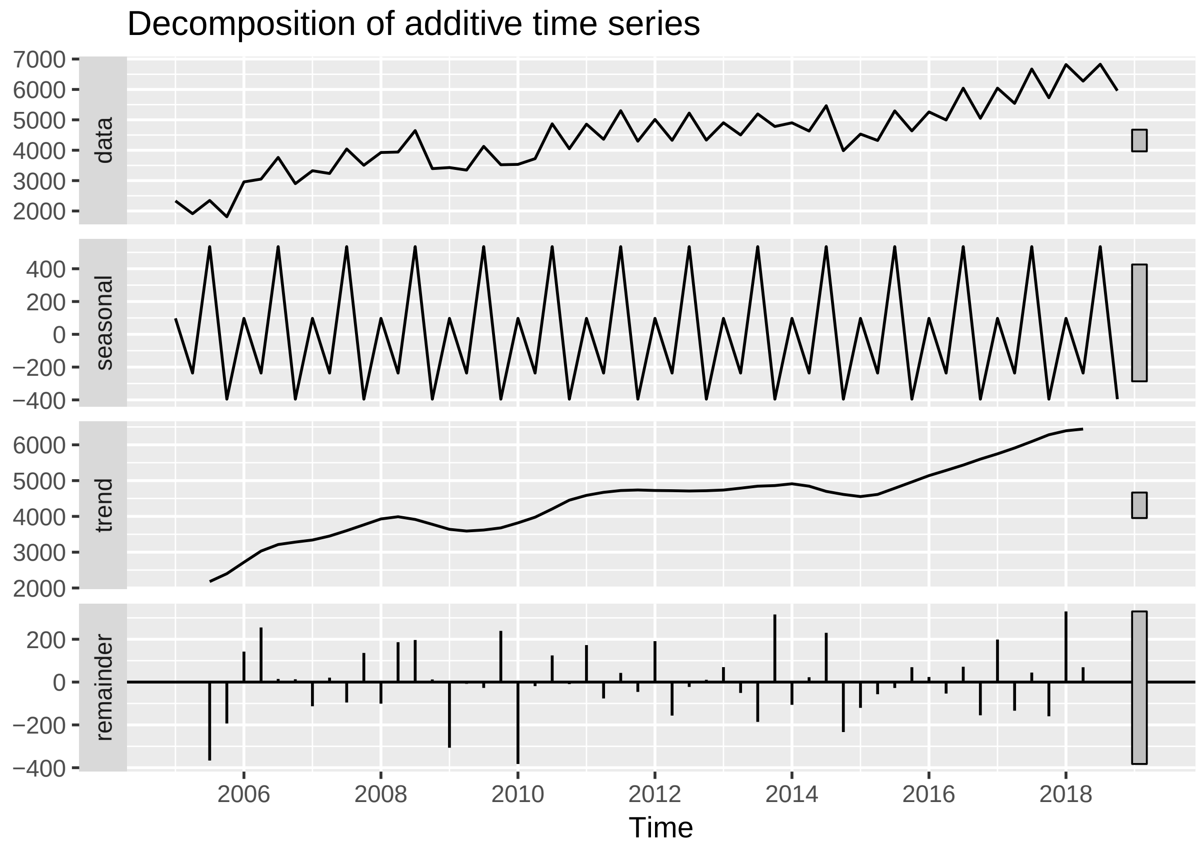

```r
autoplot(data_adi) + labs(x='date', y='sales(millionsOfUSD)',
                         title = 'Adidas quarterly sales from 2005 to 2018')
```



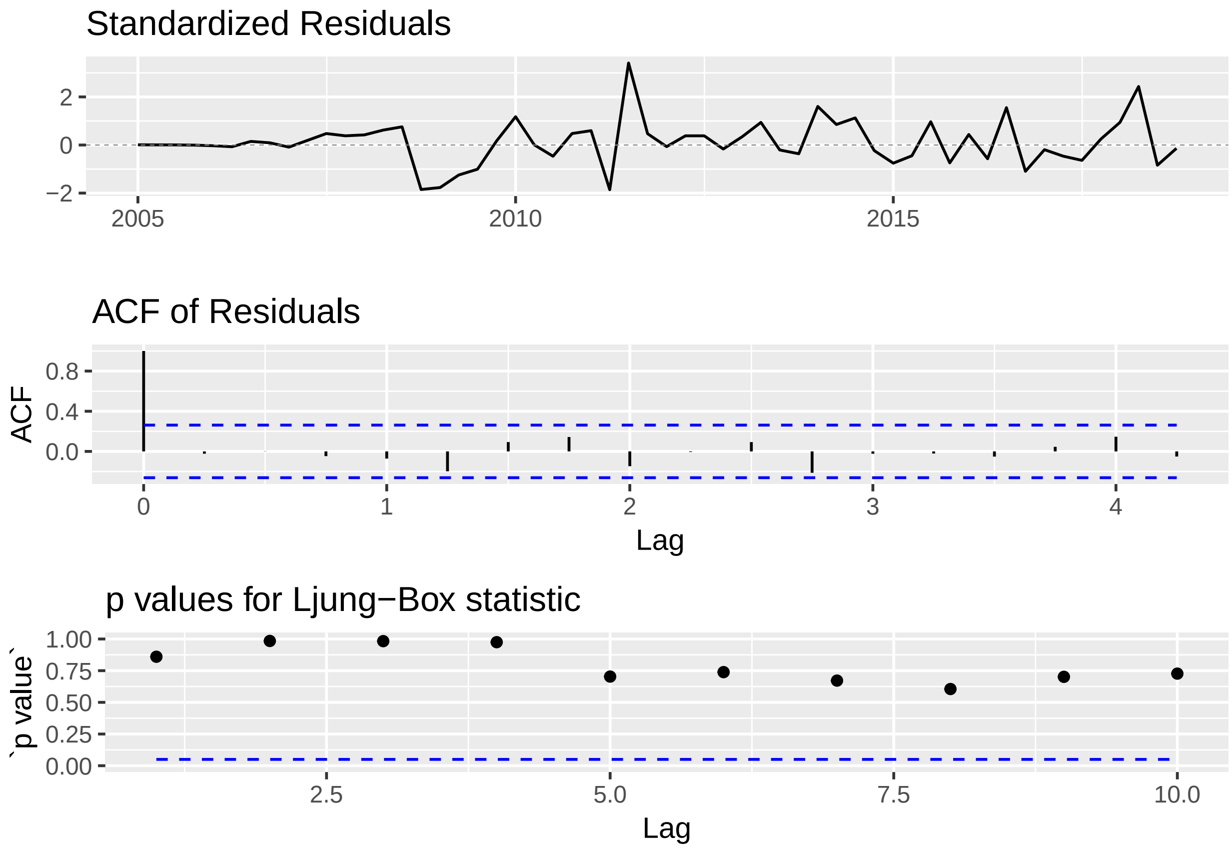

```
# Time Series Decomposition
decompose_nike <- decompose(data_nike, "additive")
autoplot(decompose_nike)
```



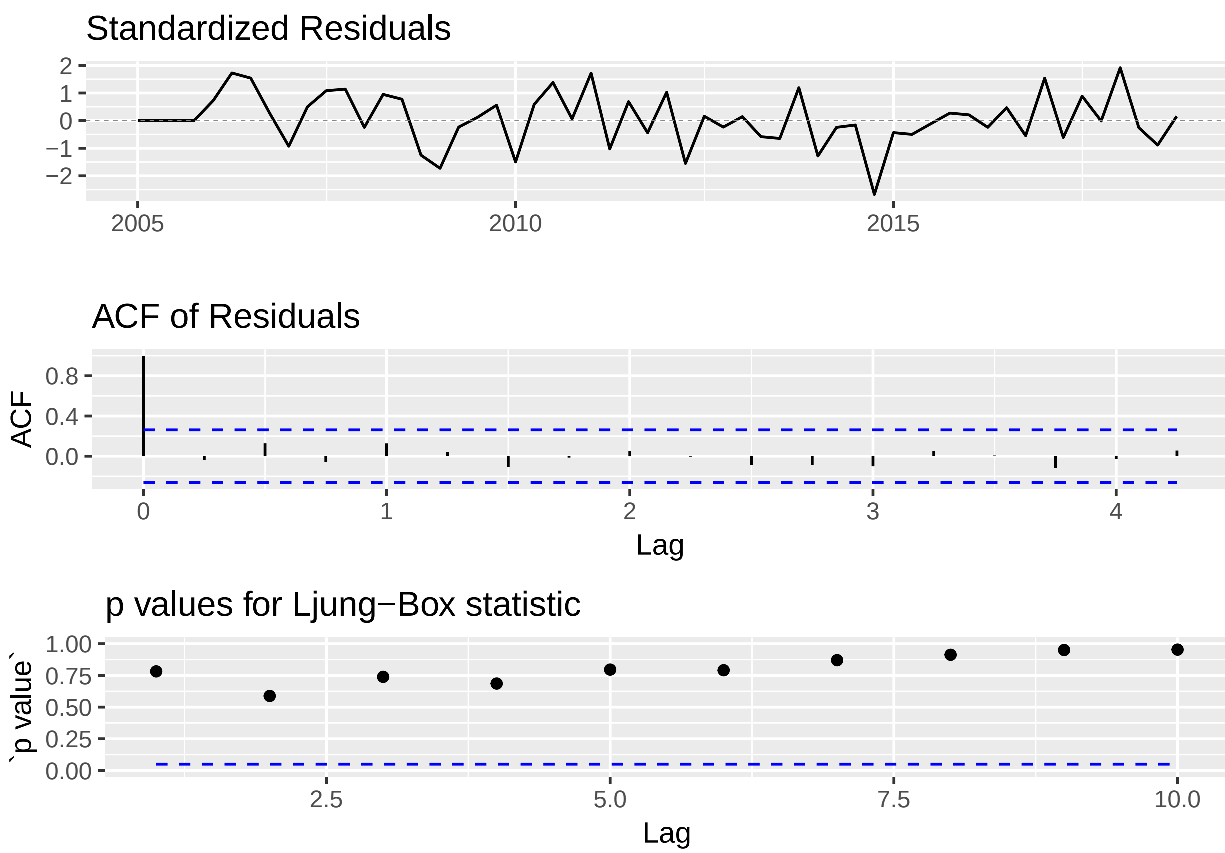

```
decompose_adi <- decompose(data_adi, "additive")
autoplot(decompose_adi)
```




```r
# Fit Arima model
arima_nike <- auto.arima(data_nike)
arima_adi <- auto.arima(data_adi)

# Model diagnostic
ggtsdiag(arima_nike)
```




```r
ggtsdiag(arima_adi)
```




```
# correlation between trends
# two trends have strong positive correlation 0.9211
cor(decompose_nike$trend[3:54], decompose_adi$trend[3:54])
```

```
## [1] 0.9210953
```

```
# correlation between seasonal
# positive correlation 0.7563
cor(decompose_nike$seasonal,decompose_adi$seasonal)
```

```
## [1] 0.7563281
```

The residuals:

- seem like white noise around 0
- have no significant correlations
- p values are well above 0.05

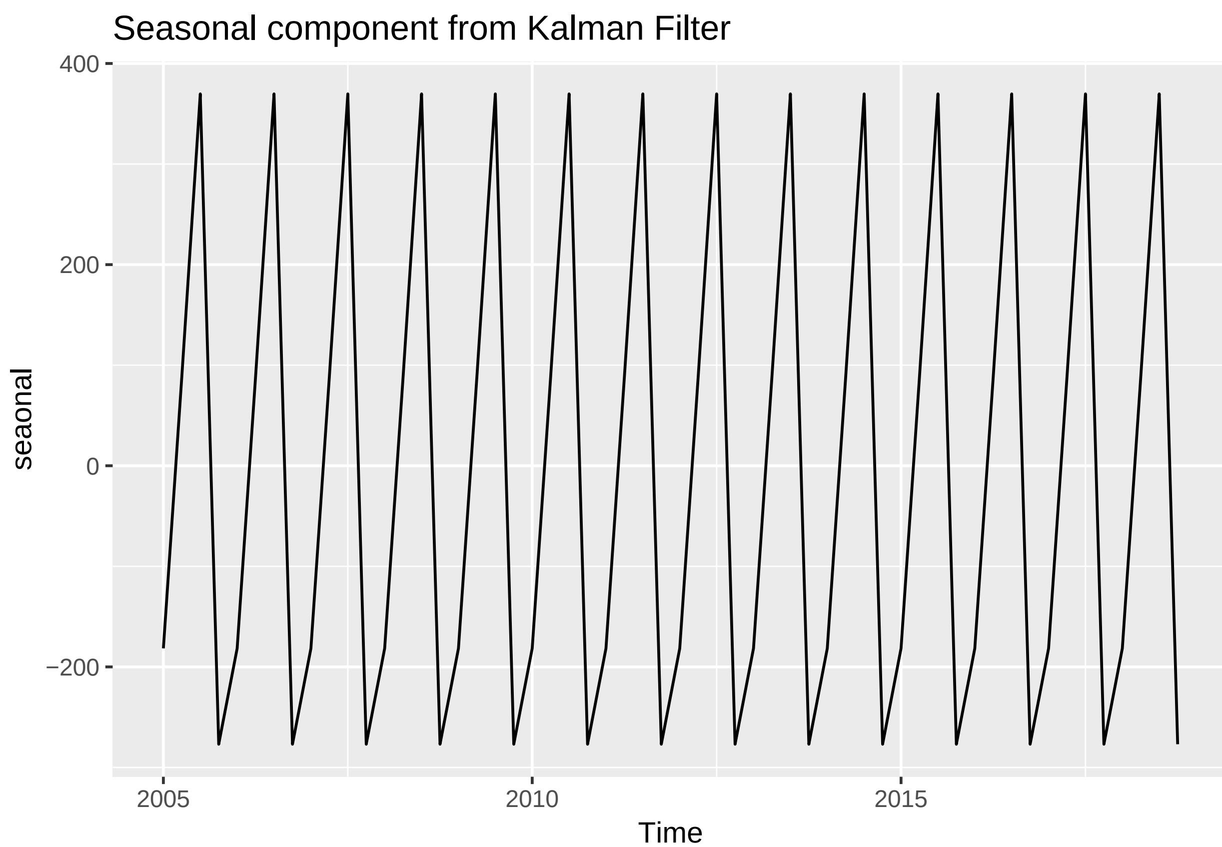
*Comment:* Ljung-Box is used to test the independence, and the hypothesis being tested is that the residuals from the ARIMA model have no autocorrelation.

# Kalman Filter

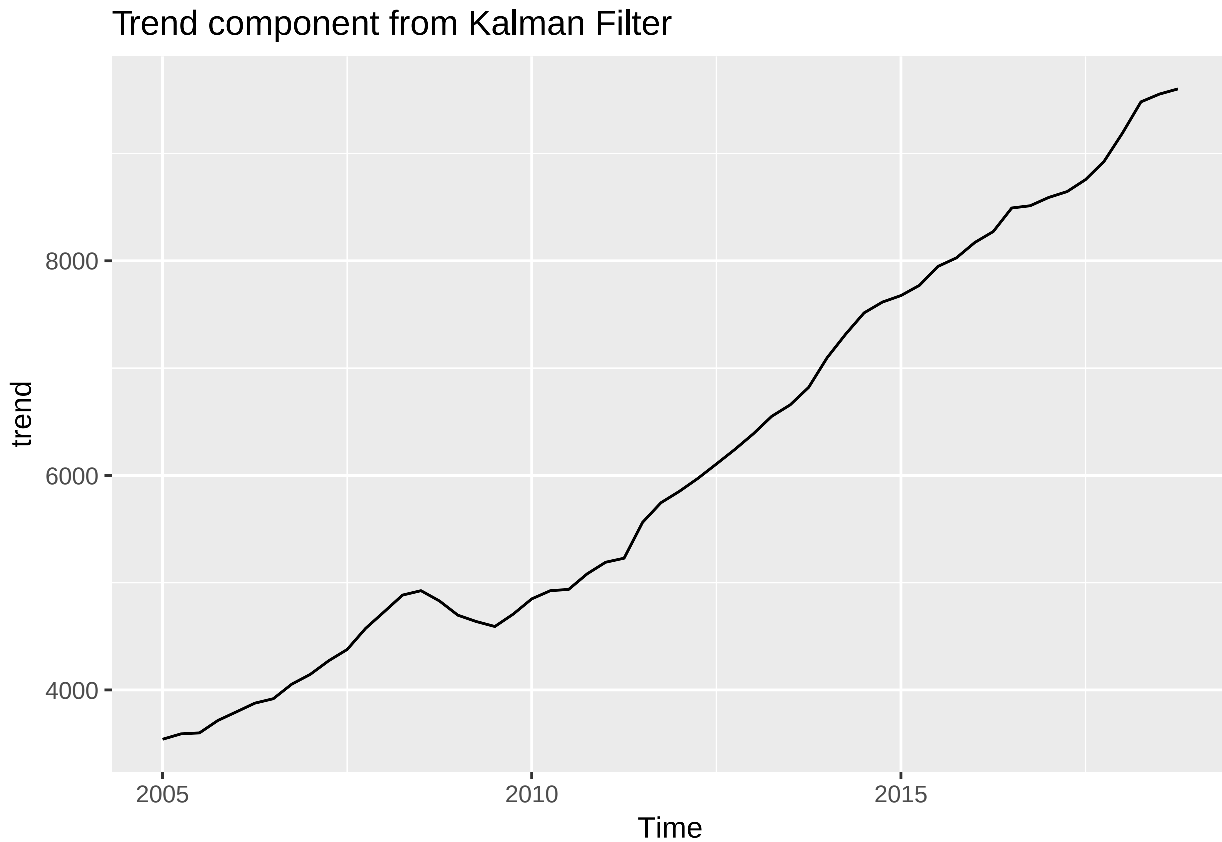
## Analyse separately

### NIKE

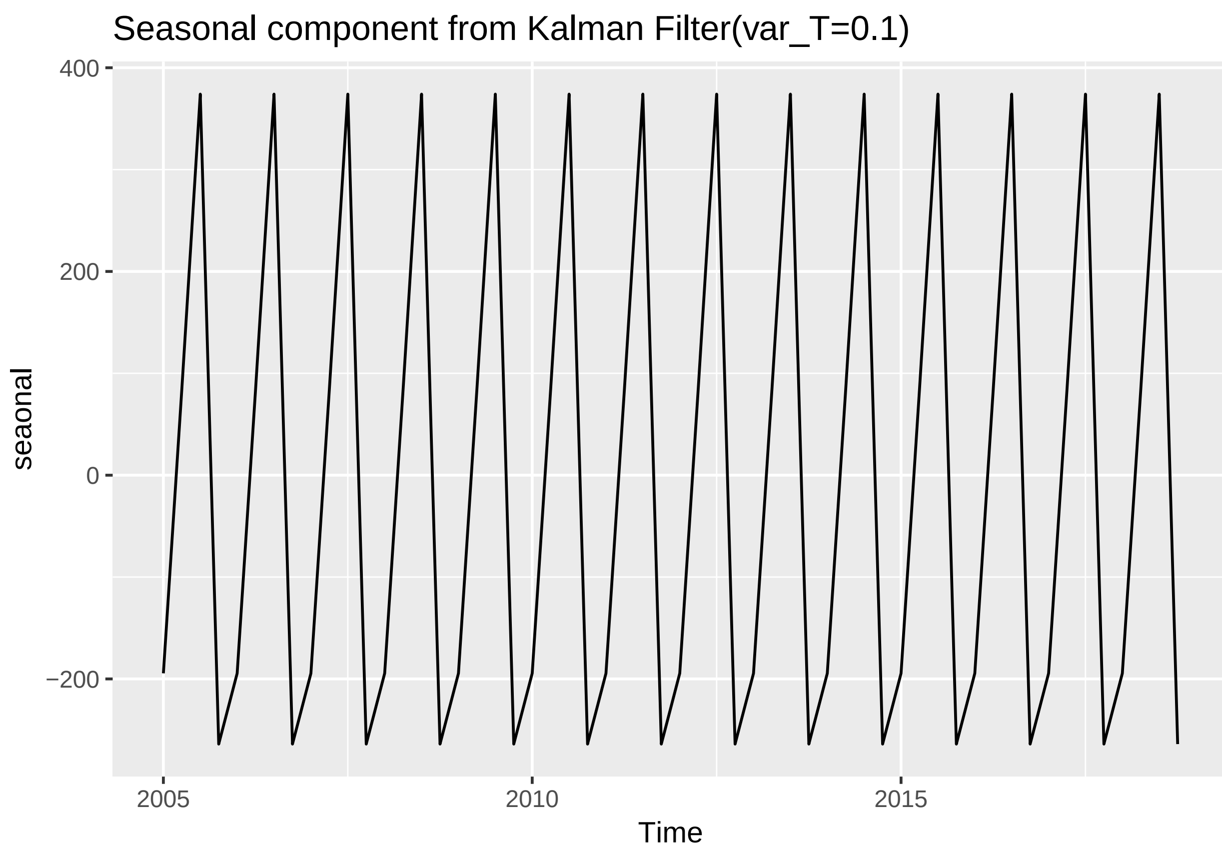
```
mod_nike <- SSModel(data_nike ~ SSMtrend(1,Q=list(1)) +
                      SSMseasonal(4,sea.type = "trigonometric"), H = 1)
out_nike <- KFS(mod_nike, filtering = "state")
out_nikeSmooth <- KFS(mod_nike, smoothing = "state")
autoplot(ts(rowSums(coef(out_nike, states = "seasonal")[,c(1,3)]),
            frequency = 4, start = c(2005,1))) +
  labs(title = 'Seasonal component from Kalman Filter', y = 'seaonal')
```



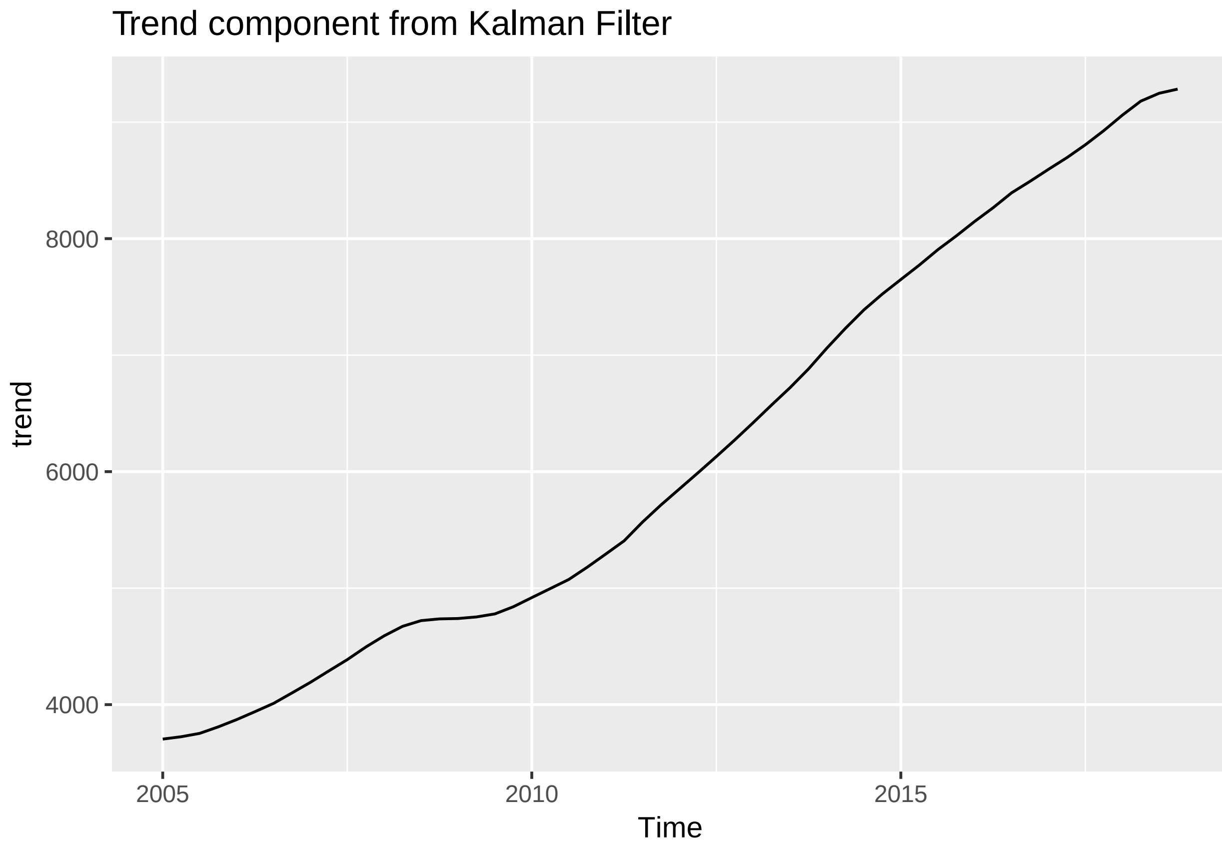

```
autoplot(coef(out_nike, states = "trend")) +
  labs(title = 'Trend component from Kalman Filter', y = 'trend')
```



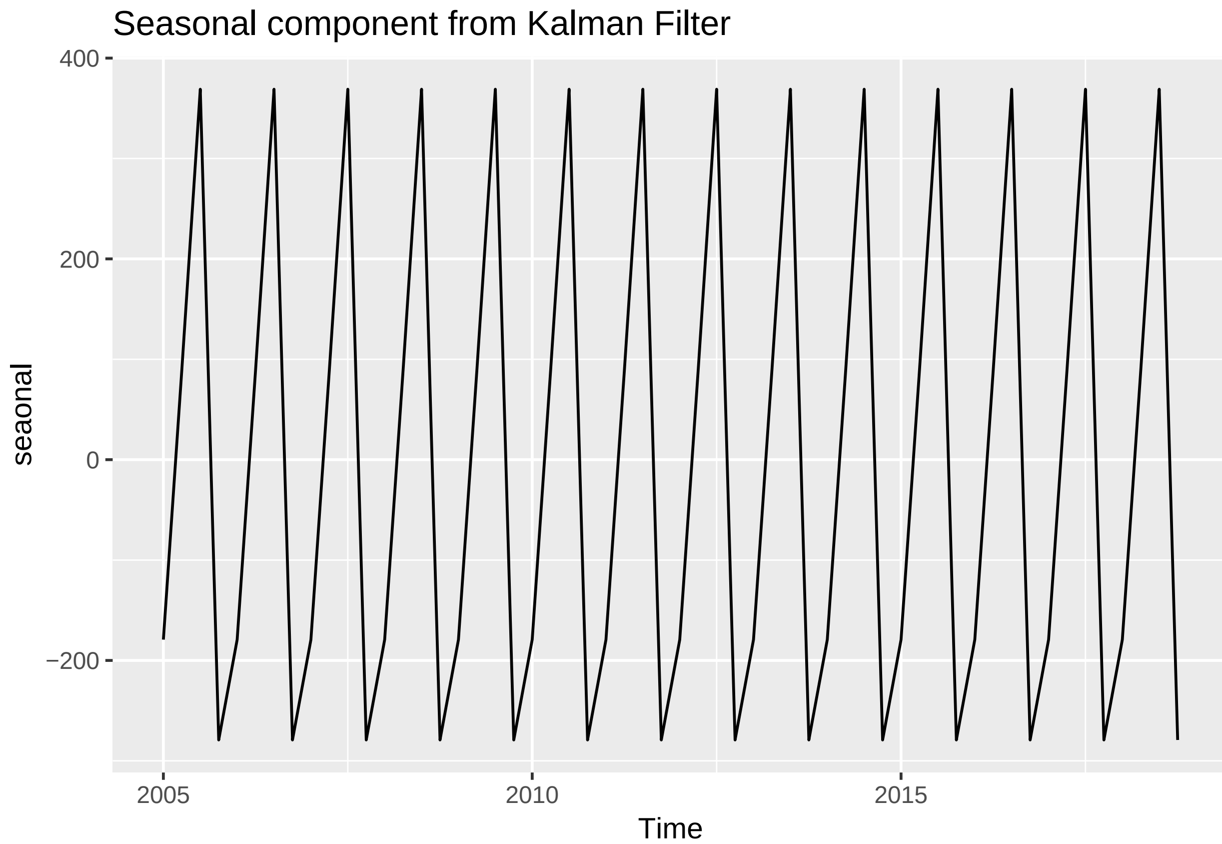

```
# Change variation of Trend from 1 to 0.1
mod_nike1 <- SSModel(data_nike ~ SSMtrend(1,Q=list(.1)) +
                       SSMseasonal(4,sea.type = "trigonometric"), H = 1)
out_nike1 <- KFS(mod_nike1, filtering = "state")

autoplot(ts(rowSums(coef(out_nike1, states = "seasonal")[,c(1,3)]),
            frequency = 4, start = c(2005,1))) +
  labs(title = 'Seasonal component from Kalman Filter(var_T=0.1)', y = 'seaonal')
```



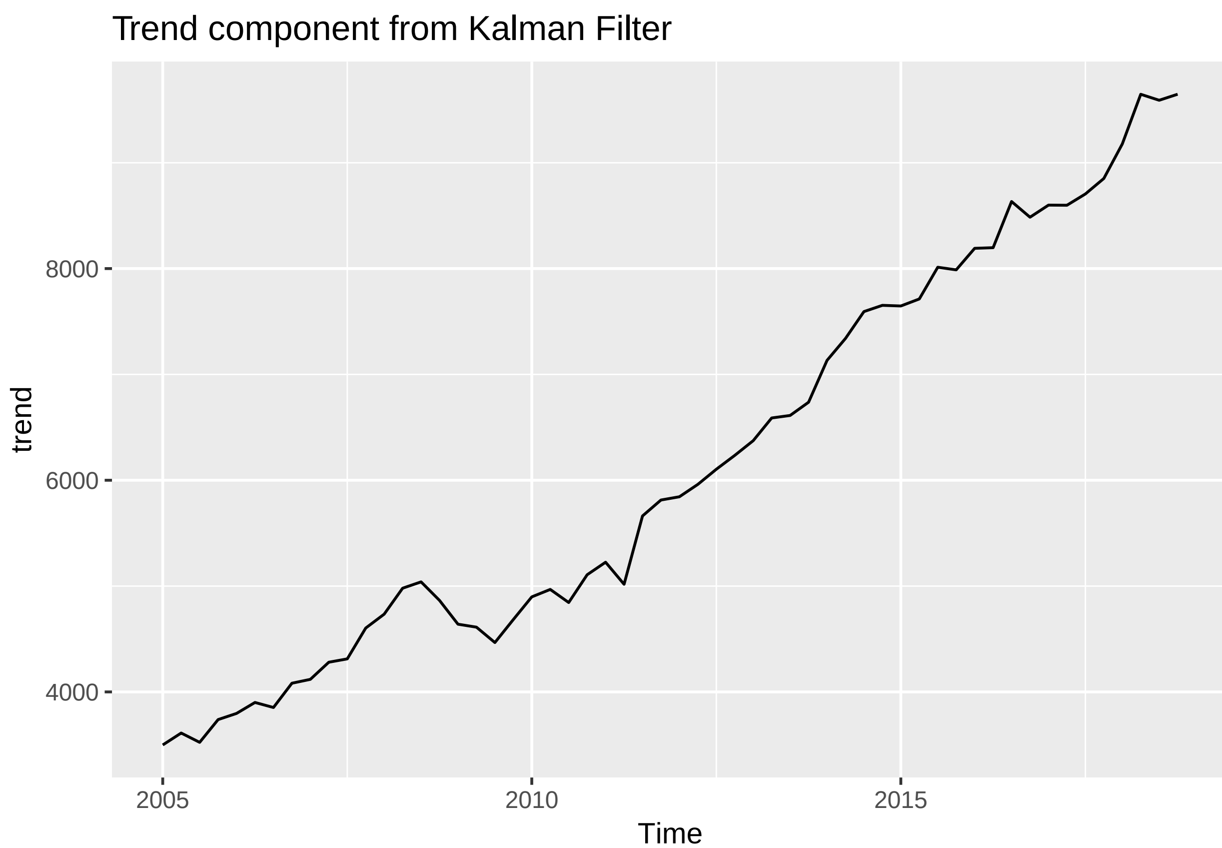

```r
autoplot(coef(out_nike1, states = "trend")) +
  labs(title = 'Trend component from Kalman Filter', y = 'trend')
```




```
# Change the variance of noise in obs eq'n from 1 to 0.1
mod_nike2 <- SSModel(data_nike ~ SSMtrend(1,Q=list(1)) +
                      SSMseasonal(4,sea.type = "trigonometric"), H = .1)
out_nike2 <- KFS(mod_nike2, filtering = "state")

autoplot(ts(rowSums(coef(out_nike2, states = "seasonal")[,c(1,3)]),
            frequency = 4, start = c(2005,1))) +
  labs(title = 'Seasonal component from Kalman Filter', y = 'seaonal')
```




```r
autoplot(coef(out_nike2, states = "trend")) +
  labs(title = 'Trend component from Kalman Filter', y = 'trend')
```

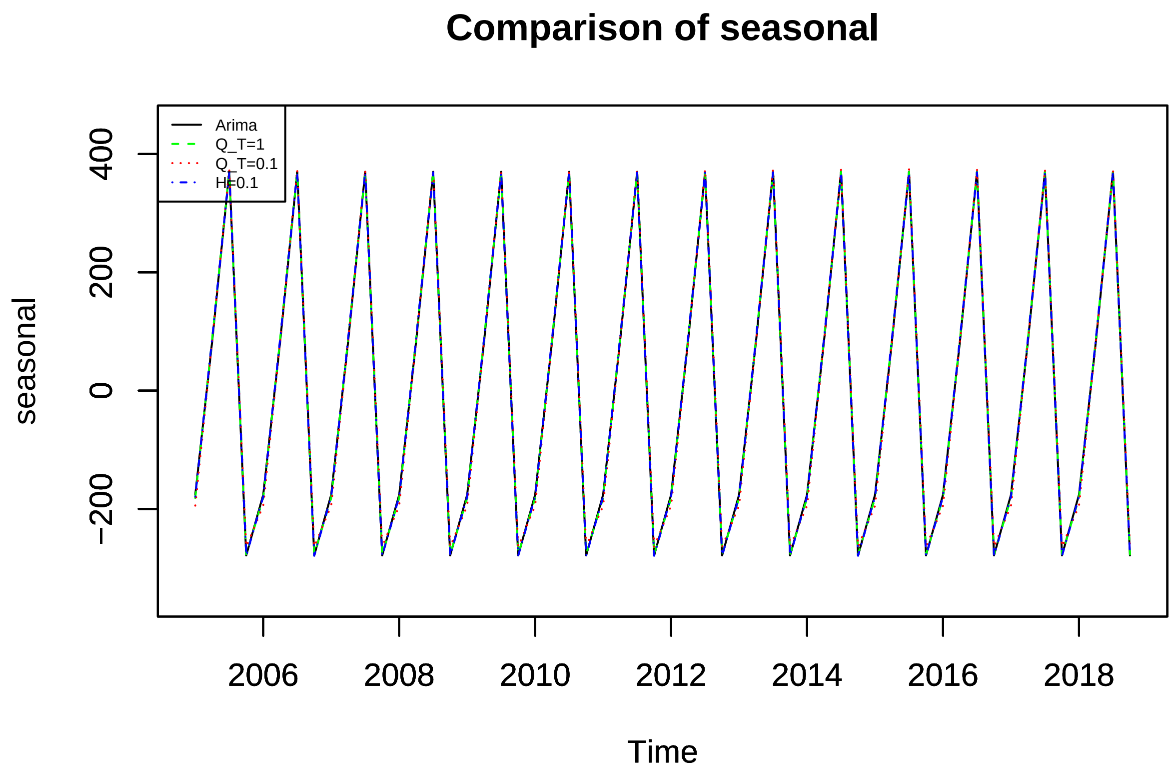



We can tell from the results above:

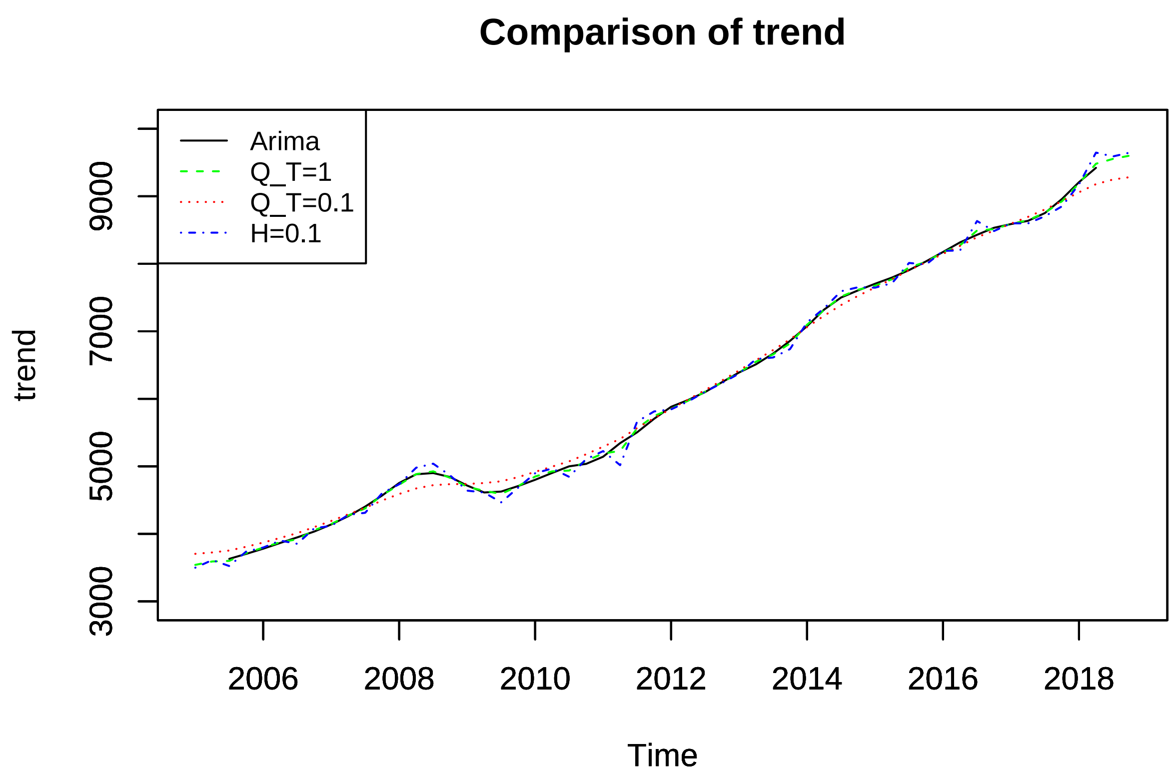
- we don't name Q in SSMseasonal -> seasonal is invariant
- decrease variance of noise in obs equation(keep other params' values, the same below) -> Trend will be spickier
- decrease variance of noise of trend -> Trend will be smoother

**Comparison**

```
# Seasonal
plot(decompose_nike$seasonal,ylab='',ylim=c(-350,450))
par(new=TRUE)
plot(ts(rowSums(coef(out_nike, states = "seasonal")[,c(1,3)]),
            frequency = 4, start = c(2005,1)),
     lty=2, col="green", ylab='',ylim=c(-350,450))
par(new=TRUE)
plot(ts(rowSums(coef(out_nike1, states = "seasonal")[,c(1,3)]),
            frequency = 4, start = c(2005,1)),
     lty=3, col="red", ylab='',ylim=c(-350,450))
par(new=TRUE)
plot(ts(rowSums(coef(out_nike2, states = "seasonal")[,c(1,3)]),
            frequency = 4, start = c(2005,1)),
     lty=4, col="blue", ylab='',ylim=c(-350,450))
title(main = 'Comparison of seasonal', ylab = 'seasonal')
legend('topleft', c('Arima','Q_T=1','Q_T=0.1','H=0.1'),lty=c(1,2,3,4),col=c('black','green','red','blue
```




```r
# Trend
plot(decompose_nike$trend, ylab='', ylim=c(3000,10000))
par(new=TRUE)
plot(coef(out_nike, states = 'trend'),
          lty=2, col="green", ylab='',ylim=c(3000,10000))
par(new=TRUE)
plot(coef(out_nike1, states = 'trend'),
          lty=3, col="red", ylab='',ylim=c(3000,10000))
par(new=TRUE)
plot(coef(out_nike2, states = 'trend'),
          lty=4, col="blue", ylab='',ylim=c(3000,10000))
title(main = 'Comparison of trend', ylab = 'trend')
legend('topleft', c('Arima','Q_T=1','Q_T=0.1','H=0.1'),lty=c(1,2,3,4),col=c('black','green','red','blue
```



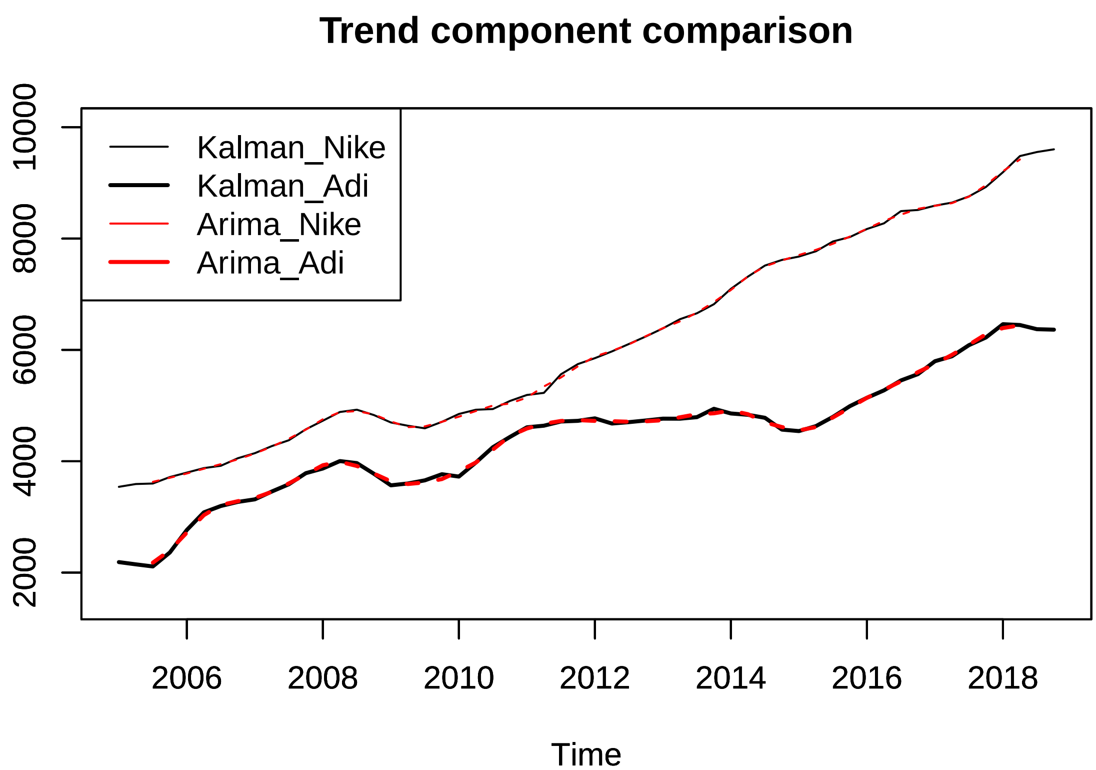

**ADIDAS(SKIP)**

## Partial Pooling

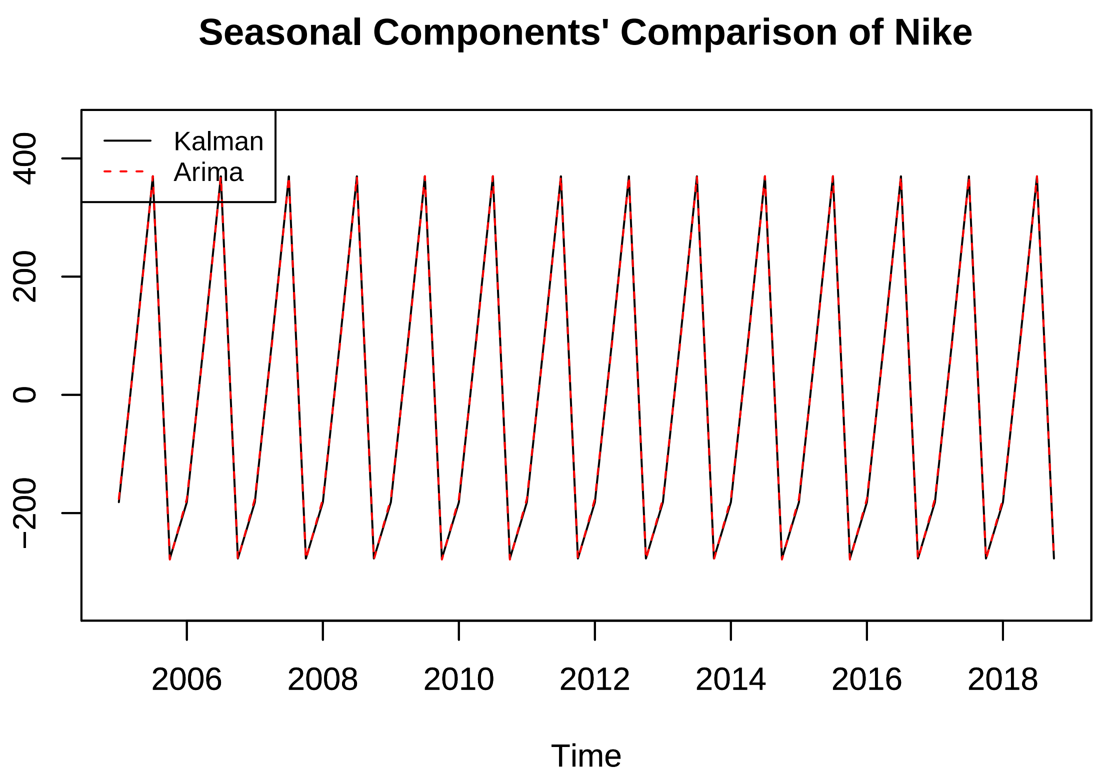
```
mod_combination <- SSModel(cbind(data_nike,data_adi) ~
                             SSMtrend(1,Q=list(diag(2))) +
                             SSMseasonal(4, sea.type = "trigonometric", index=1)+
                             SSMseasonal(4, sea.type = "trigonometric", index=2),
                           H = diag(2))

out_combination <- KFS(mod_combination, filtering = "state", smoothing = "state")
print(out_combination)
```

```
## Smoothed values of states and standard errors at time n = 56:
##                      Estimate   Std. Error
## level.data_nike      9602.5839     0.8007
## level.data_adi       6363.6043     0.8007
## sea_trig1.data_nike  -182.7943     0.2330
## sea_trig*1.data_nike -275.6700     0.2330
## sea_trig2.data_nike   -94.1114     0.1500
## sea_trig1.data_adi    -81.9219     0.2330
## sea_trig*1.data_adi  -210.1138     0.2330
## sea_trig2.data_adi   -314.8360     0.1500
```

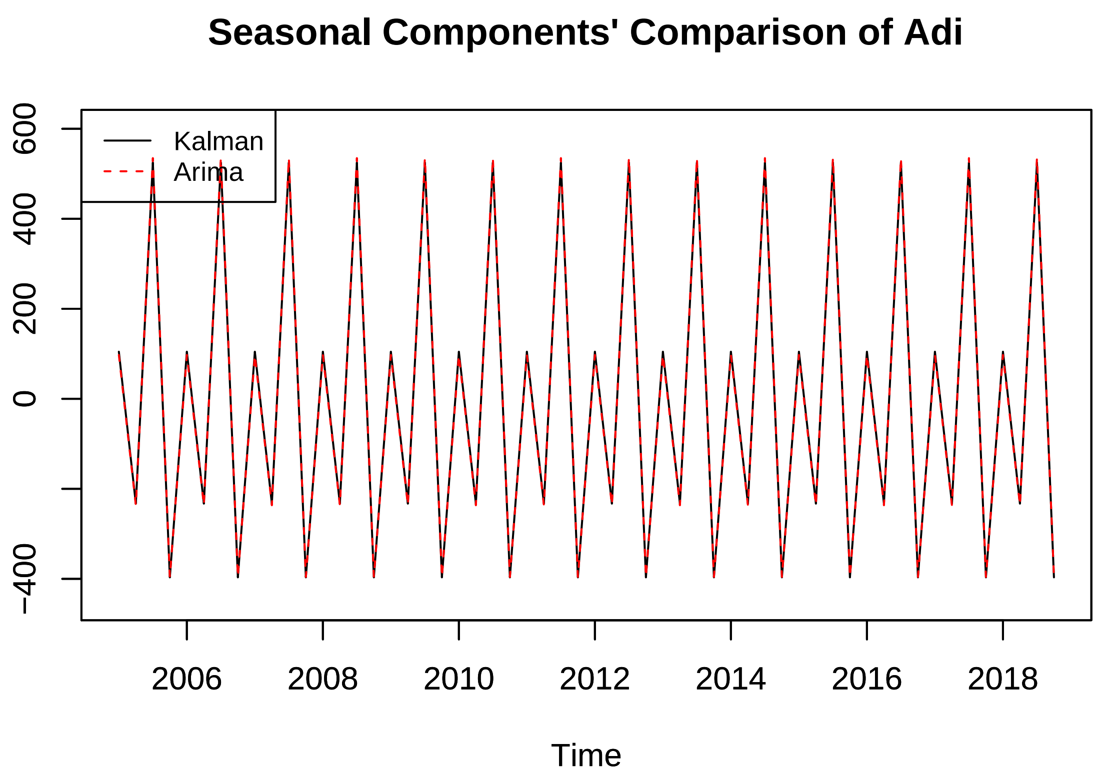
```r
# trend
ts.plot(coef(out_combination,states = "trend"),
        col=c(1,1),lwd=c(1,2), ylim=c(1500,10000), ylab='')
par(new=TRUE)
ts.plot(decompose_nike$trend, col=2, lwd=1, lty=2, ylim=c(1500,10000), ylab='')
par(new=TRUE)
ts.plot(decompose_adi$trend, col=2, lwd=2, lty=2, ylim=c(1500,10000), ylab='')
title(main="Trend component comparison")
legend('topleft', c('Kalman_Nike', 'Kalman_Adi', 'Arima_Nike', 'Arima_Adi'), col=c(1,1,2,2), lwd=c(1,2,
```



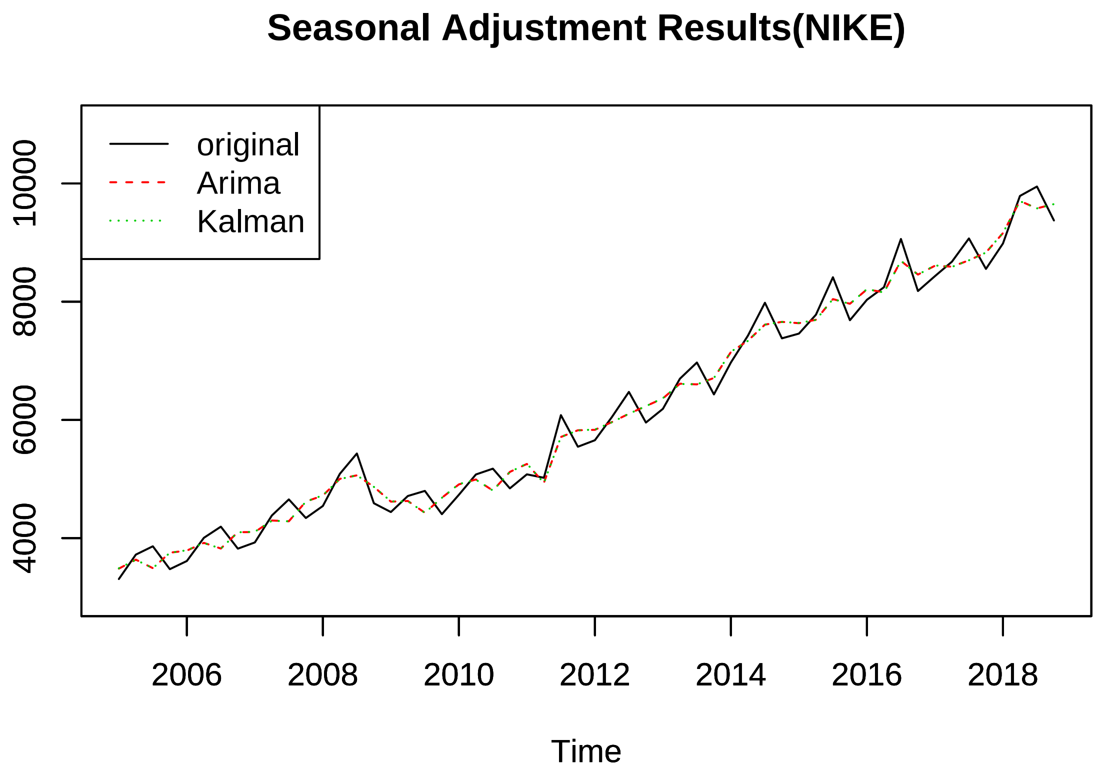

```r
# seasonal
## Nike
ts.plot(ts(rowSums(coef(out_combination,states="seasonal")[,c(1,3)]),
           frequency = 4, start = c(2005,1)),
        ylim=c(-350,450), ylab='')
par(new=TRUE)
ts.plot(decompose_nike$seasonal, col=2, lty=2, ylim=c(-350,450), ylab='')
legend("topleft", c('Kalman', 'Arima'), col=c(1,2), lty=c(1,2),cex=0.8)
title(main="Seasonal Components' Comparison of Nike")
```



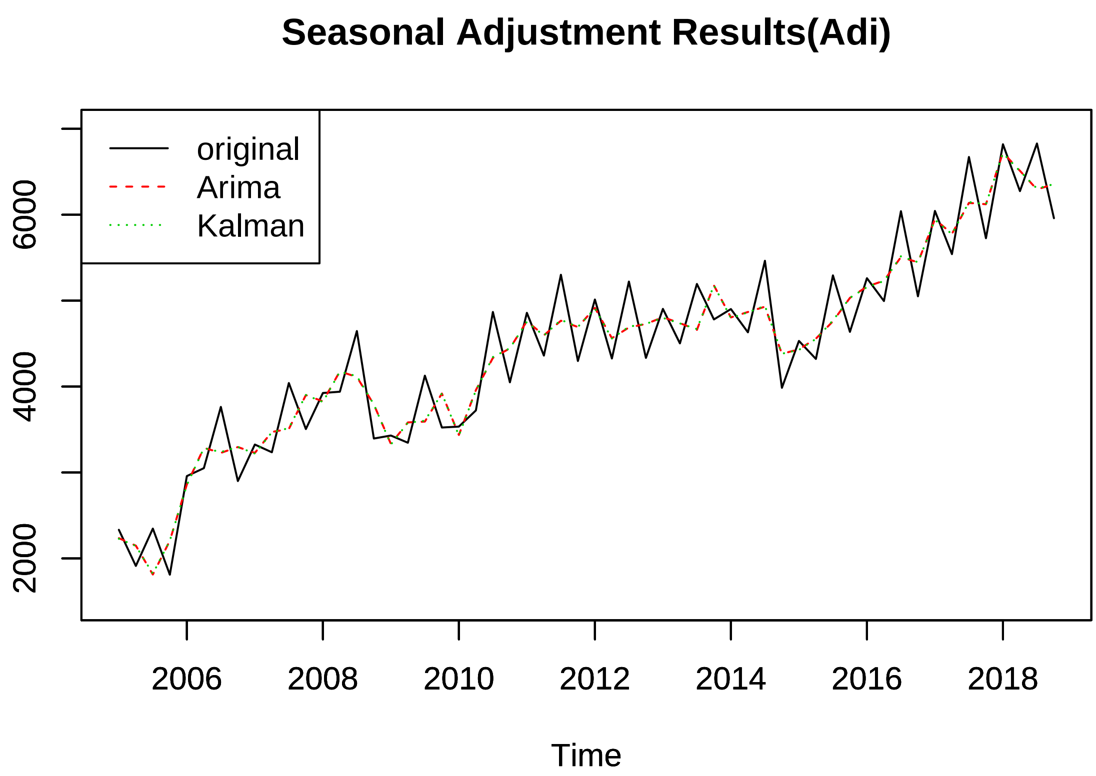

```r
## Adi
ts.plot(ts(rowSums(coef(out_combination,states="seasonal")[,c(4,6)]),
           frequency = 4, start = c(2005,1)),
        ylim=c(-450,600), ylab='')
par(new=TRUE)
ts.plot(decompose_adi$seasonal, col=2, lty=2, ylim=c(-450,600), ylab='')
legend("topleft", c('Kalman', 'Arima'), col=c(1,2), lty=c(1,2),cex=0.8)
title(main="Seasonal Components' Comparison of Adi")
```




```
# seasonal adjustment
## Nike
ts.plot(data_nike, ylab='',ylim=c(3000,11000))
par(new=TRUE)
ts.plot(data_nike-decompose_nike$seasonal,
        col=2, lty =2, ylab='',ylim=c(3000,11000))
par(new=TRUE)
ts.plot(data_nike-ts(rowSums(coef(out_combination,states = "seasonal")[,c(1,3)]),
                     frequency = 4, start = c(2005,1)),
        col=3, lty=3, ylab='',ylim=c(3000,11000))
title(main = "Seasonal Adjustment Results(NIKE)")
legend("topleft", c("original","Arima","Kalman"), lty=c(1,2,3), col=c(1,2,3))
```




```r
## Adi
ts.plot(data_adi, ylab='',ylim=c(1500,7000))
par(new=TRUE)
ts.plot(data_adi-decompose_adi$seasonal,
        col=2, lty =2, ylab='',ylim=c(1500,7000))
par(new=TRUE)
ts.plot(data_adi-ts(rowSums(coef(out_combination,states = "seasonal")[,c(4,6)]),
                    frequency = 4, start = c(2005,1)),
        col=3, lty=3, ylab='',ylim=c(1500,7000))
title(main = "Seasonal Adjustment Results(Adi)")
legend("topleft", c("original","Arima","Kalman"), lty=c(1,2,3), col=c(1,2,3))
```



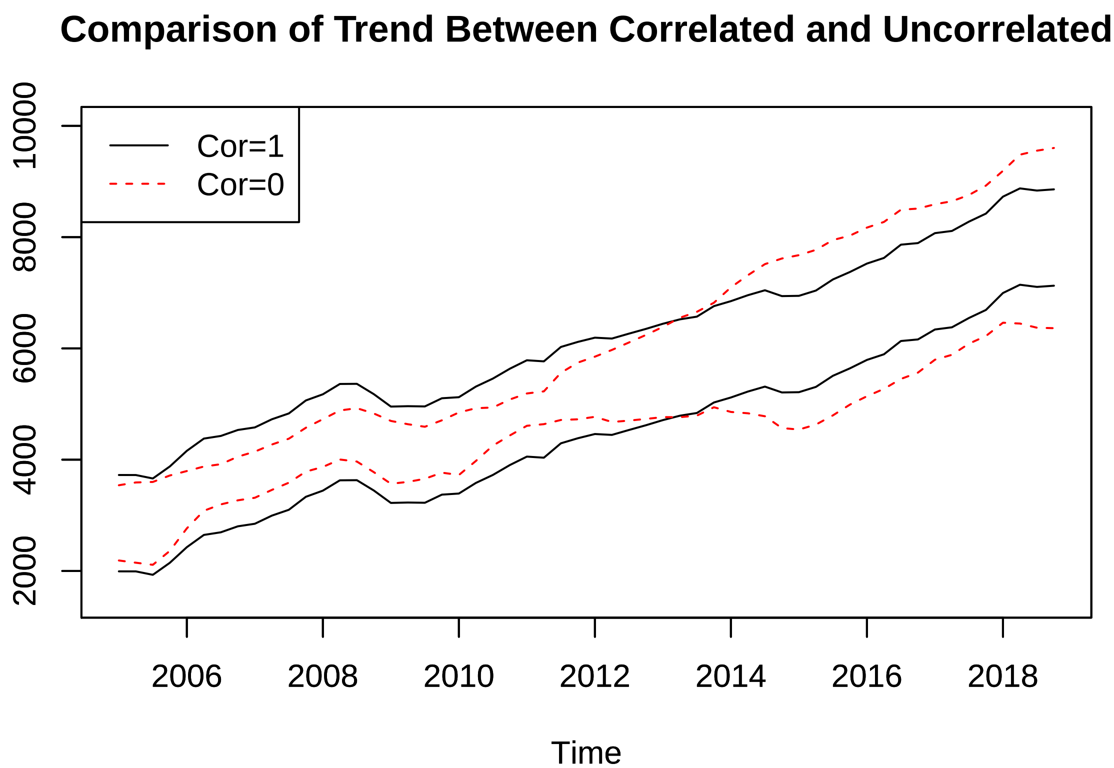

## Different Situations

### Change the correlation of trend into 1 (correlated)

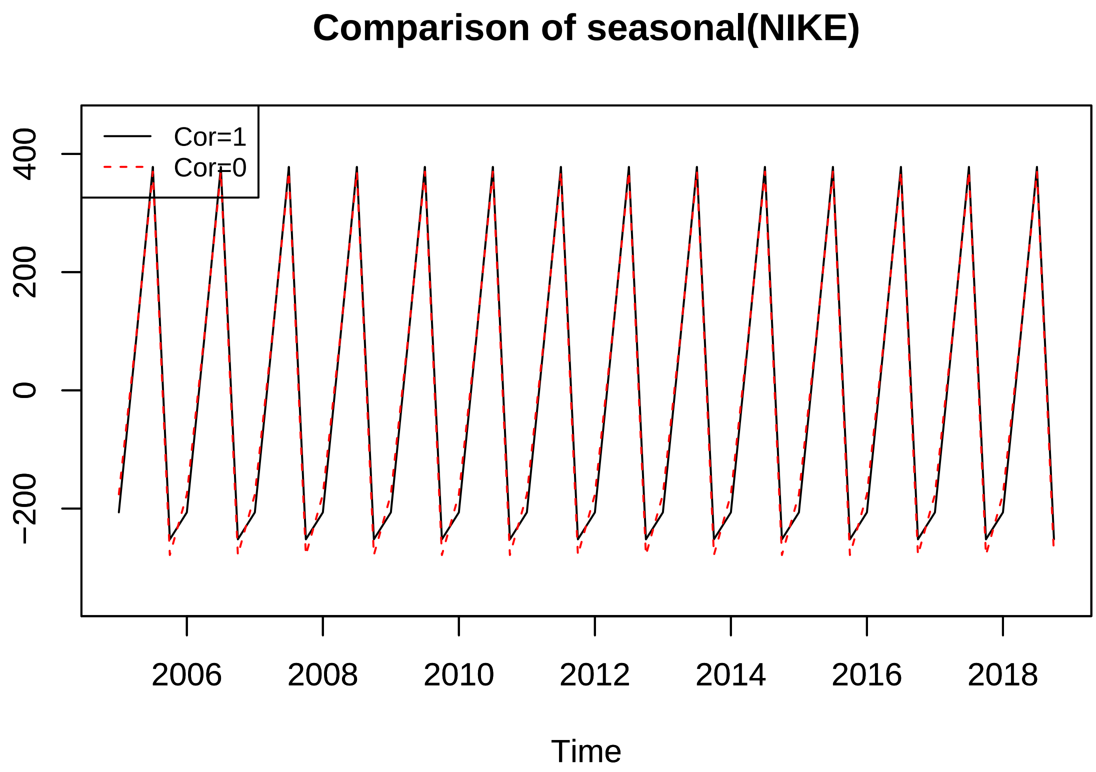
```
mod_combination1 <- SSModel(cbind(data_nike,data_adi)~
                              SSMtrend(1,Q=list(matrix(c(1,1,1,1),ncol=2)))+
                              SSMseasonal(4,sea.type = "trigonometric", index = 1)+
                              SSMseasonal(4,sea.type = "trigonometric", index = 2),
                            H = diag(2))

out_combination1 <- KFS(mod_combination1, filtering = "state", smoothing = "state")
print(out_combination1)
```

```
## Smoothed values of states and standard errors at time n = 56:
##                       Estimate   Std. Error
## level.data_nike       8858.8732     0.6309
## level.data_adi        7127.0518     0.6309
## sea_trig1.data_nike   -166.2265     0.2327
## sea_trig*1.data_nike  -292.2378     0.2327
## sea_trig2.data_nike    -85.8275     0.1499
## sea_trig1.data_adi     -99.0479     0.2327
## sea_trig*1.data_adi   -192.9878     0.2327
## sea_trig2.data_adi    -323.3989     0.1499
```

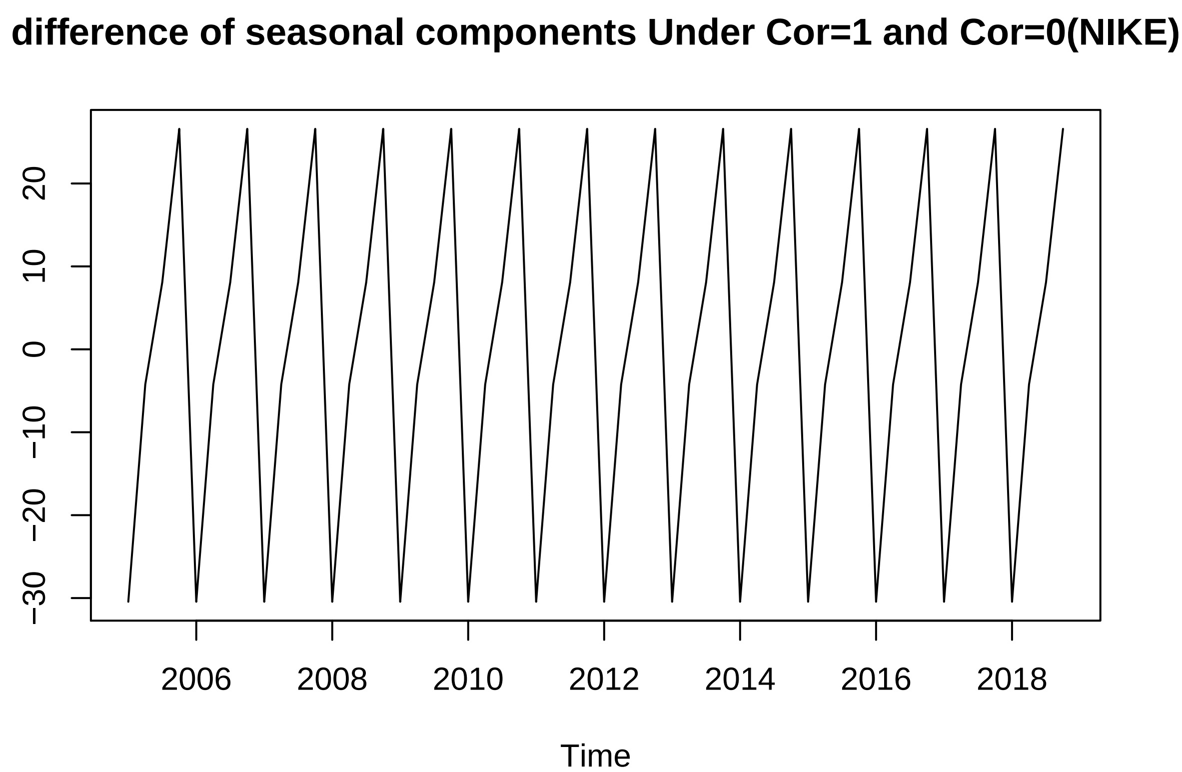
```r
# Trend
## upper line is Nike, the other one is Adidas
ts.plot(coef(out_combination1, states="trend"), ylim=c(1500,10000))
par(new=TRUE)
ts.plot(coef(out_combination, states = "trend"), col=2, lty=2, ylim=c(1500,10000))
legend('topleft', c('Cor=1','Cor=0'),lty=c(1,2),col=c(1,2))
title(main='Comparison of Trend Between Correlated and Uncorrelated')
```



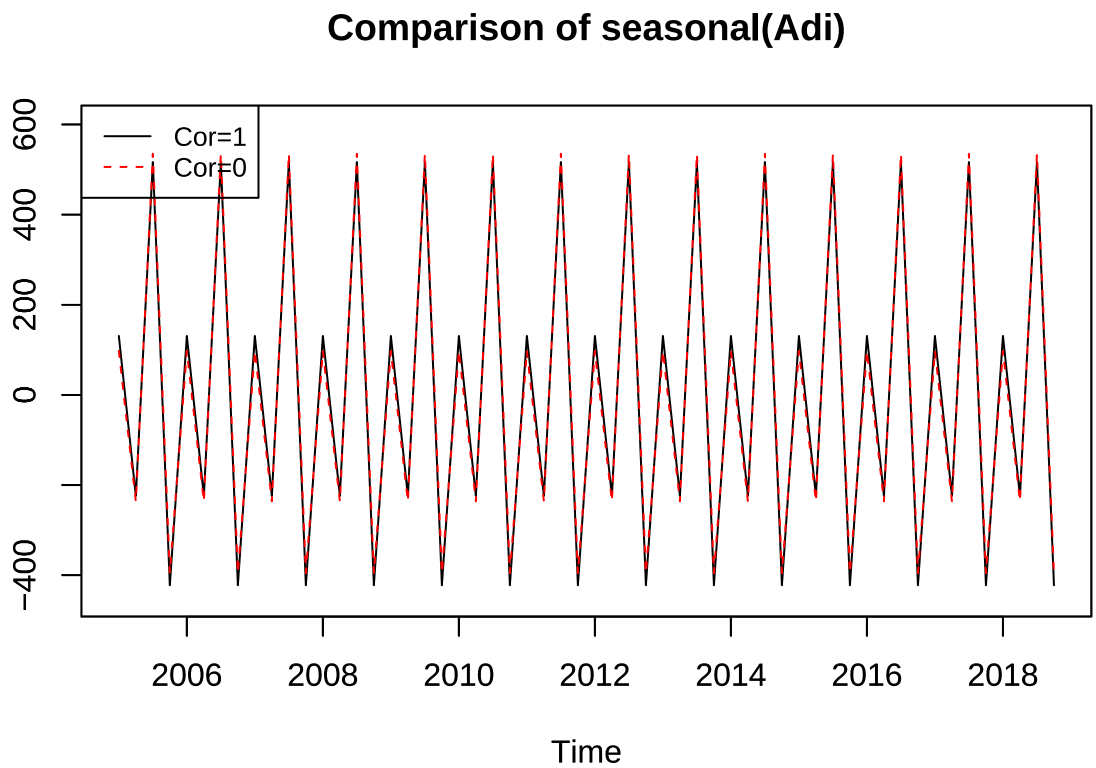

```r
# Seasonal
## Nike
plot(ts(rowSums(coef(out_combination1,states = "seasonal")[,c(1,3)]),
        frequency = 4, start=c(2005,1)),
     ylim=c(-350,450), ylab='')
par(new=TRUE)
plot(decompose_nike$seasonal, lty=2, col=2, ylim=c(-350,450),ylab='')
title(main='Comparison of seasonal(NIKE)')
legend('topleft',c('Cor=1','Cor=0'),lty=c(1,2),col=c(1,2),cex=0.8)
```



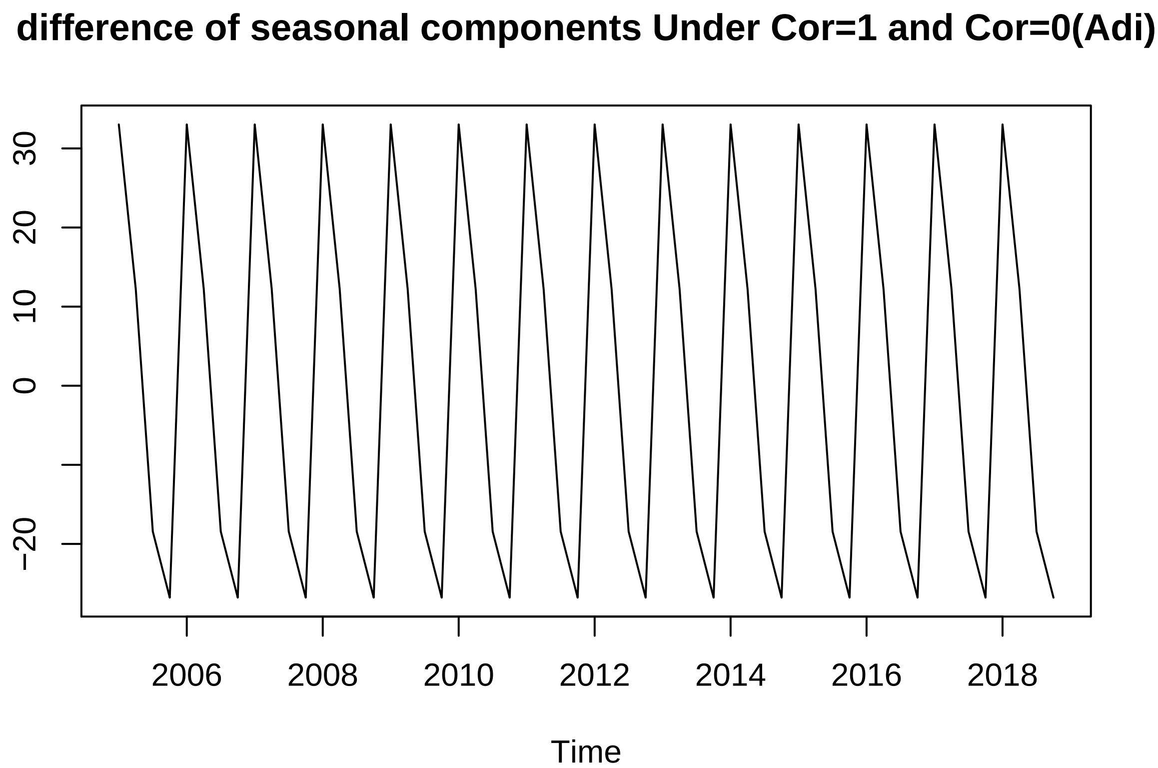

```
plot(ts(rowSums(coef(out_combination1,states = "seasonal")[,c(1,3)]),
        frequency = 4, start=c(2005,1)) - decompose_nike$seasonal,ylab='',
      main='difference of seasonal components Under Cor=1 and Cor=0(NIKE)')
```




```
## Adi
plot(ts(rowSums(coef(out_combination1,states = "seasonal")[,c(4,6)]),
        frequency = 4, start=c(2005,1)),
     ylim=c(-450,600), ylab='')
par(new=TRUE)
plot(decompose_adi$seasonal, lty=2, col=2, ylim=c(-450,600),ylab='')
title(main='Comparison of seasonal(Adi)')
legend('topleft',c('Cor=1','Cor=0'),lty=c(1,2),col=c(1,2),cex=0.8)
```




```r
plot(ts(rowSums(coef(out_combination1,states = "seasonal")[,c(4,6)]),
        frequency = 4, start=c(2005,1)) - decompose_adi$seasonal,ylab='',
      main='difference of seasonal components Under Cor=1 and Cor=0(Adi)')
```



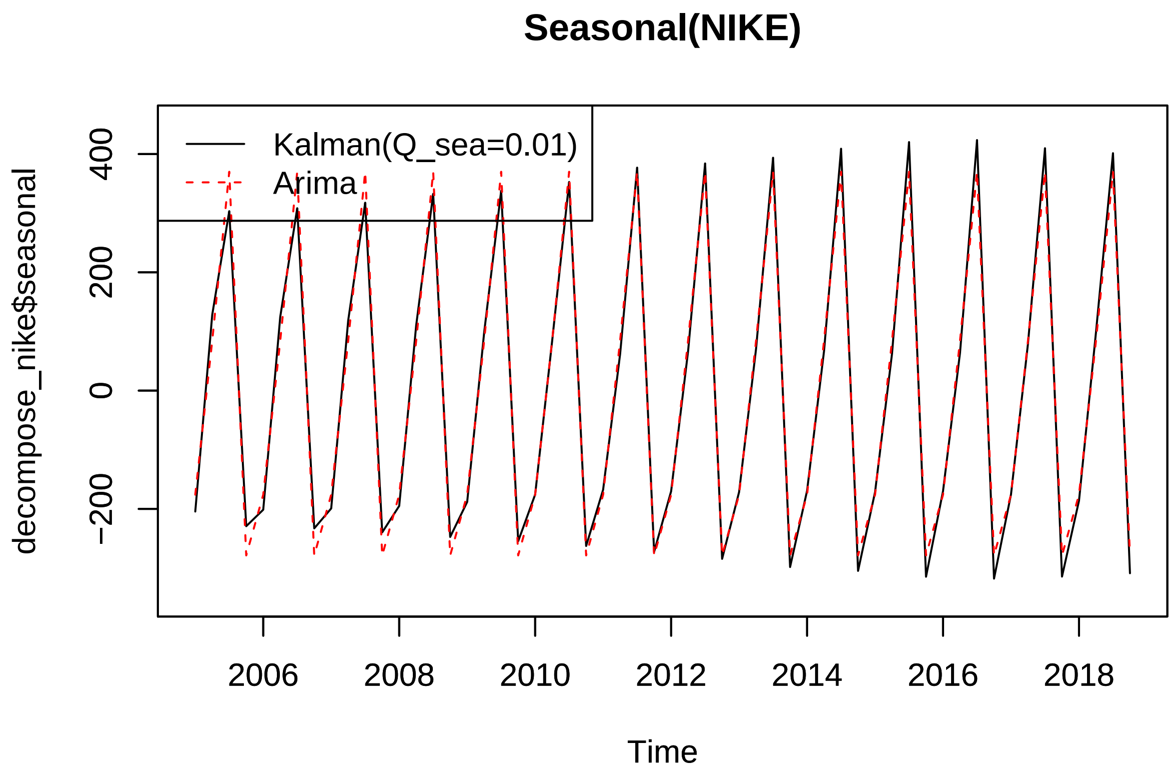

### Change the covariance matrix(variance) of seasonal

**Note:** The models we used before are all based on invariant seasonal component, now we move on to Gaussian seasonal.

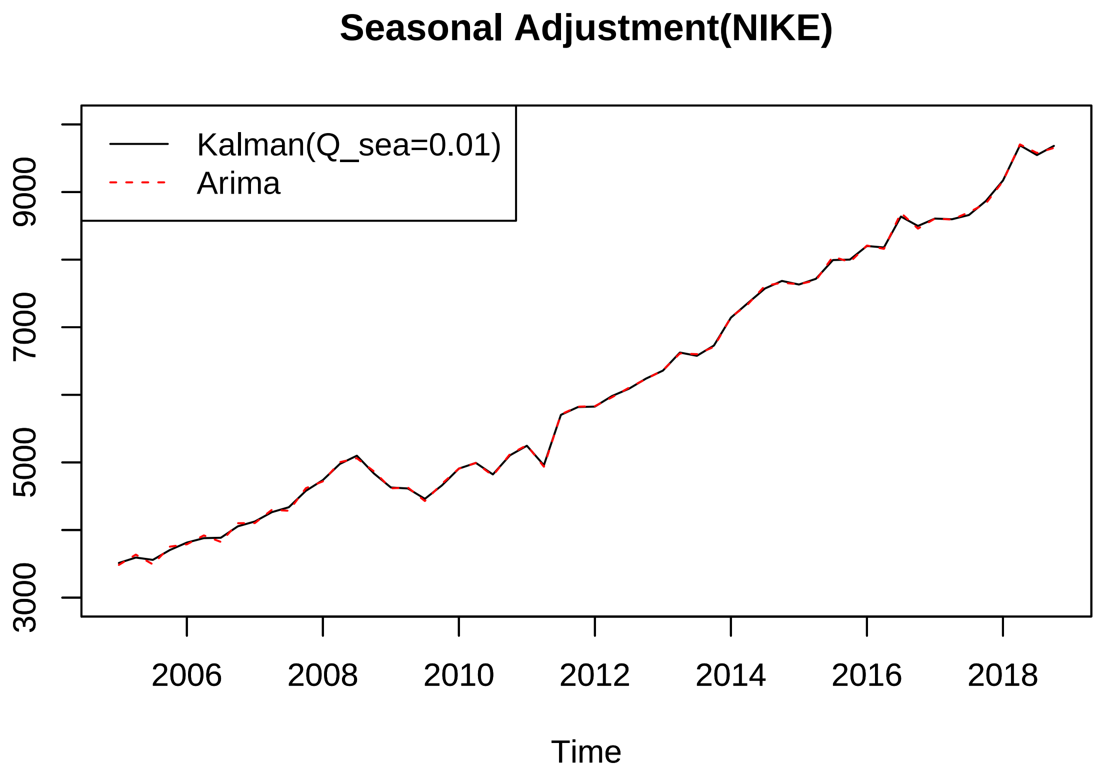
#### Working on single case first(NIKE)

#### Change the variance of seasonal from 0 to 0.01 (Q=H=1)

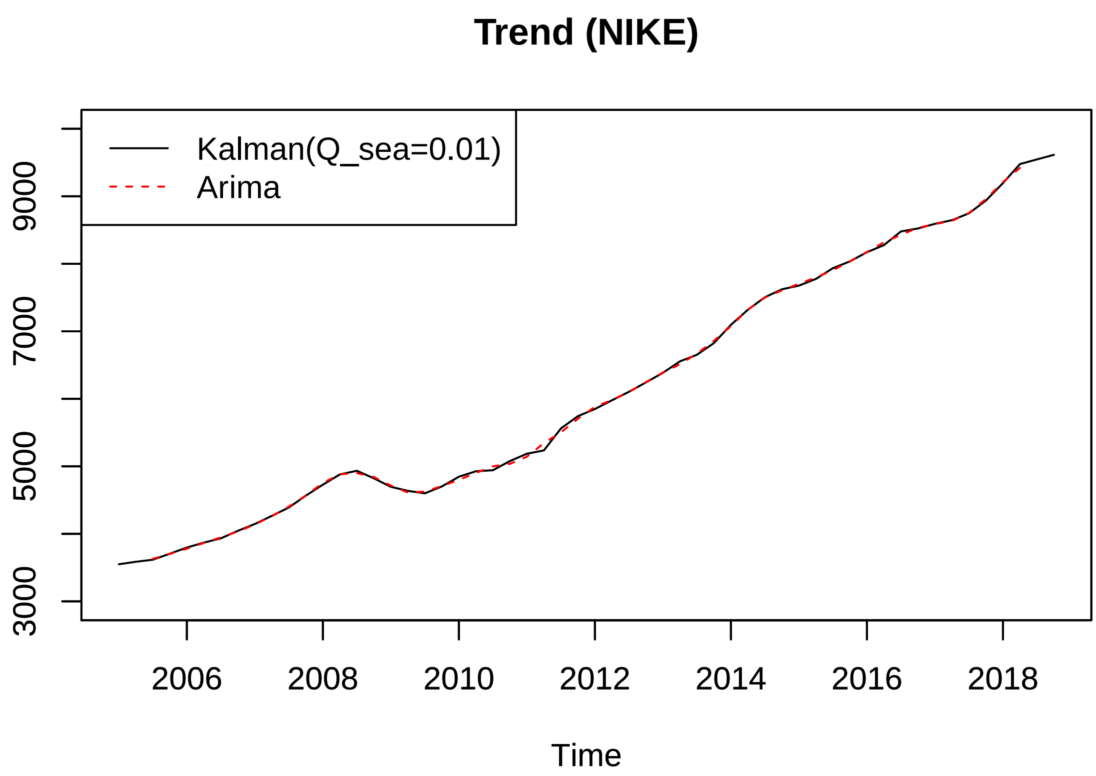
```
mod_nike_seasonal0.01 <- SSModel(data_nike ~ SSMtrend(degree=1,Q=list(1))+
                              SSMseasonal(4,Q=0.01,sea.type = "trigonometric"),
                     H = 1)
out_nike_seasonal0.01 <- KFS(mod_nike_seasonal0.01,
                            filtering="state",smoothing='state')

# Seasonal
plot(ts(rowSums(coef(out_nike_seasonal0.01,states = "seasonal")[,c(1,3)]),
        frequency = 4, start = c(2005,1)),
     ylab='',ylim=c(-350,450))
par(new=TRUE)
plot(decompose_nike$seasonal,ylim=c(-350,450),col=2,lty=2)
legend("topleft", c('Kalman(Q_sea=0.01)','Arima'), col=c(1,2), lty=c(1,2))
title(main='Seasonal(NIKE)')
```



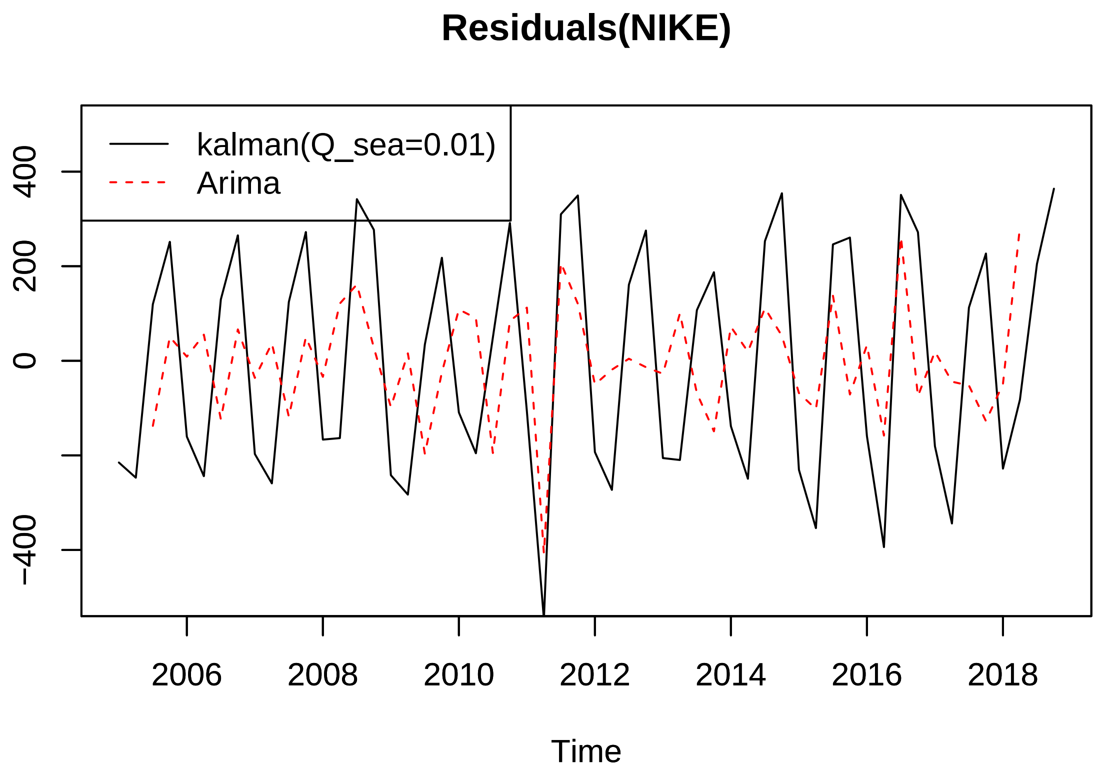

```r
# Seasonal Adjustment
plot(data_nike-ts(rowSums(coef(out_nike_seasonal0.01,
                              states = "seasonal")[,c(1,3)]),
        frequency = 4, start = c(2005,1)),
     ylab='', ylim=c(3000,10000))
par(new=TRUE)
plot(data_nike-decompose_nike$seasonal,col='red',lty=2,ylab='',ylim=c(3000,10000))
legend("topleft", c('Kalman(Q_sea=0.01)','Arima'),lty=c(1,2),col=c(1,2))
title(main='Seasonal Adjustment(NIKE)')
```




```
# trend
plot(coef(out_nike_seasonal0.01,states = "trend"), ylab='',ylim=c(3000,10000))
par(new=TRUE)
plot(decompose_nike$trend,ylab='',col=2,lty=2,ylim=c(3000,10000))
legend("topleft", c('Kalman(Q_sea=0.01)','Arima'),lty=c(1,2),col=c(1,2))
title(main='Trend (NIKE)')
```

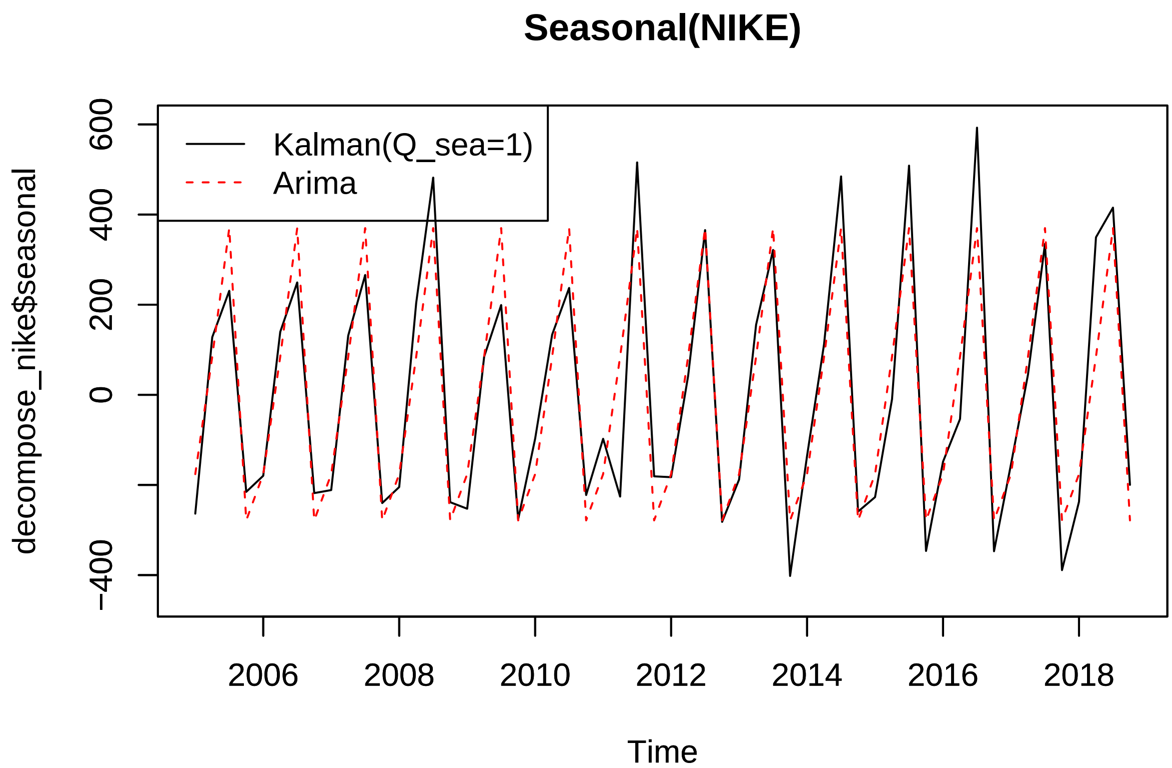



```r
# residuals
plot(data_nike-rowSums(coef(out_nike_seasonal0.01)),ylim=c(-500,500),ylab='')
par(new=TRUE)
plot(decompose_nike$random, ylim=c(-500,500), ylab='',lty=2,col=2)
legend("topleft", c('kalman(Q_sea=0.01)','Arima'),lty=c(1,2),col=c(1,2))
title(main="Residuals(NIKE)")
```

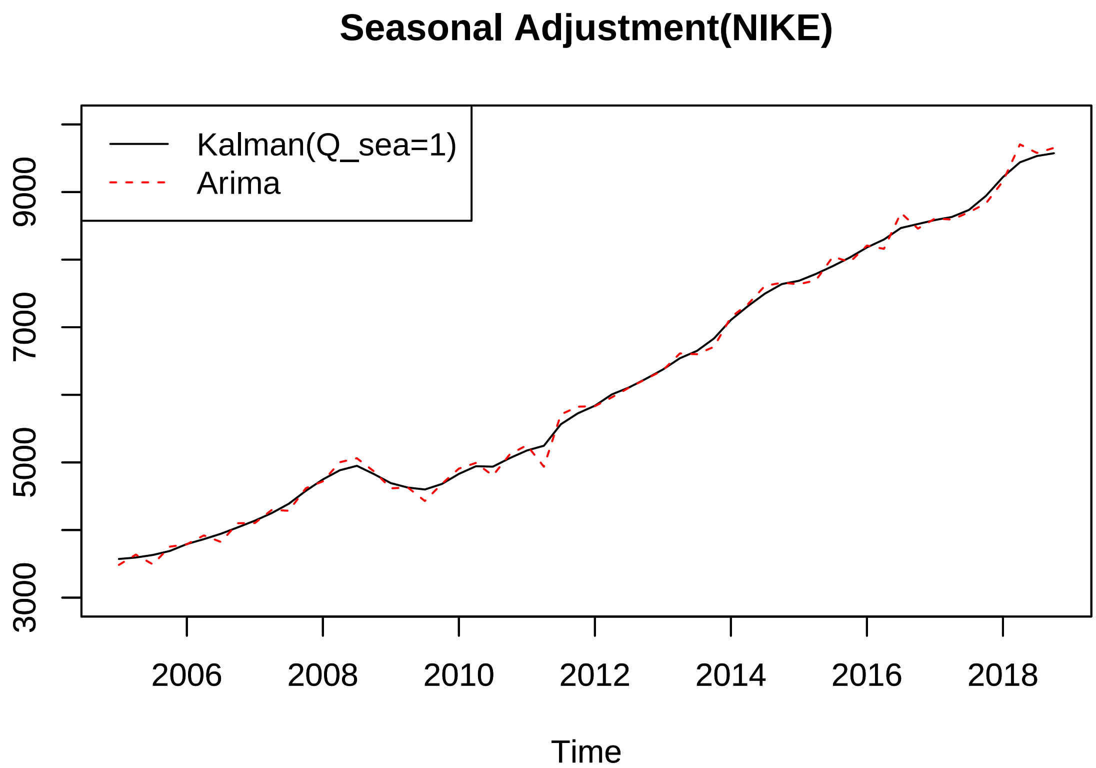



**variance of seasonal is equal to 1**

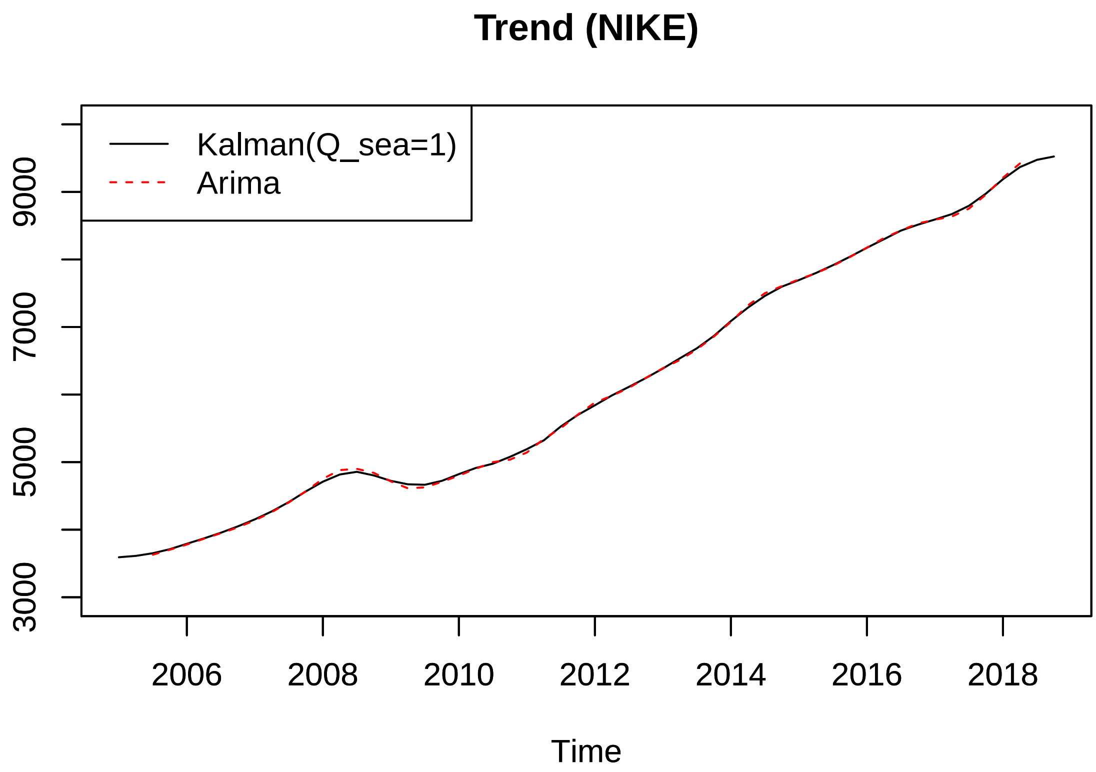
```
mod_nike_seasonal1 <- SSModel(data_nike ~ SSMtrend(degree=1,Q=list(1))+
                              SSMseasonal(4,Q=1,sea.type = "trigonometric"),
                        H = 1)
out_nike_seasonal1 <- KFS(mod_nike_seasonal1,
                          filtering="state",smoothing='state')

# Seasonal
plot(ts(rowSums(coef(out_nike_seasonal1,states = "seasonal")[,c(1,3)]),
        frequency = 4, start = c(2005,1)),
     ylab='',ylim=c(-450,600))
par(new=TRUE)
plot(decompose_nike$seasonal,ylim=c(-450,600),col=2,lty=2)
legend("topleft", c('Kalman(Q_sea=1)','Arima'), col=c(1,2), lty=c(1,2))
title(main='Seasonal(NIKE)')
```



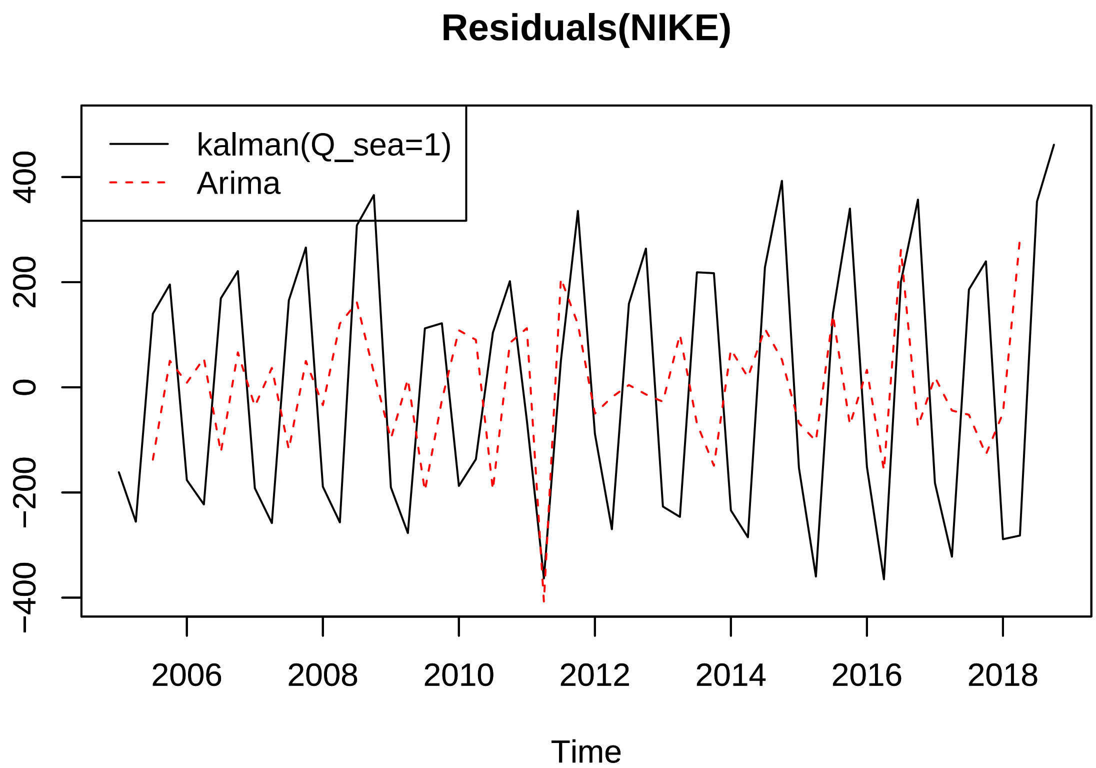

```r
# Seasonal Adjustment
# which is StatCan's result
plot(data_nike-ts(rowSums(coef(out_nike_seasonal1,
                               states = "seasonal")[,c(1,3)]),
        frequency = 4, start = c(2005,1)),
     ylab='', ylim=c(3000,10000))
par(new=TRUE)
plot(data_nike-decompose_nike$seasonal,col='red',lty=2,ylab='',ylim=c(3000,10000))
legend("topleft", c('Kalman(Q_sea=1)','Arima'),lty=c(1,2),col=c(1,2))
title(main='Seasonal Adjustment(NIKE)')
```




```r
# trend
plot(coef(out_nike_seasonal1,states = "trend"), ylab='',ylim=c(3000,10000))
par(new=TRUE)
plot(decompose_nike$trend,ylab='',col=2,lty=2,ylim=c(3000,10000))
legend("topleft", c('Kalman(Q_sea=1)','Arima'),lty=c(1,2),col=c(1,2))
title(main='Trend (NIKE)')
```




```
# residuals
plot(data_nike-rowSums(coef(out_nike_seasonal1)),ylim=c(-400,500),ylab='')
par(new=TRUE)
plot(decompose_nike$random, ylim=c(-400,500), ylab='',lty=2,col=2)
legend("topleft", c('kalman(Q_sea=1)','Arima'),lty=c(1,2),col=c(1,2))
title(main="Residuals(NIKE)")
```


Residuals(NIKE)
kalman(Q_sea=1)
Arima
400
200
0
−200
−400
2006
2008
2010
2012
2014
2016
2018
Time