Lab1_Decomposition_a24kimwu

2025-09-06

Basic R computations

```r
a <- 5
b <- 2
```

```r
a + b # addition
```

```
## [1] 7
```

```r
a - b # subtraction
```

```
## [1] 3
```

```r
a * b # multiplication
```

```
## [1] 10
```

```r
a/b # division
```

```
## [1] 2.5
```

```r
total <- a + b
print(total)
```

```
## [1] 7
```

```r
total == sum(a,b)
```

```
## [1] TRUE
```

```r
total == a-b
```

```
## [1] FALSE
```

```r
c(5,8)
```

```
## [1] 5 8
```

```r
all <- c(a, b, total)
print(all)
```

```
## [1] 5 2 7
```

Loading packages into R

```r
# forecast package
library(forecast)
```

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

```r
# tsutils package
library(tsutils)
```

Loading data into R

```r
Y <- read.csv('./workshop1R.csv')
print(Y)
```

```
##     Level_A  Level_B LevelShift  Trend_A  Trend_B  Season_A Season_B
## 1  309.5927 484.7822   687.9850 4811.254 1911.870  231.1930 229.3392
## 2  285.0966 494.9082   687.2746 4785.975 1950.702  467.6818 308.8277
## 3  298.8200 478.1126   682.8712 4746.325 2074.533  649.9551 271.2460
## 4  284.3028 479.0342   718.7764 4791.332 2027.001  465.7579 293.9912
## 5  308.5171 489.2530   733.6778 4748.645 2168.685  726.8946 491.7330
## 6  306.8993 502.9581   662.9156 4733.944 2134.204  926.6489 754.2631
## 7  325.4628 475.7626   702.4218 4860.476 2143.586 1043.4760 732.8038
## 8  326.9226 481.5800   730.0287 4837.283 2203.608  890.6451 482.7710
## 9  302.5102 518.7196   738.7401 4805.012 2269.670  936.9840 548.9135
## 10 319.5777 468.4896   749.2881 4792.921 2288.267  972.0107 676.5493
## 11 332.5051 467.0762   736.9927 4772.660 2318.459  392.7123 349.2937
## 12 342.4510 487.6958   718.5036 4719.646 2354.575  773.3365 260.8061
## 13 339.0753 482.6020   705.9423 4774.294 2348.642  231.3019 194.3558
## 14 334.1232 458.8387   745.0246 4725.823 2410.823  482.3914 268.9270
## 15 361.4546 505.8385   733.2173 4674.412 2425.872  686.6944 242.6627
## 16 336.1173 457.3302   739.2857 4758.510 2443.260  508.5456 243.3807
## 17 319.6460 480.6295   746.1170 4800.288 2418.268  783.0867 416.5342
## 18 317.3951 463.1917   719.2866 4734.328 2441.637  948.8349 637.2254
## 19 328.0461 515.9150   718.5111 4626.869 2501.871 1033.9909 643.0106
## 20 304.5760 487.7321   766.2992 4611.724 2490.253  885.7968 449.8246
## 21 355.3091 501.9705   699.9061 4660.885 2547.162  884.2049 505.8062
## 22 330.1500 475.2583   743.0381 4696.642 2597.297  950.9984 645.9131
## 23 342.8376 526.4980   755.6695 4606.577 2579.102  414.5146 339.8396
## 24 327.5952 442.9300   770.0157 4644.255 2632.576  831.5753 227.8289
## 25 328.0343 499.9444   777.8046 4704.586 2651.229  249.5404 219.6523
## 26 322.7103 510.9078   758.6962 4574.058 2654.030  490.2678 330.9740
## 27 329.1623 518.4830   754.8626 4602.505 2664.998  702.6915 273.7864
## 28 312.8083 470.6329   734.4742 4638.239 2659.445  487.3626 262.4567
## 29 319.9644 484.1704   698.7198 4440.421 2752.220  751.0515 437.6518
## 30 310.1535 471.2600   729.9694 4525.592 2759.949  933.2986 666.4505
## 31 338.1874 489.2770   817.1418 4506.761 2689.676 1047.3553 638.4910
## 32 314.8126 490.8059   871.6709 4500.839 2819.194  933.6356 470.9215
## 33 345.3493 493.5122   864.1622 4407.498 2811.123  917.5312 565.4009
## 34 313.4708 491.7161   879.3891 4415.625 2817.265  978.5181 676.0848
## 35 304.8354 505.7463   839.3719 4392.075 2903.696  375.8654 360.7307
## 36 311.6021 528.2059   861.2266 4330.295 2847.171  743.5117 282.9457
## 37 307.4285 509.4413   879.8566 4331.313 2960.159  230.7031 178.8305
## 38 347.2009 503.8770   882.4076 4244.963 2914.750  507.5720 260.3417
## 39 326.6501 486.3193   843.9639 4245.427 2927.487  737.4308 277.9441
## 40 308.4443 483.7063   859.9885 4117.801 2980.284  534.9390 288.4126
## 41 299.1128 504.0208   850.4899 4282.850 2940.516  896.3776 535.3208
## 42 341.9722 500.8155   839.6968 4338.188 2956.346 1060.8528 818.7349
## 43 302.1470 538.5808   864.6994 4261.027 2963.466 1164.6262 771.2875
## 44 295.1502 486.6391   837.5950 4274.002 2938.626 1044.2644 584.1875
```

```
## 45 351.9557 517.4006   865.0451 4142.335 3045.697 1051.6022 625.2942
## 46 332.8738 503.0216   897.2701 4115.690 3047.369 1154.3498 795.1606
## 47 324.6155 451.8103   826.6039 4146.546 3106.114  468.5850 397.7713
## 48 306.7664 502.4365   890.3355 4059.242 3071.805  876.5367 318.8312
## 49 320.3624 504.7120   855.6515 4110.735 3021.431  263.3852 231.9475
## 50 350.8271 495.3900   875.2794 4026.853 3070.378  515.9486 317.1494
## 51 317.0764 493.6047   875.9614 4005.306 3129.974  734.3243 328.6238
## 52 300.9870 505.4475   888.6250 3959.988 3118.365  484.5399 300.6523
## 53 336.8473 499.1231   815.7227 3941.407 3168.412  765.2680 525.8733
## 54 316.1264 504.4805   901.7243 3809.953 3153.210  920.0402 807.4535
## 55 321.9198 483.0237   852.6084 3833.826 3128.485 1037.8768 743.6858
## 56 336.2565 511.3350   875.0078 3724.205 3174.239  911.4822 557.7259
## 57 335.7531 502.4504   877.0743 3664.212 3215.049  948.8460 626.3542
## 58 332.0578 511.9520   838.1259 3630.718 3149.663 1026.4180 809.1212
## 59 324.6613 498.7966   840.1391 3727.921 3250.898  385.7735 386.1633
## 60 352.9658 531.7996   869.4072 3539.250 3212.137  795.1730 303.0199
##    TrendSeason
## 1          284
## 2          277
## 3          317
## 4          313
## 5          318
## 6          374
## 7          413
## 8          405
## 9          355
## 10         306
## 11         271
## 12         306
## 13         315
## 14         301
## 15         356
## 16         348
## 17         355
## 18         422
## 19         465
## 20         467
## 21         404
## 22         347
## 23         305
## 24         336
## 25         340
## 26         318
## 27         362
## 28         348
## 29         363
## 30         435
## 31         491
## 32         505
## 33         404
## 34         359
## 35         310
## 36         337
## 37         360
```

```
## 38          342
## 39          406
## 40          396
## 41          420
## 42          472
## 43          548
## 44          559
## 45          463
## 46          407
## 47          362
## 48          405
## 49          417
## 50          391
## 51          419
## 52          461
## 53          472
## 54          535
## 55          622
## 56          606
## 57          508
## 58          461
## 59          390
## 60          432
```

```r
colnames(Y)
```

```
## [1] "Level_A"     "Level_B"     "LevelShift"  "Trend_A"     "Trend_B"
## [6] "Season_A"    "Season_B"    "TrendSeason"
```

```r
#Using the column Level_A
y <- Y[,1]
print(y)
```

```
##  [1] 309.5927 285.0966 298.8200 284.3028 308.5171 306.8993 325.4628 326.9226
##  [9] 302.5102 319.5777 332.5051 342.4510 339.0753 334.1232 361.4546 336.1173
## [17] 319.6460 317.3951 328.0461 304.5760 355.3091 330.1500 342.8376 327.5952
## [25] 328.0343 322.7103 329.1623 312.8083 319.9644 310.1535 338.1874 314.8126
## [33] 345.3493 313.4708 304.8354 311.6021 307.4285 347.2009 326.6501 308.4443
## [41] 299.1128 341.9722 302.1470 295.1502 351.9557 332.8738 324.6155 306.7664
## [49] 320.3624 350.8271 317.0764 300.9870 336.8473 316.1264 321.9198 336.2565
## [57] 335.7531 332.0578 324.6613 352.9658
```

```r
y <- ts(y, frequency = 12)
print(y)
```

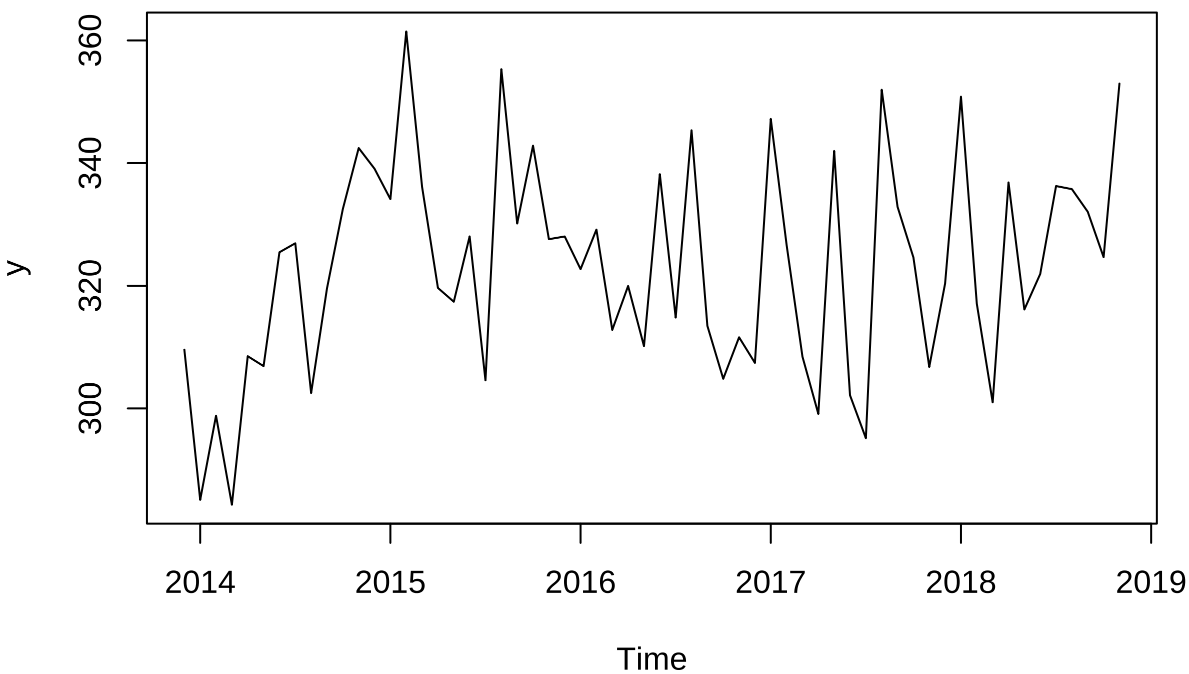
```
##        Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1 309.5927 285.0966 298.8200 284.3028 308.5171 306.8993 325.4628 326.9226
## 2 339.0753 334.1232 361.4546 336.1173 319.6460 317.3951 328.0461 304.5760
## 3 328.0343 322.7103 329.1623 312.8083 319.9644 310.1535 338.1874 314.8126
## 4 307.4285 347.2009 326.6501 308.4443 299.1128 341.9722 302.1470 295.1502
## 5 320.3624 350.8271 317.0764 300.9870 336.8473 316.1264 321.9198 336.2565
##        Sep      Oct      Nov      Dec
## 1 302.5102 319.5777 332.5051 342.4510
## 2 355.3091 330.1500 342.8376 327.5952
## 3 345.3493 313.4708 304.8354 311.6021
## 4 351.9557 332.8738 324.6155 306.7664
```

```
## 5 335.7531 332.0578 324.6613 352.9658
```

```r
y <- ts(y, frequency = 12, end = c(2018, 11))
print(y)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2013
## 2014 285.0966 298.8200 284.3028 308.5171 306.8993 325.4628 326.9226 302.5102
## 2015 334.1232 361.4546 336.1173 319.6460 317.3951 328.0461 304.5760 355.3091
## 2016 322.7103 329.1623 312.8083 319.9644 310.1535 338.1874 314.8126 345.3493
## 2017 347.2009 326.6501 308.4443 299.1128 341.9722 302.1470 295.1502 351.9557
## 2018 350.8271 317.0764 300.9870 336.8473 316.1264 321.9198 336.2565 335.7531
##           Sep      Oct      Nov      Dec
## 2013                            309.5927
## 2014 319.5777 332.5051 342.4510 339.0753
## 2015 330.1500 342.8376 327.5952 328.0343
## 2016 313.4708 304.8354 311.6021 307.4285
## 2017 332.8738 324.6155 306.7664 320.3624
## 2018 332.0578 324.6613 352.9658
```

```r
plot(y)
```



Constructing estimation and hold-out sets

```r
y.test <- tail(y, 12)
print(y.test)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2017
## 2018 350.8271 317.0764 300.9870 336.8473 316.1264 321.9198 336.2565 335.7531
##           Sep      Oct      Nov      Dec
## 2017                            320.3624
## 2018 332.0578 324.6613 352.9658
```

```r
y.train <- head(y, 48)
print(y.train)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2013                                                                        
## 2014 285.0966 298.8200 284.3028 308.5171 306.8993 325.4628 326.9226 302.5102
## 2015 334.1232 361.4546 336.1173 319.6460 317.3951 328.0461 304.5760 355.3091
## 2016 322.7103 329.1623 312.8083 319.9644 310.1535 338.1874 314.8126 345.3493
## 2017 347.2009 326.6501 308.4443 299.1128 341.9722 302.1470 295.1502 351.9557
##           Sep      Oct      Nov      Dec
## 2013                            309.5927
## 2014 319.5777 332.5051 342.4510 339.0753
## 2015 330.1500 342.8376 327.5952 328.0343
## 2016 313.4708 304.8354 311.6021 307.4285
## 2017 332.8738 324.6155 306.7664
```

```r
yy <- y[1:48]
print(yy)
```

```
##  [1] 309.5927 285.0966 298.8200 284.3028 308.5171 306.8993 325.4628 326.9226
##  [9] 302.5102 319.5777 332.5051 342.4510 339.0753 334.1232 361.4546 336.1173
## [17] 319.6460 317.3951 328.0461 304.5760 355.3091 330.1500 342.8376 327.5952
## [25] 328.0343 322.7103 329.1623 312.8083 319.9644 310.1535 338.1874 314.8126
## [33] 345.3493 313.4708 304.8354 311.6021 307.4285 347.2009 326.6501 308.4443
## [41] 299.1128 341.9722 302.1470 295.1502 351.9557 332.8738 324.6155 306.7664
```

```r
class(y) # Our time series object
```

```
## [1] "ts"
```

```r
class(yy) # A simple vector of numeric value
```

```
## [1] "numeric"
```

```r
yy <- ts(yy, frequency=frequency(y), start=start(y))
print(yy)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2013                                                                        
## 2014 285.0966 298.8200 284.3028 308.5171 306.8993 325.4628 326.9226 302.5102
## 2015 334.1232 361.4546 336.1173 319.6460 317.3951 328.0461 304.5760 355.3091
## 2016 322.7103 329.1623 312.8083 319.9644 310.1535 338.1874 314.8126 345.3493
## 2017 347.2009 326.6501 308.4443 299.1128 341.9722 302.1470 295.1502 351.9557
##           Sep      Oct      Nov      Dec
## 2013                            309.5927
## 2014 319.5777 332.5051 342.4510 339.0753
## 2015 330.1500 342.8376 327.5952 328.0343
## 2016 313.4708 304.8354 311.6021 307.4285
## 2017 332.8738 324.6155 306.7664
```

```r
class(yy)
```

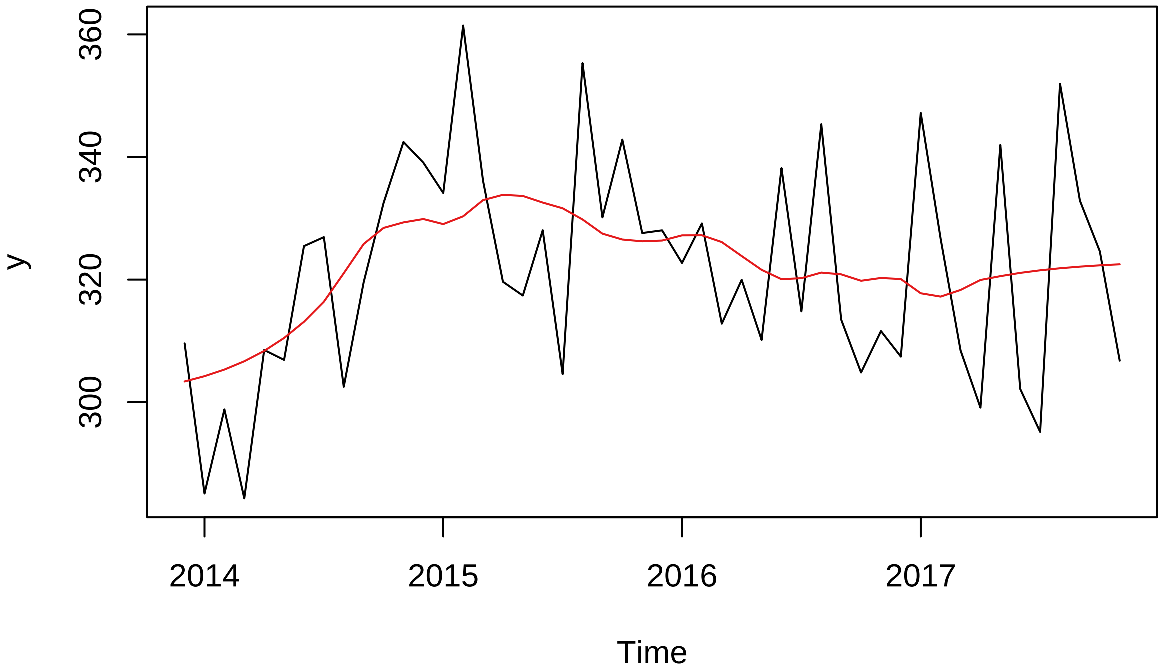
```
## [1] "ts"
```

```r
all(yy==y.train)
```

```
## [1] TRUE
```

Exploring a time series

```r
cma <- cmav(y.train, outplot = 1)
```



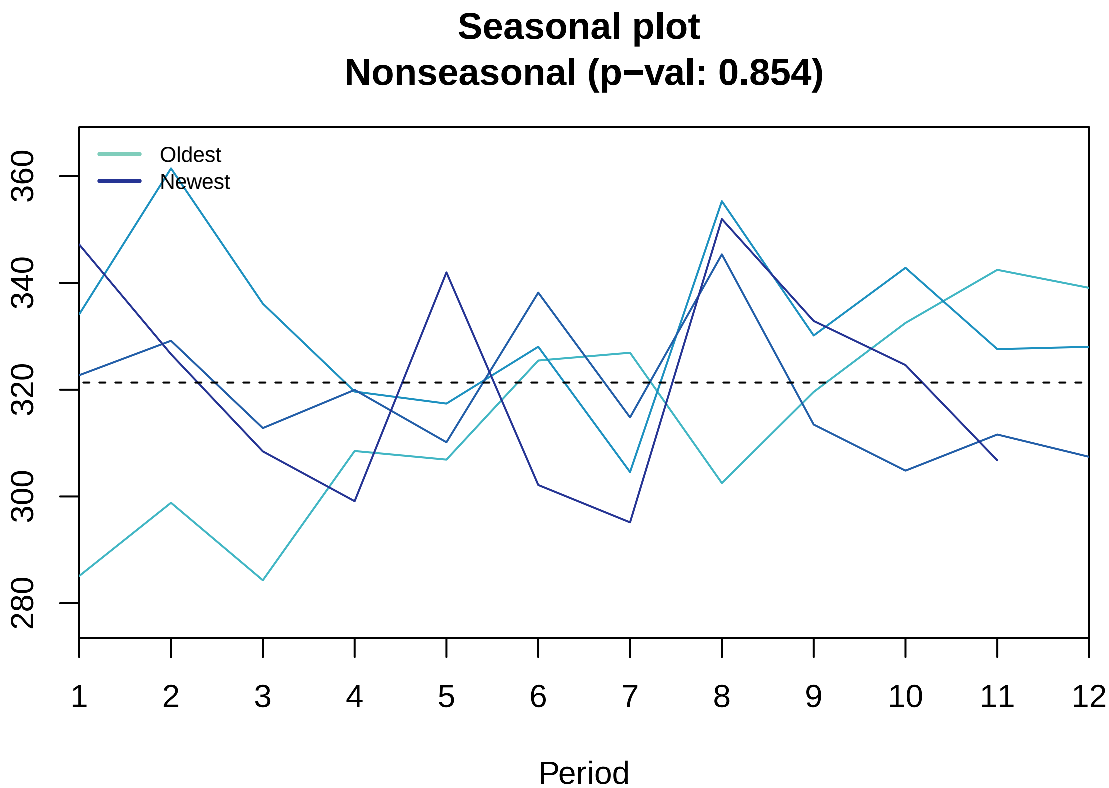
Question: Is this time series trended?

Answer: No, there is a slight positive development visible in the data but not enough to call it a trend.

```r
print(cma)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2013
## 2014 304.2395 305.3201 306.6708 308.3591 310.4696 313.1166 316.3878 321.0404
## 2015 329.0542 330.3231 332.9636 333.8346 333.6461 332.5671 331.6315 329.8105
## 2016 327.2172 327.2288 326.1188 323.8404 321.5906 320.0656 320.2275 321.1433
## 2017 317.7662 317.2222 318.3059 319.9386 320.5613 321.0955 321.5123 321.8458
##           Sep      Oct      Nov      Dec
## 2013                            303.3751
## 2014 325.8091 328.4317 329.3328 329.8777
## 2015 327.4937 326.5358 326.2473 326.3682
## 2016 320.8568 319.8061 320.2631 320.0872
## 2017 322.1126 322.3260 322.4968
```

```r
seasplot(y.train)
```

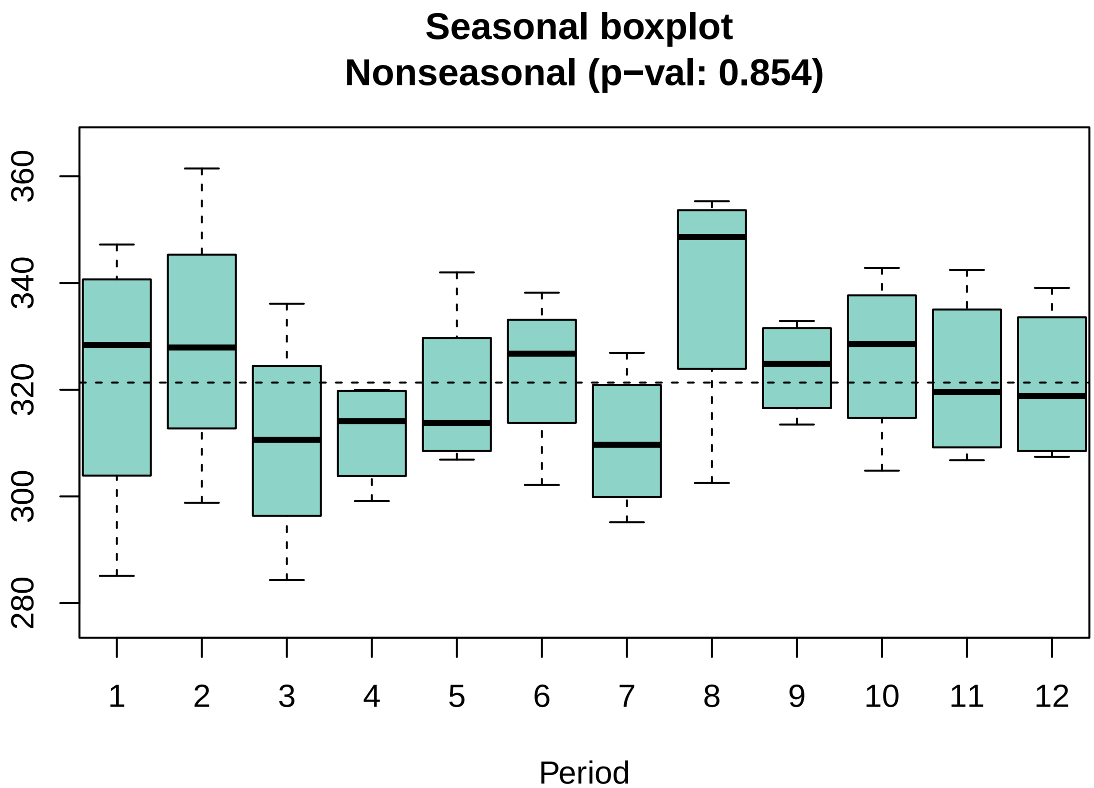



```
## Results of statistical testing
## Evidence of trend: FALSE  (pval: 0.154)
## Evidence of seasonality: FALSE  (pval: 0.854)
```

Question: Is this time series seasonal?

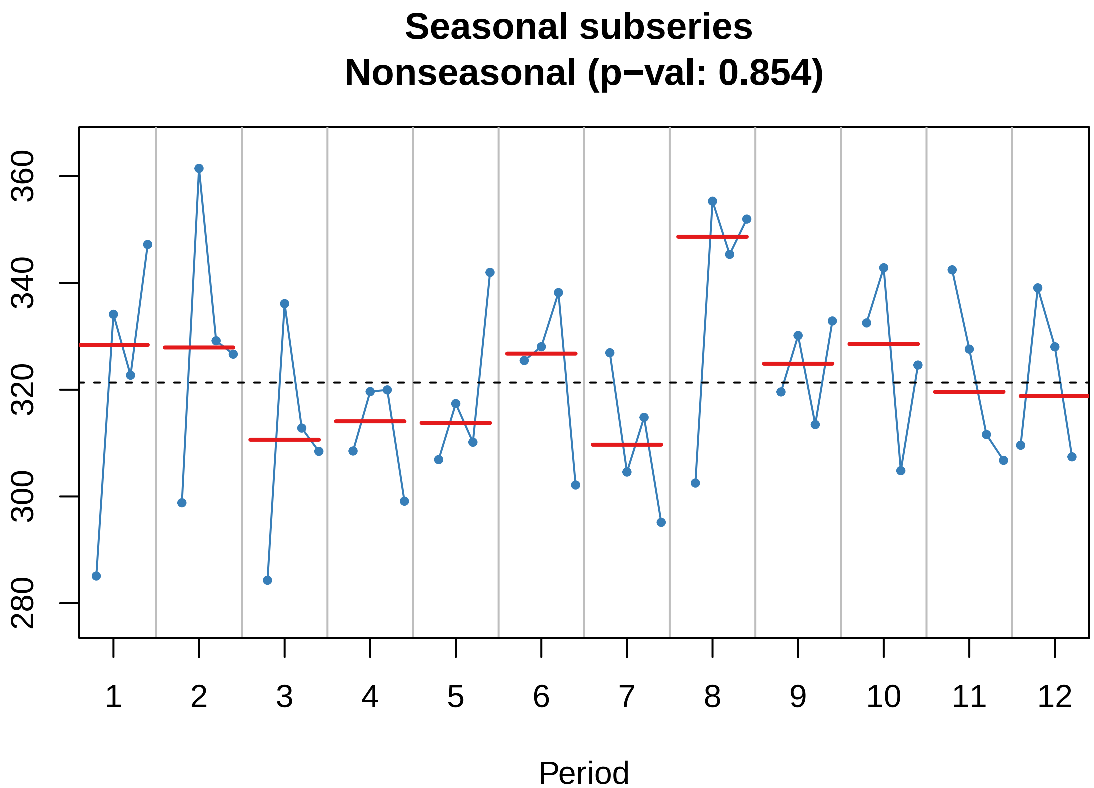
Answer: No, this time series is not seasonal since the 4 different years do not follow a similar pattern each month.

```r
seasplot(y.train,outplot=2) # Boxplots of the values of each month
```



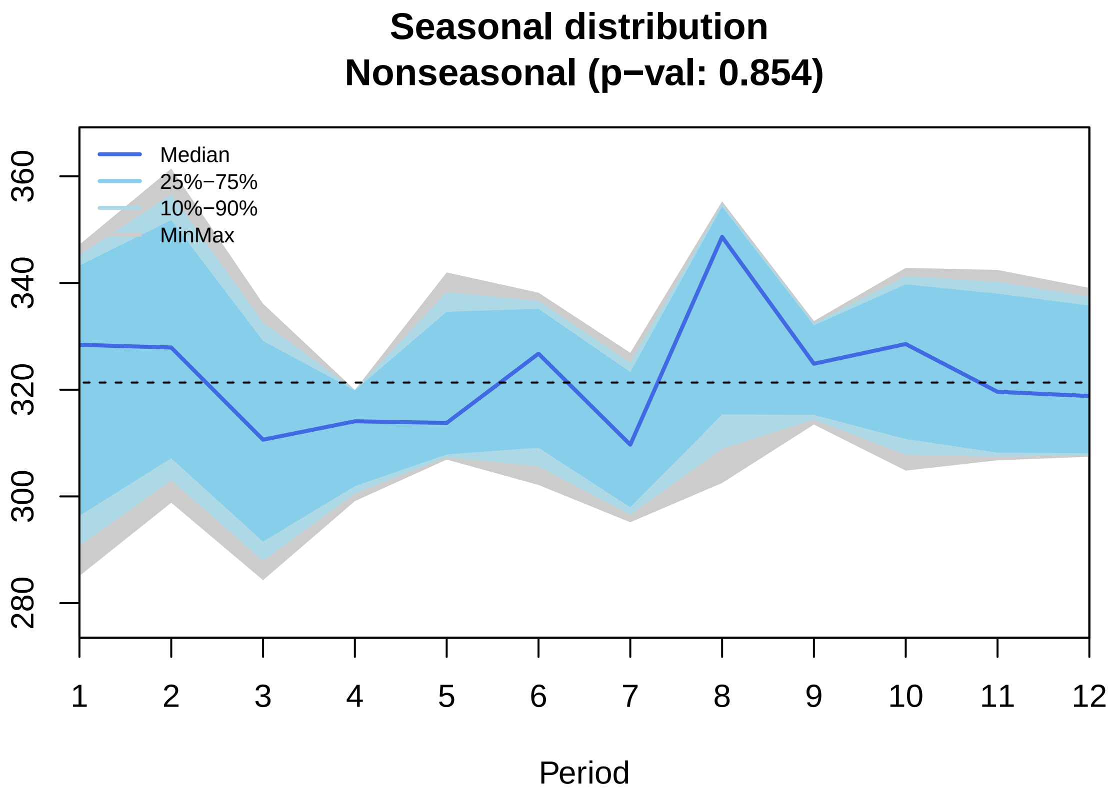

```
## Results of statistical testing
## Evidence of trend: FALSE  (pval: 0.154)
## Evidence of seasonality: FALSE  (pval: 0.854)
```

```
# The average (red) value for each month with a series of each month across years (blue)
seasplot(y.train,outplot=3)
```



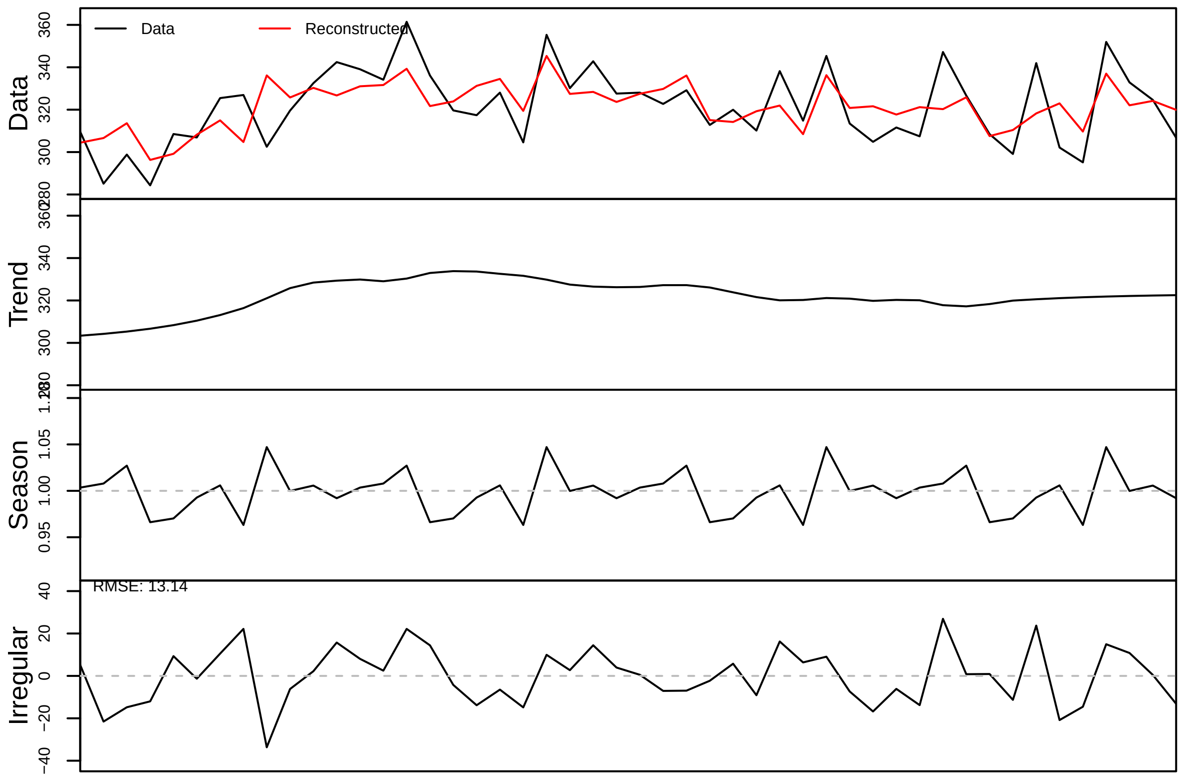

```
## Results of statistical testing
## Evidence of trend: FALSE  (pval: 0.154)
## Evidence of seasonality: FALSE  (pval: 0.854)
```

```
seasplot(y.train,outplot=4) #A`connected' boxplot across months.
```




```
## Results of statistical testing
## Evidence of trend: FALSE  (pval: 0.154)
## Evidence of seasonality: FALSE  (pval: 0.854)
```

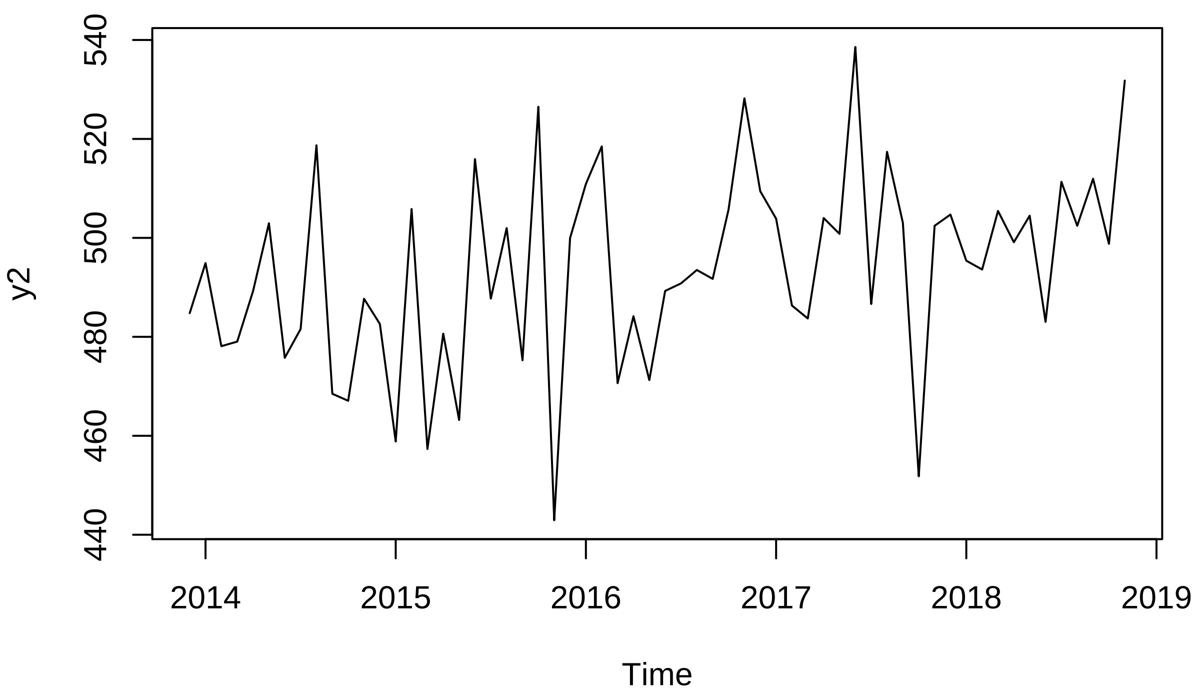
```
dc <-decomp(y.train,outplot=1)
```



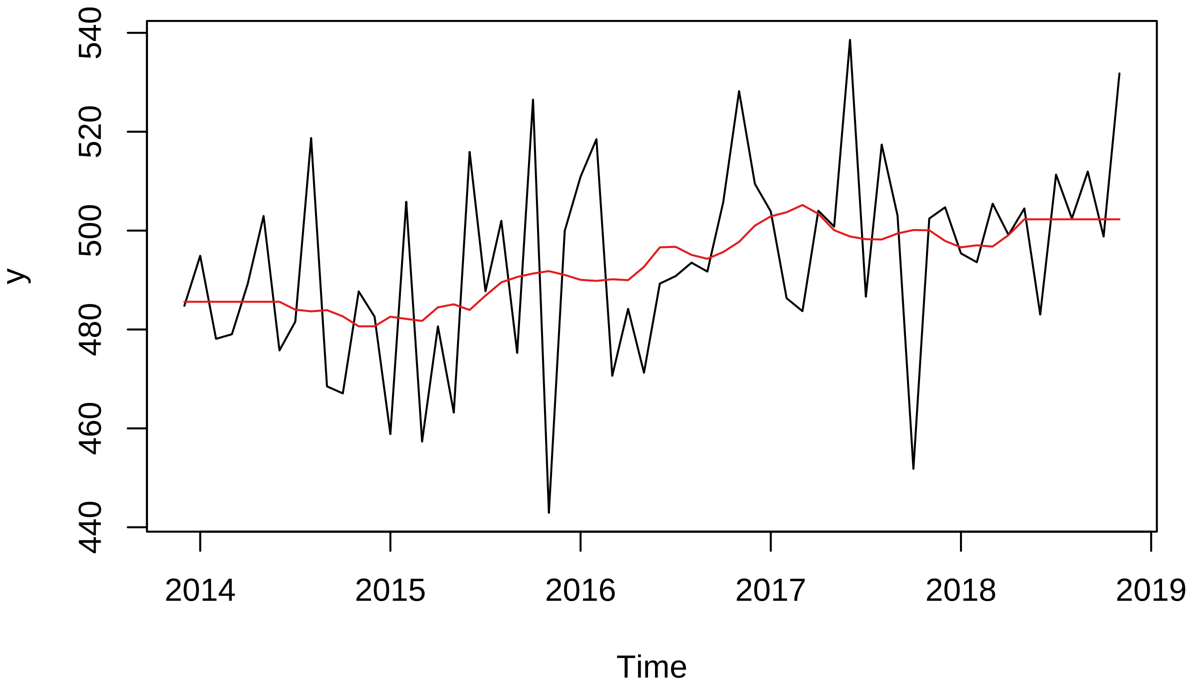
Exercises

1. Data Exploration

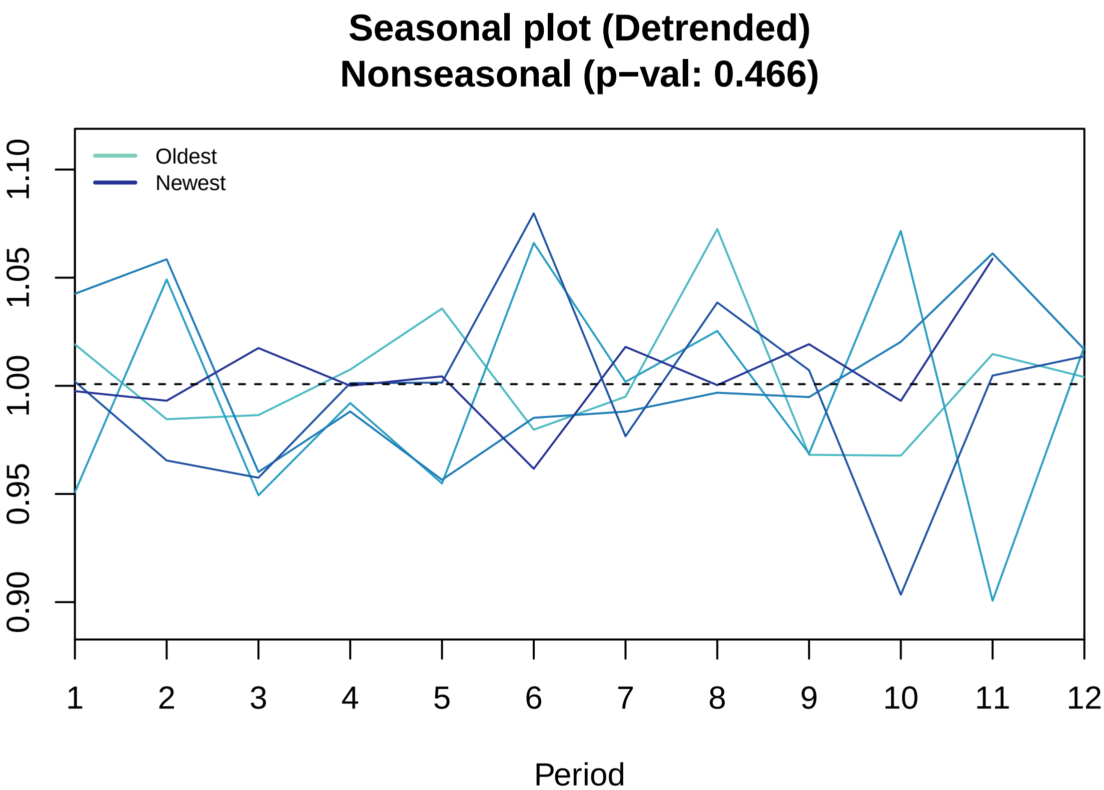
```r
#Using the column Level_B
y2 <- ts(Y[,2], frequency = 12, end = c(2018, 11))
plot(y2)
```



```r
cma2 <- cmav(y2, outplot = 1)
```

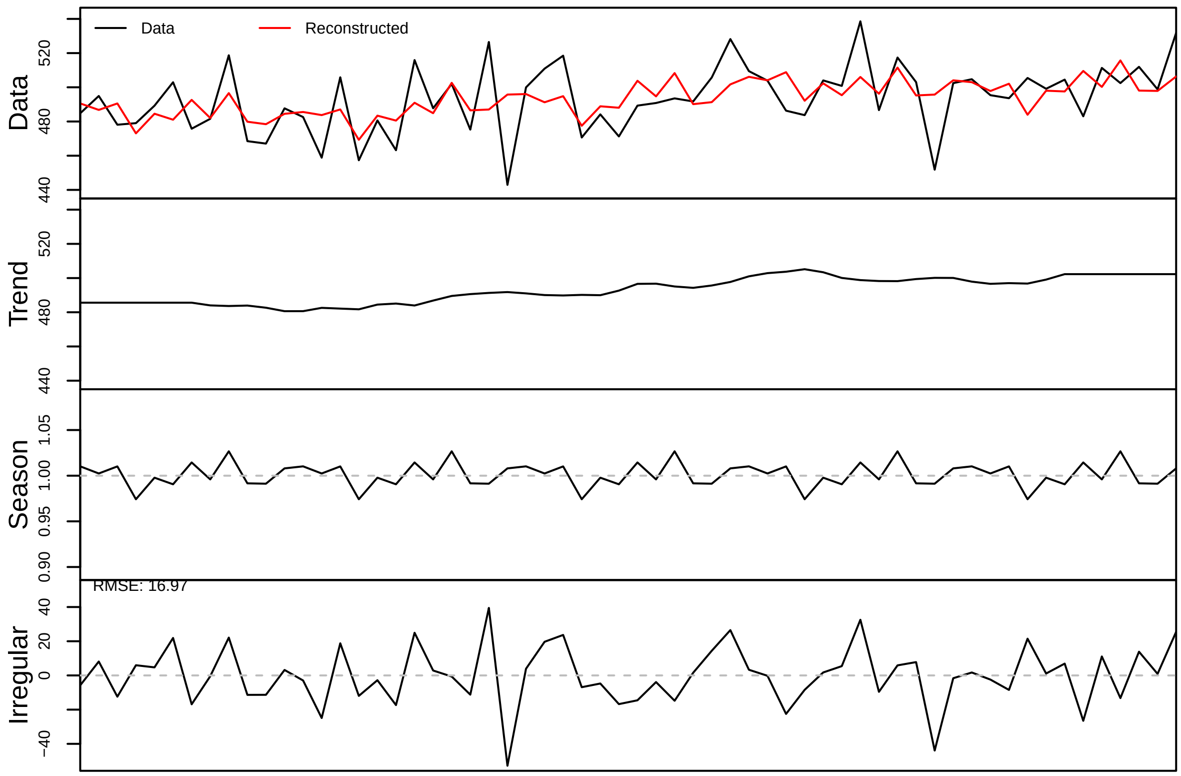



```r
seasplot(y2)
```




```
## Results of statistical testing
## Evidence of trend: TRUE  (pval: 0)
## Evidence of seasonality: FALSE  (pval: 0.466)
```

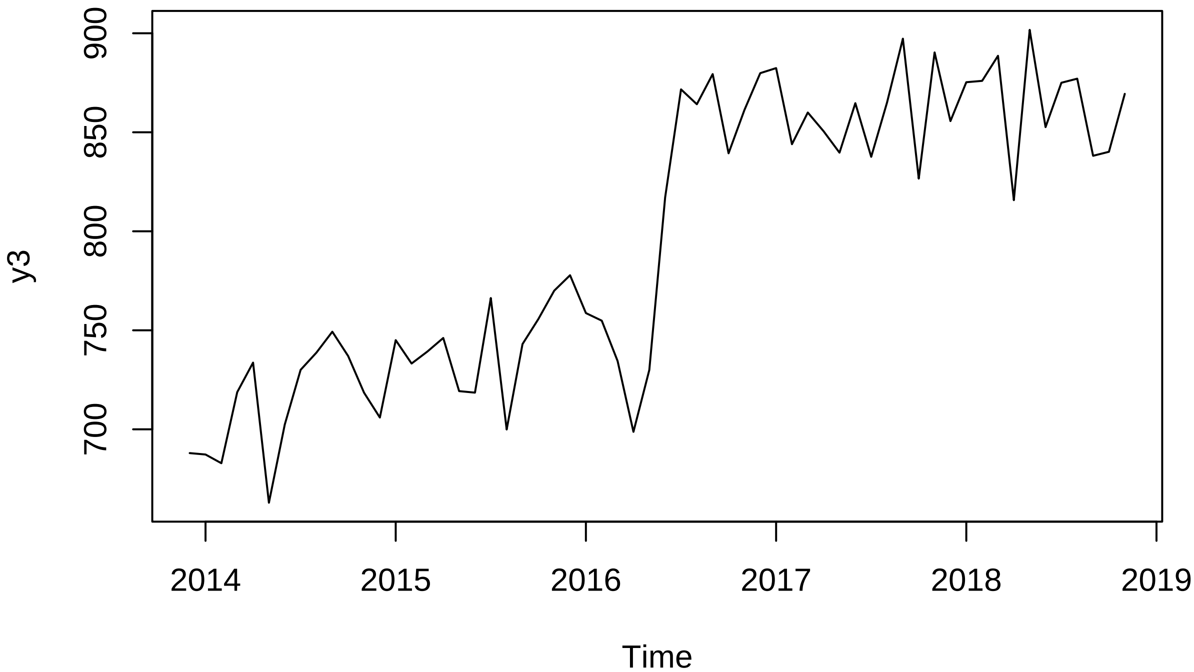
```r
dc2 <-decomp(y2 ,outplot=1)
```



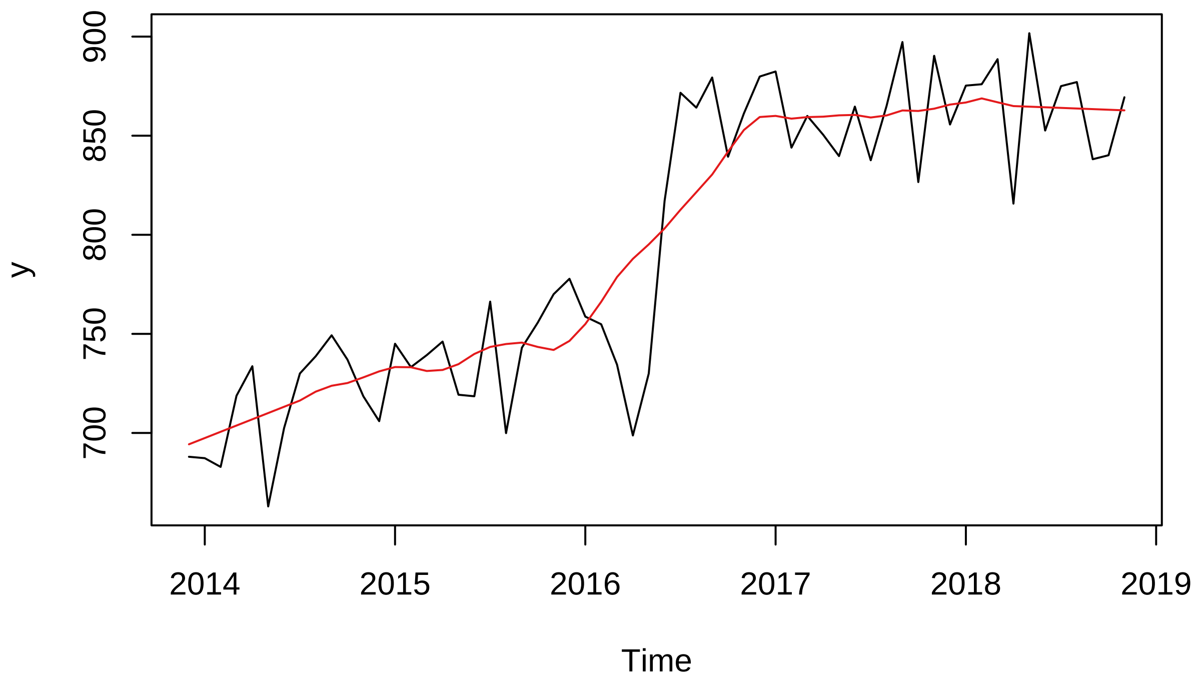
Question: Does your understanding of the plots agree with the underlying model?

Answer: The column Level B is similar to Level A where there is no clear seasonality and the trend is more additive than in Level A but not completely linear. Therefore it fits well to the description Level B since the time series stays on one level for the most part.

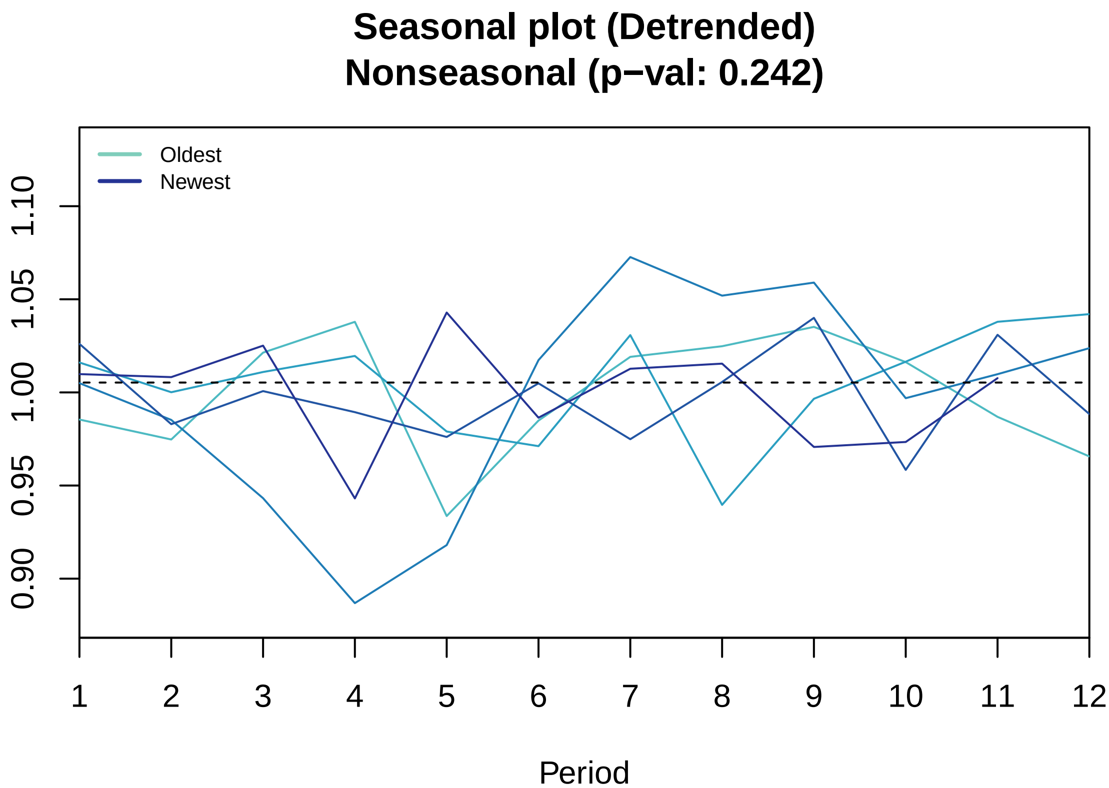
```r
#Using the column LevelShift
y3 <- ts(Y[,3], frequency = 12, end = c(2018, 11))
plot(y3)
```




```
cma3 <- cmav(y3, outplot = 1)
```

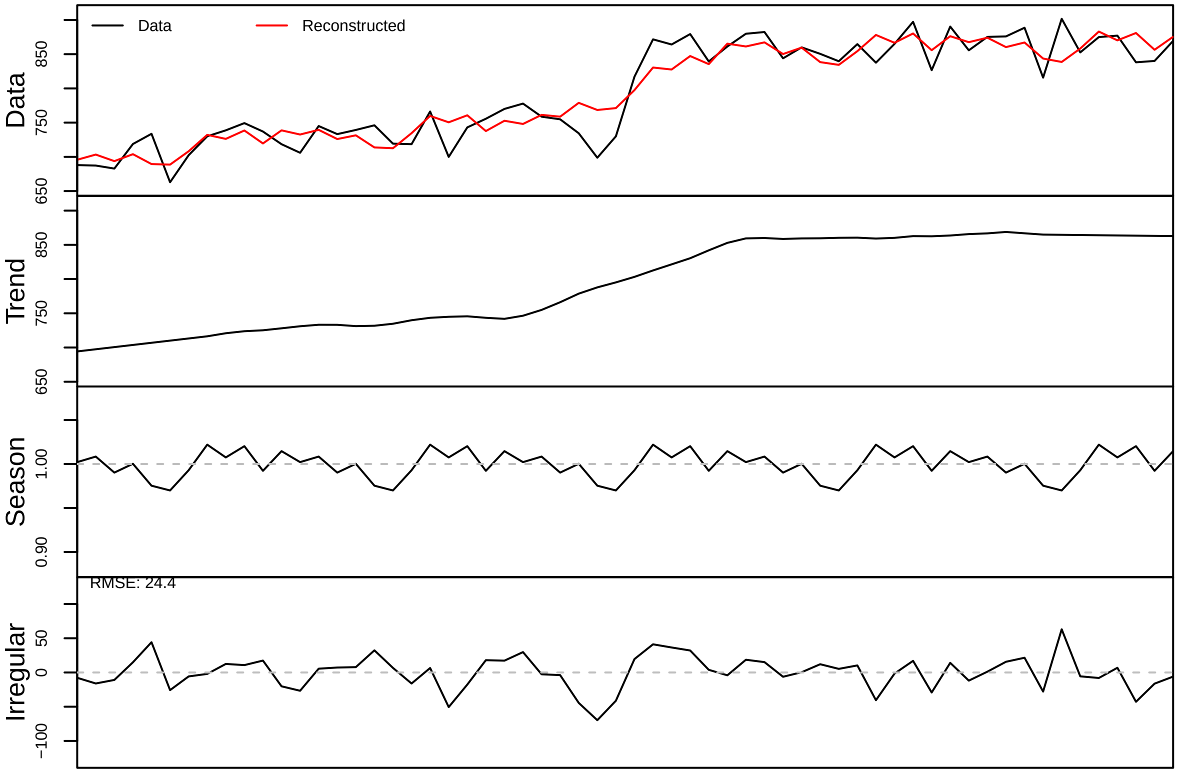



```
seasplot(y3)
```




```
## Results of statistical testing
## Evidence of trend: TRUE  (pval: 0)
## Evidence of seasonality: FALSE  (pval: 0.242)
```

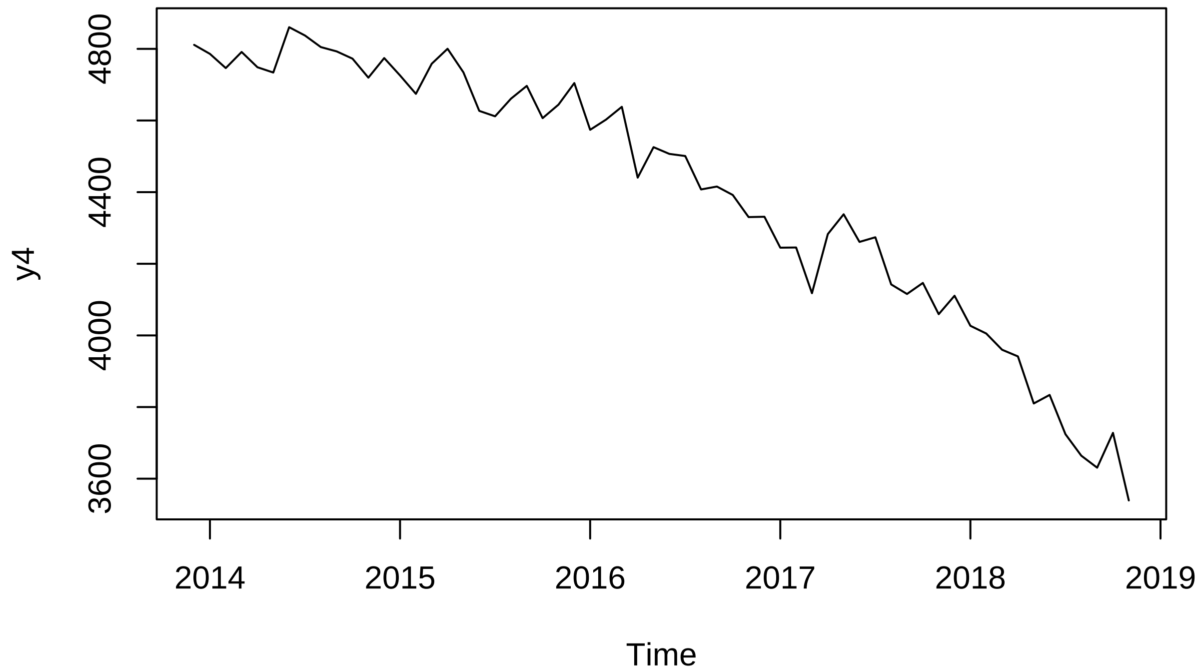
```
dc3 <-decomp(y3 ,outplot=1)
```



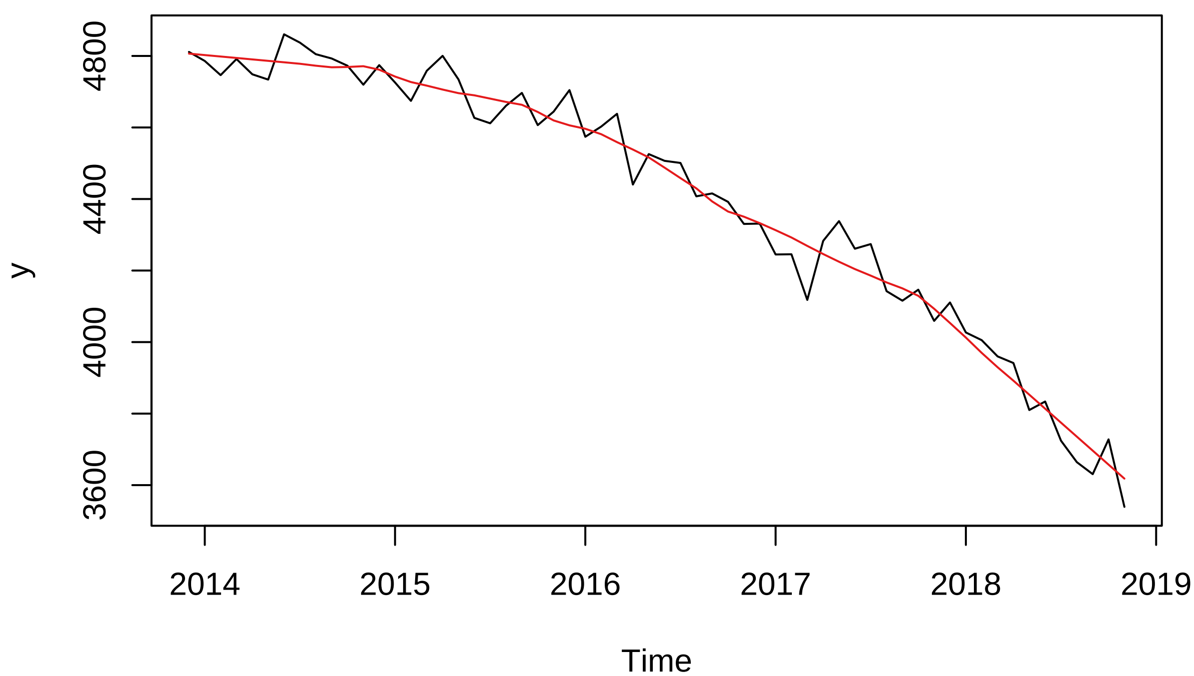
Question: Does your understanding of the plots agree with the underlying model?

Answer: There is a clear shift visible in the level where the years 2014 - 2016 are having a CMA between 700 and 750 and the later years (2017 - 2019) are experiencing a CMA around 850. This shift in level happend between the start of 2016 and the end of 2017, therefore the model name LevelShift fits very well to this time series.

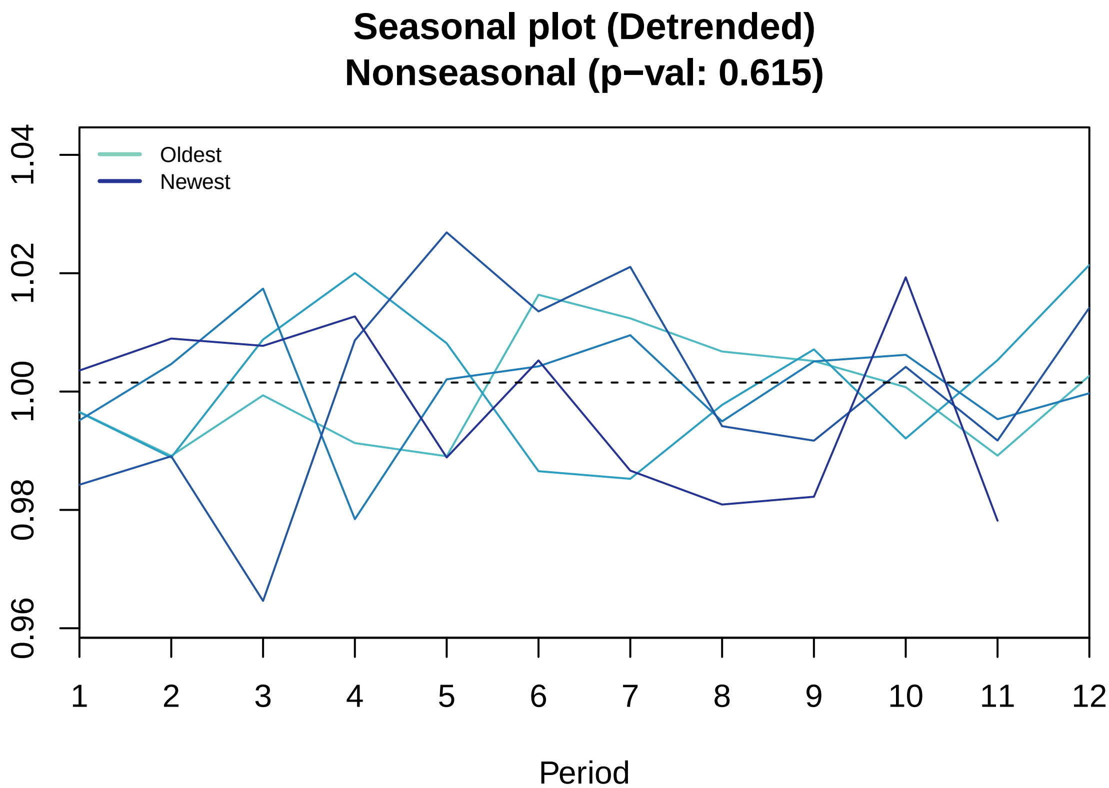
```
#Using the column Trend_A
y4 <- ts(Y[,4], frequency = 12, end = c(2018, 11))
plot(y4)
```




```
cma4 <- cmav(y4, outplot = 1)
```

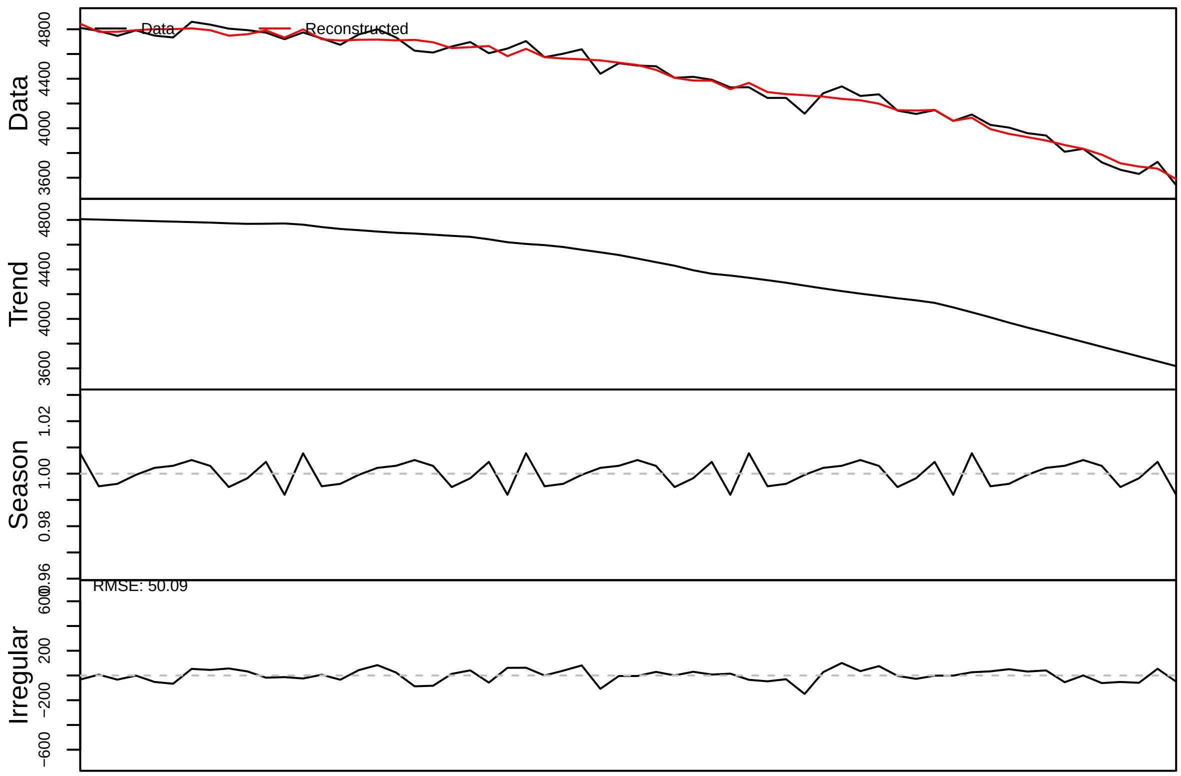



```
seasplot(y4)
```




```
## Results of statistical testing
## Evidence of trend: TRUE  (pval: 0)
## Evidence of seasonality: FALSE  (pval: 0.615)
```

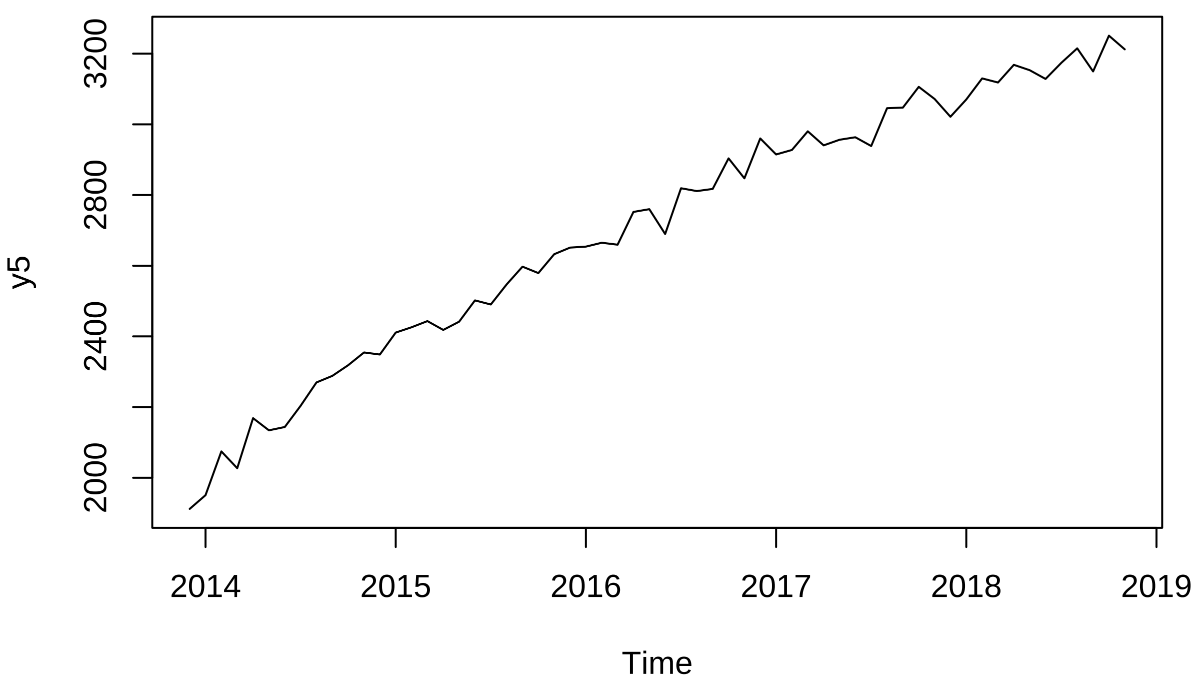
```
dc4 <-decomp(y4 ,outplot=1)
```



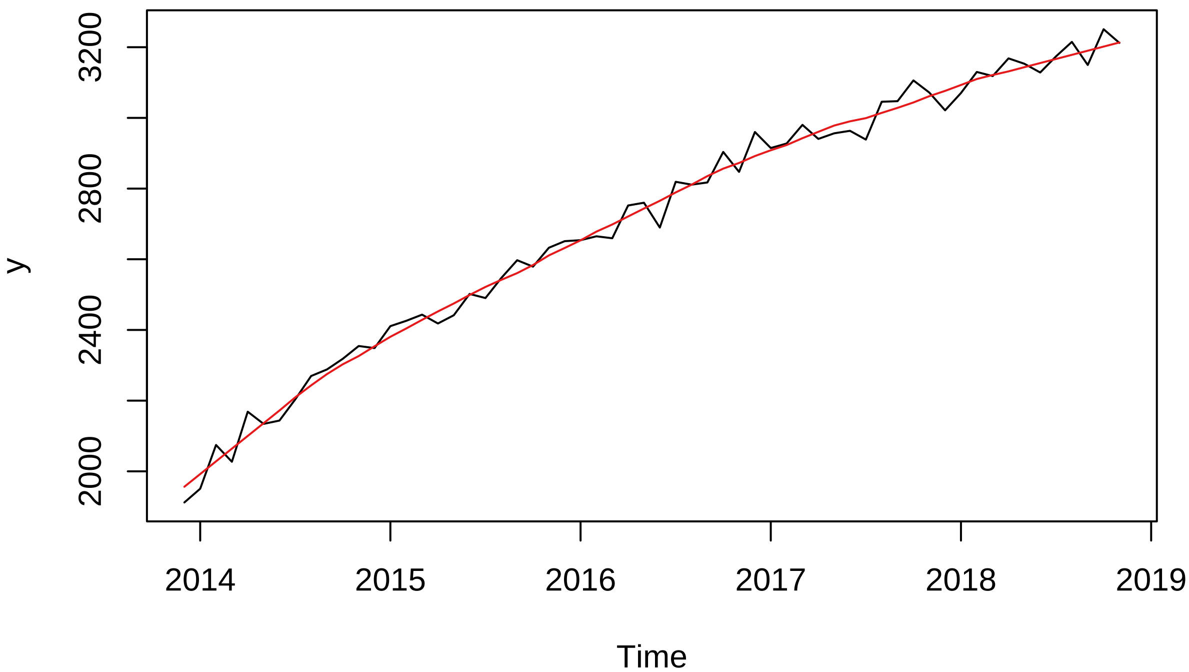
Question: Does your understanding of the plots agree with the underlying model?

Answer: There is a negative trend in the time series which can be seen on the CMA (red line) that is negatively damped additive and therefore with time gets lower and lower. The negative trend is very visible and therefore the model name Trend A also fits the time series.

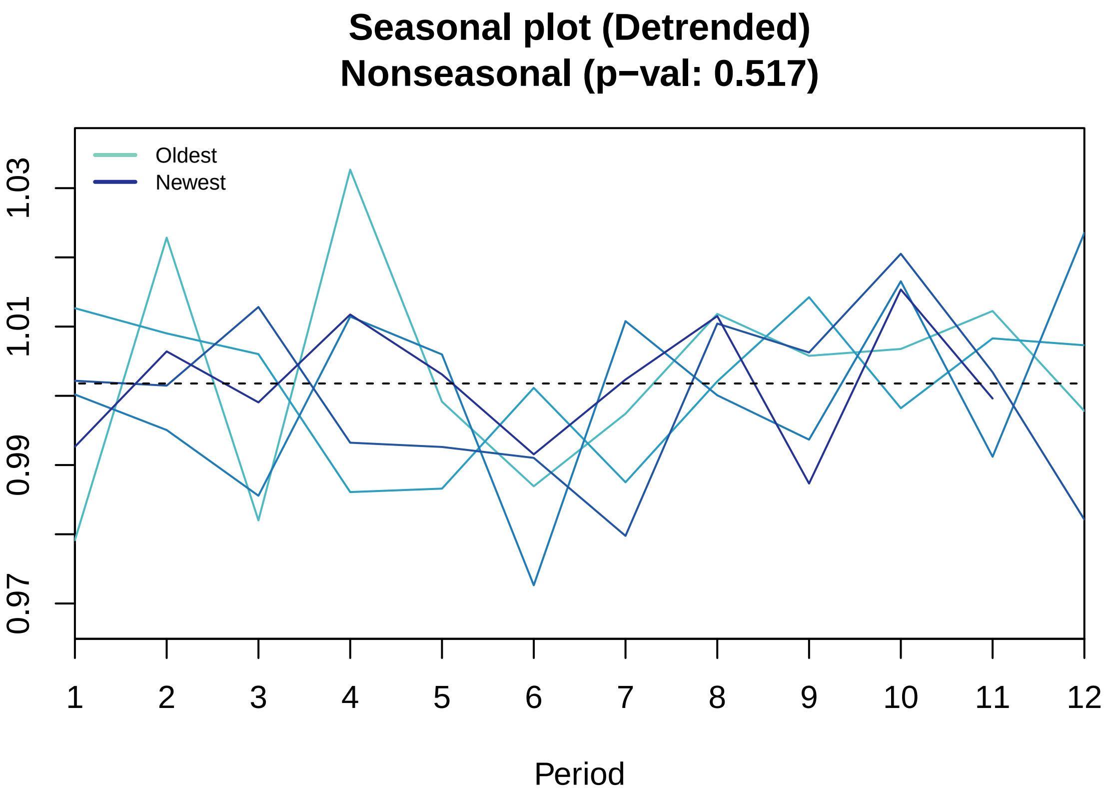
```r
#Using the column Trend_B
y5 <- ts(Y[,5], frequency = 12, end = c(2018, 11))
plot(y5)
```



```r
cma5 <- cmav(y5, outplot = 1)
```

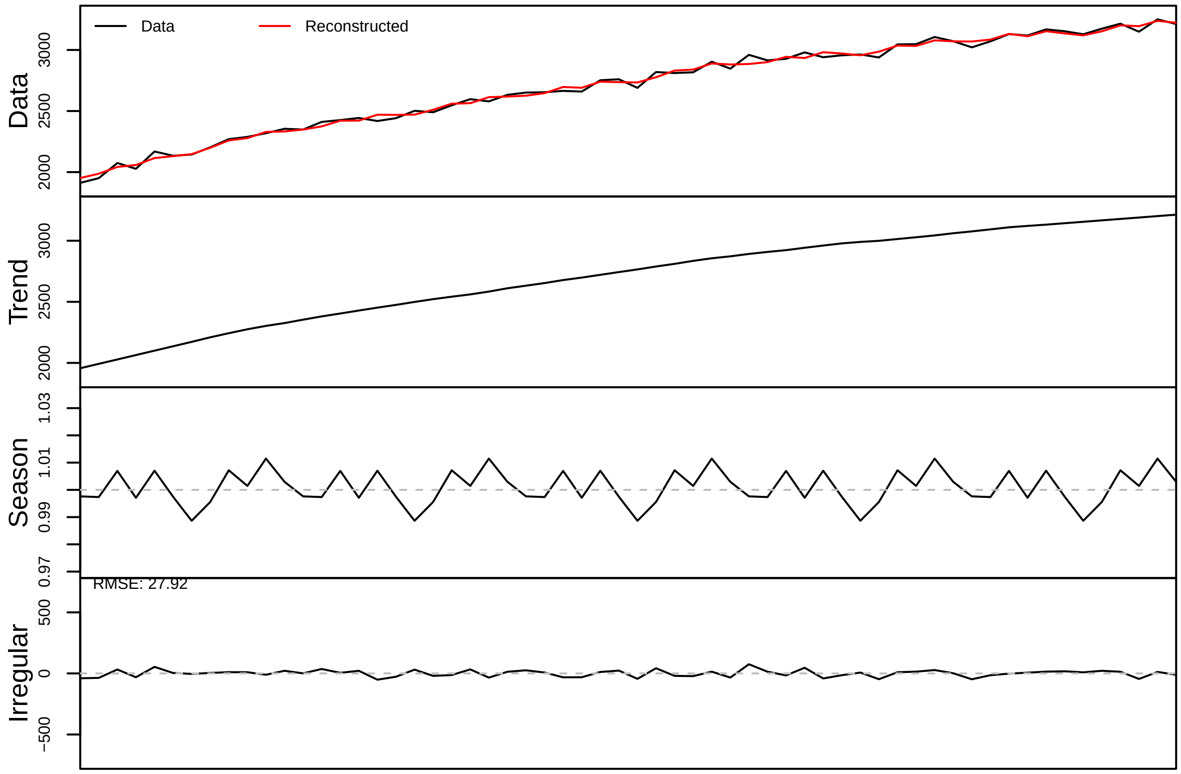



```r
seasplot(y5)
```




```
## Results of statistical testing
## Evidence of trend: TRUE  (pval: 0)
## Evidence of seasonality: FALSE  (pval: 0.517)
```

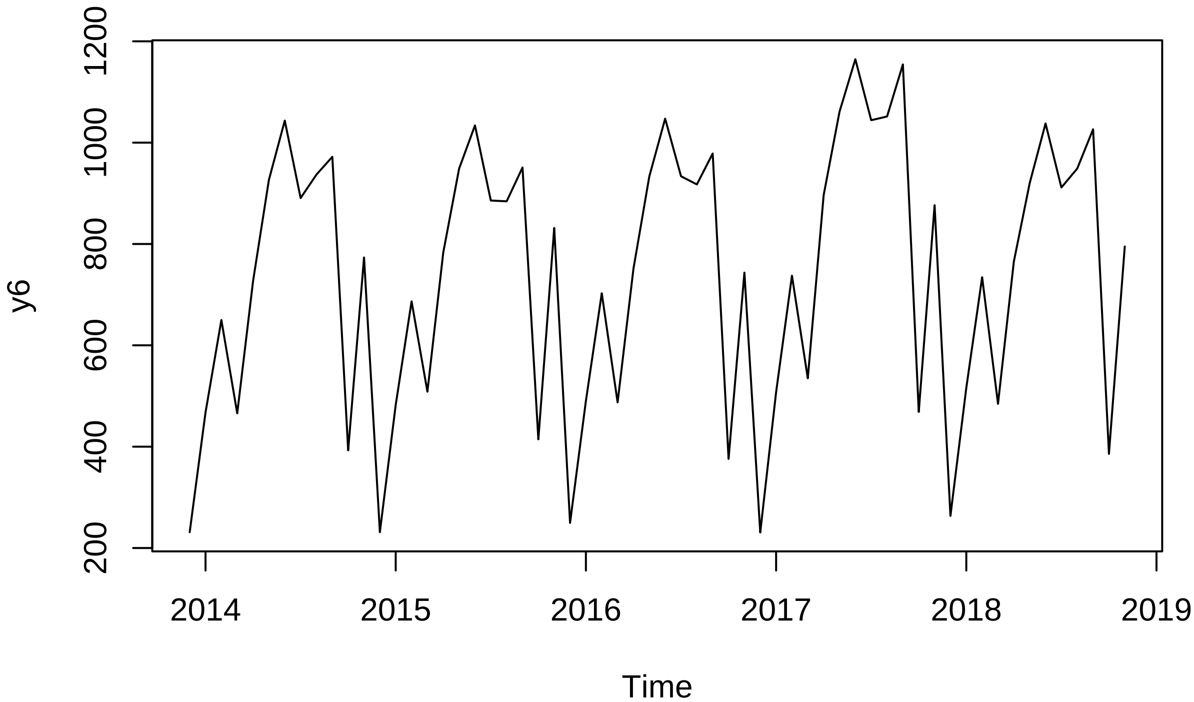
```r
dc5 <-decomp(y5 ,outplot=1)
```



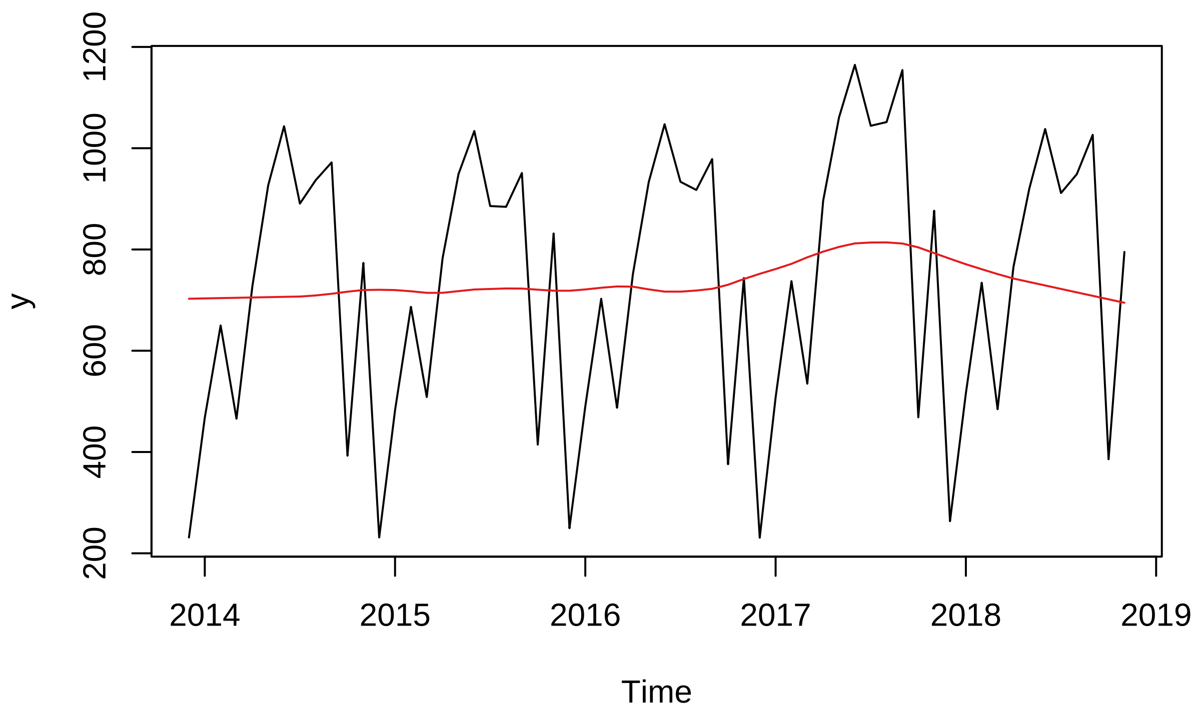
Question: Does your understanding of the plots agree with the underlying model?

Answer: There is a positive trend in the time series which can be seen on the CMA (red line) that is damped additive and therefore with time gets higher and higher with every year. The positive trend is very visible and therefore the model name Trend B also fits the time series.

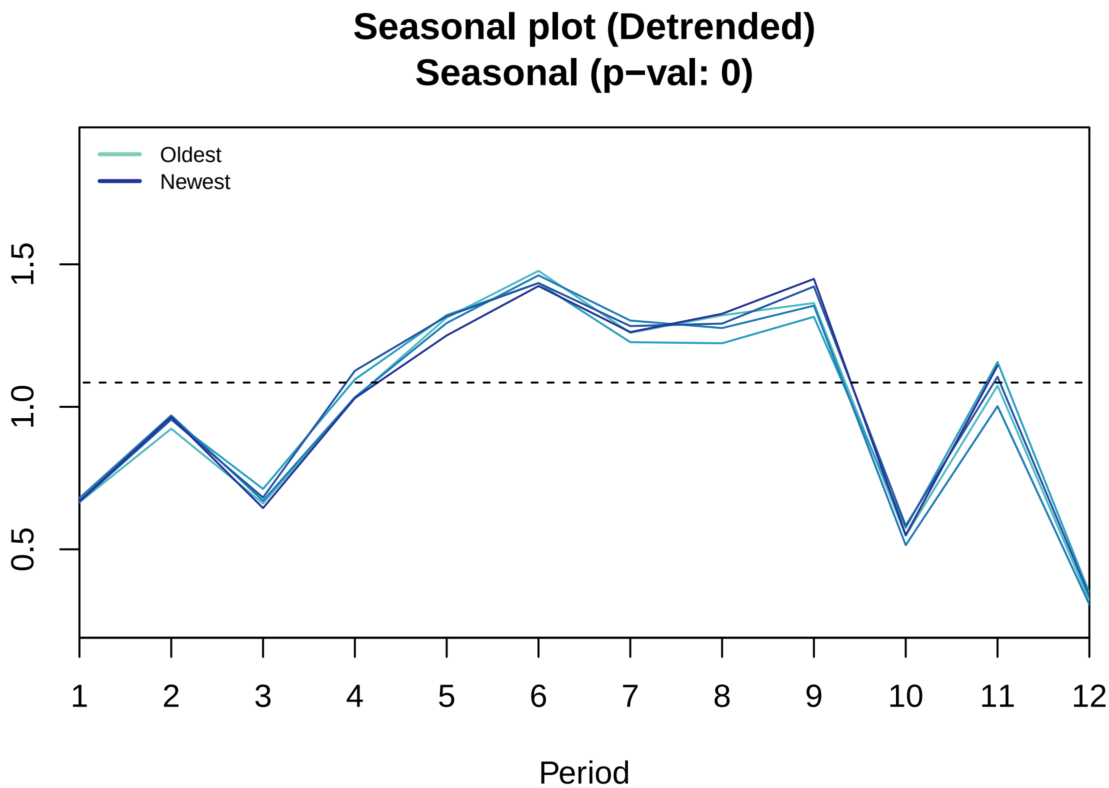
```r
#Using the column Season_A
y6 <- ts(Y[,6], frequency = 12, end = c(2018, 11))
plot(y6)
```




```r
cma6 <- cmav(y6, outplot = 1)
```

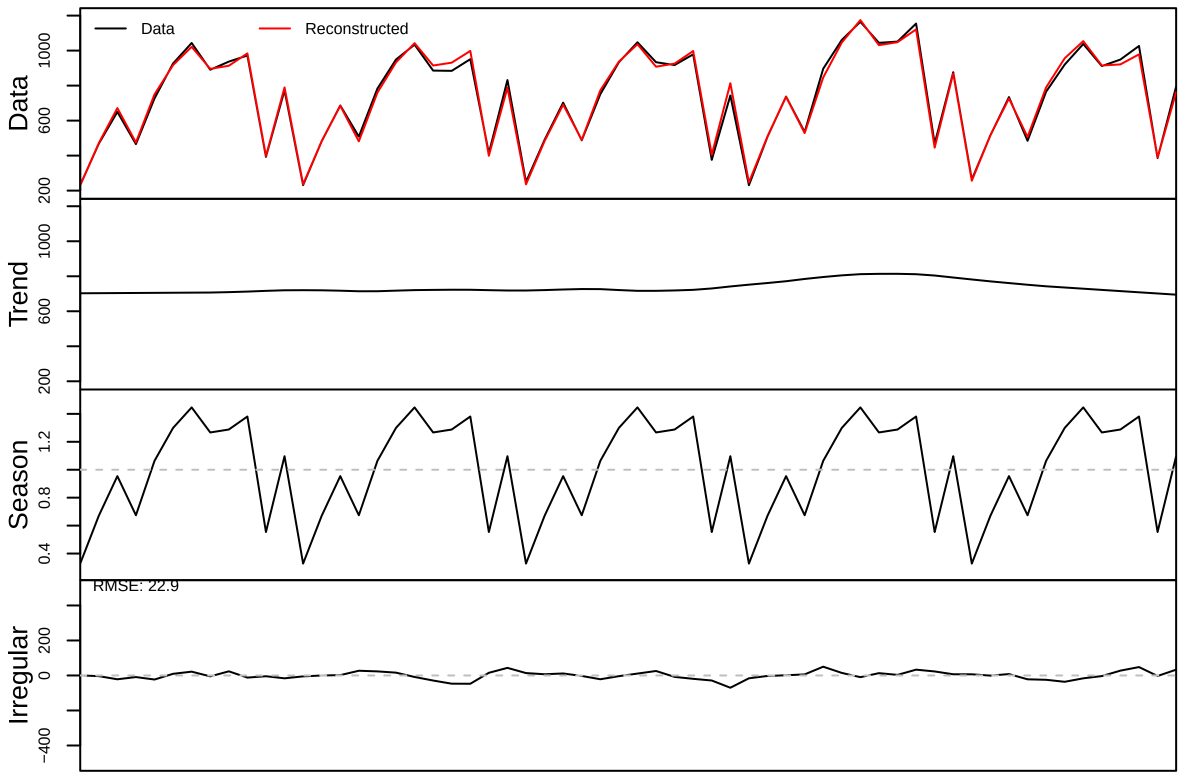



```r
seasplot(y6)
```




```
## Results of statistical testing
## Evidence of trend: TRUE  (pval: 0)
## Evidence of seasonality: TRUE  (pval: 0)
```

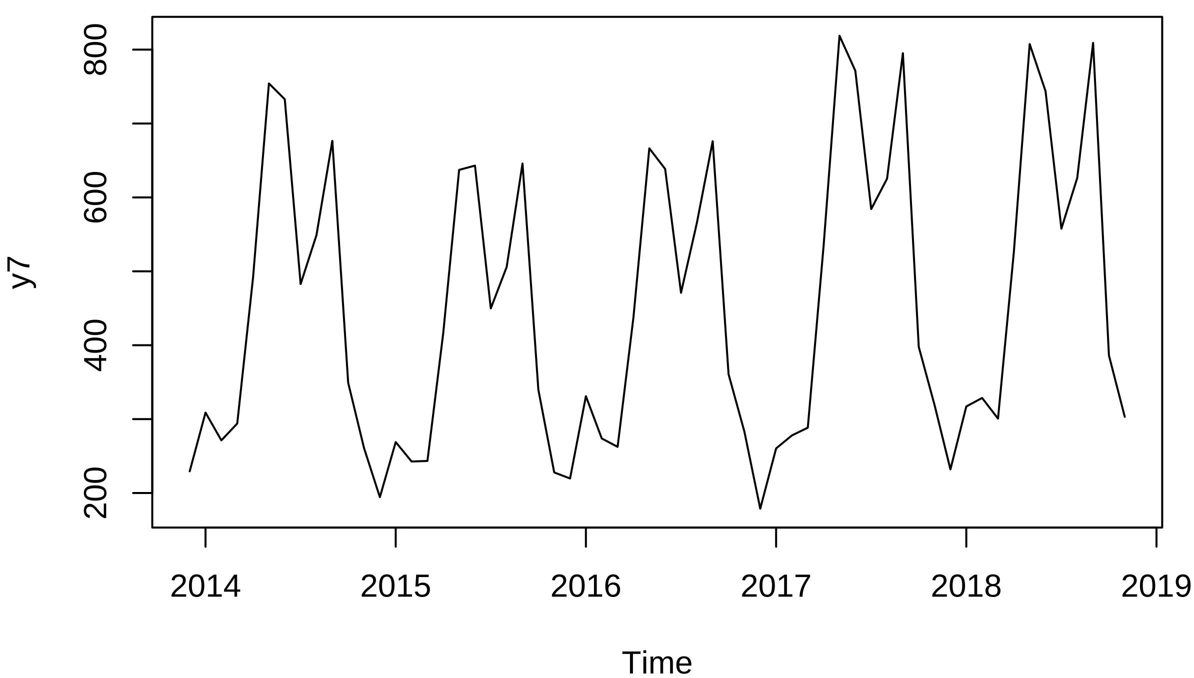
```r
dc6 <-decomp(y6 ,outplot=1)
```



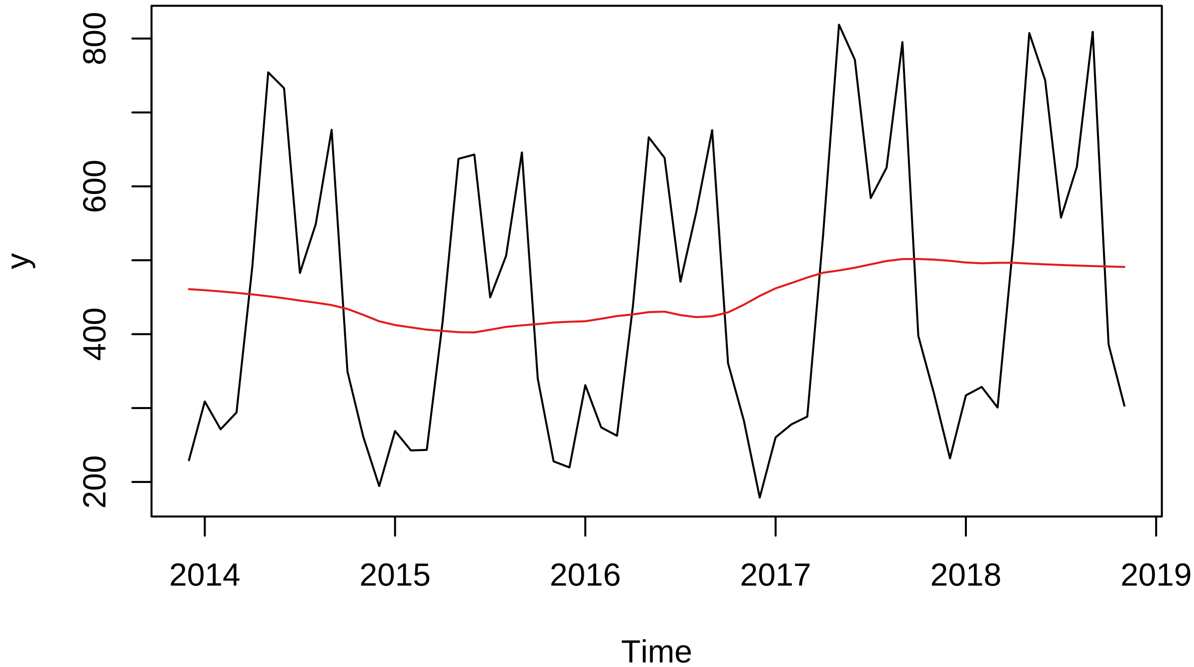
Question: Does your understanding of the plots agree with the underlying model?

Answer: Based on the seasonal plot we can see that the time series is very seasonal since the lines from each year follow exactly the same pattern. Therefore the name Season A fits well for this time series.

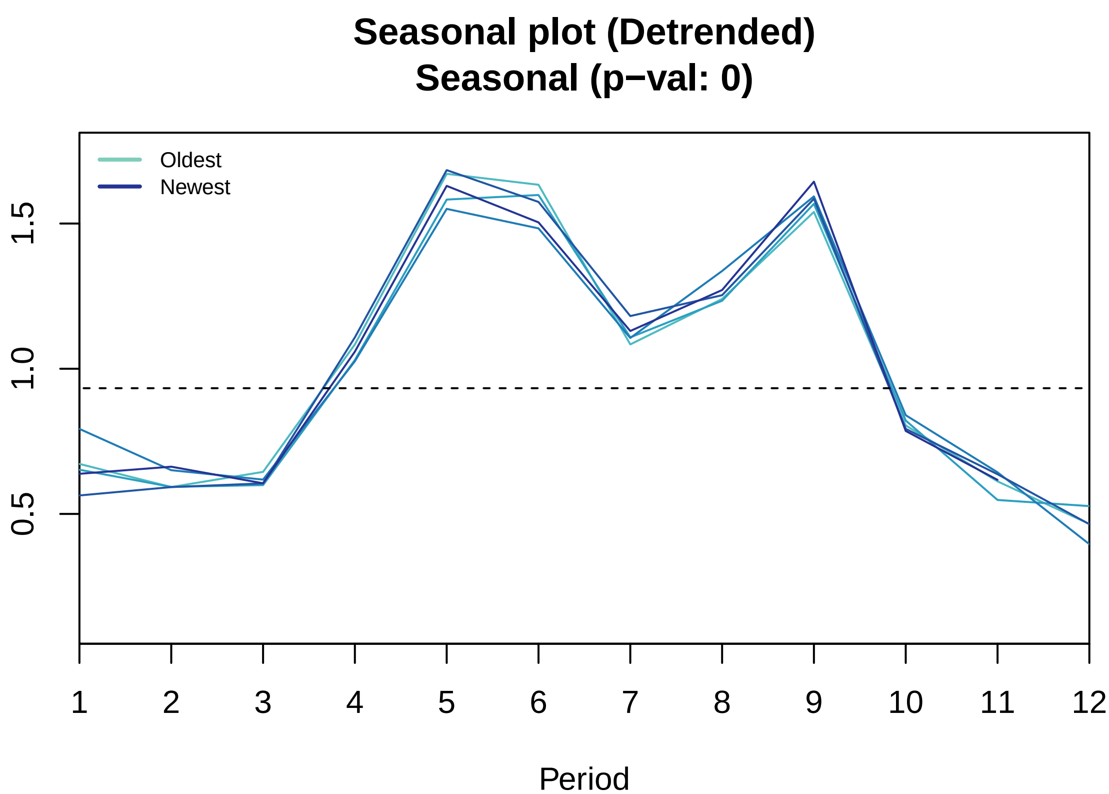
```r
#Using the column Season_B
y7 <- ts(Y[,7], frequency = 12, end = c(2018, 11))
plot(y7)
```



```r
cma7 <- cmav(y7, outplot = 1)
```

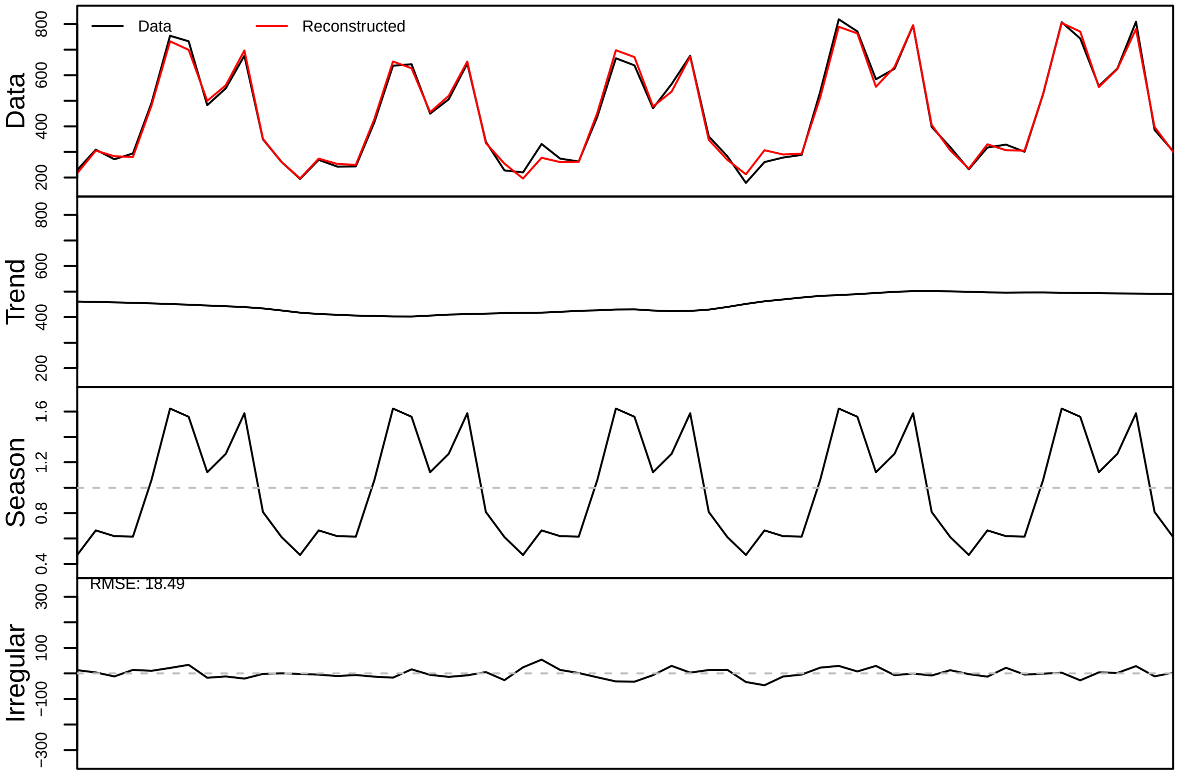



```r
seasplot(y7)
```




```
## Results of statistical testing
## Evidence of trend: TRUE  (pval: 0.001)
## Evidence of seasonality: TRUE  (pval: 0)
```

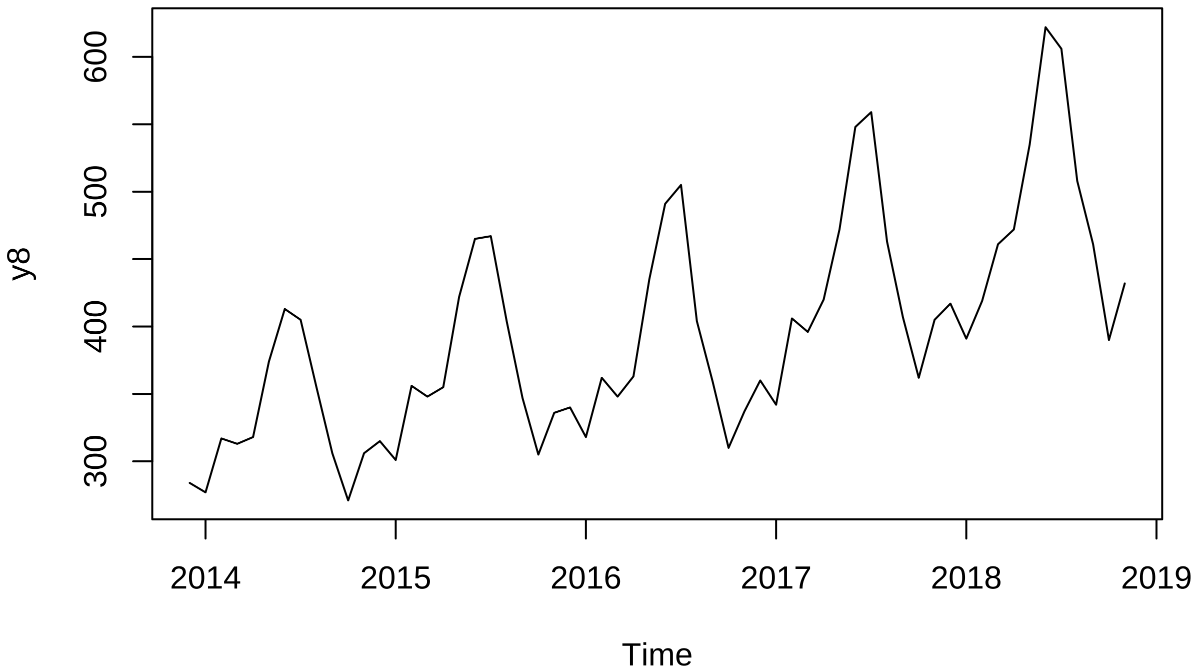
```r
dc7 <-decomp(y7 ,outplot=1)
```



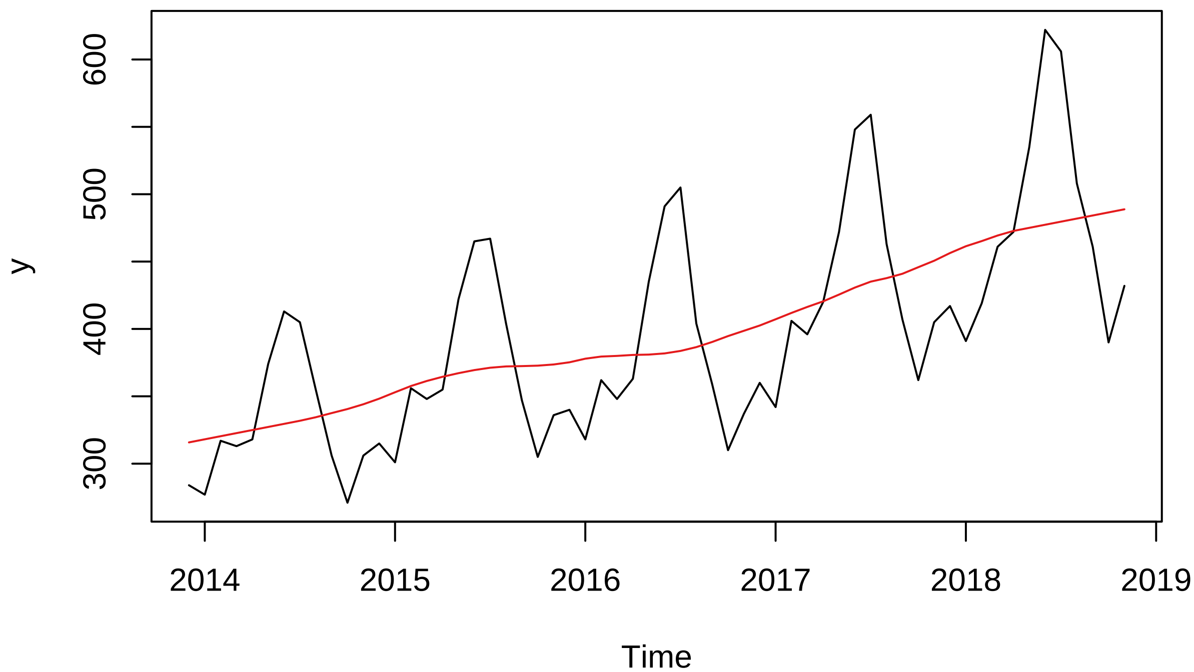
Question: Does your understanding of the plots agree with the underlying model?

Answer: Based on the seasonal plot we can see that the time series is very seasonal since the lines from each year follow exactly the same pattern. Therefore the name Season B fits well for this time series.

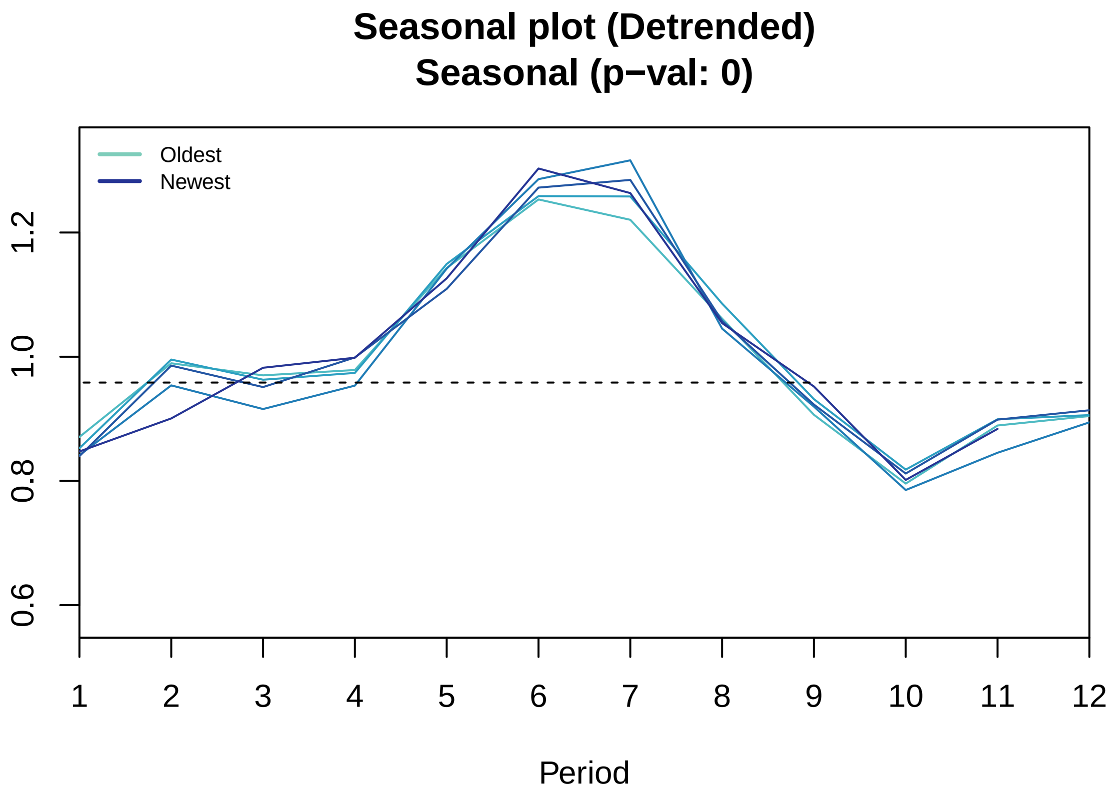
```r
#Using the column TrendSeason
y8 <- ts(Y[,8], frequency = 12, end = c(2018, 11))
plot(y8)
```




```r
cma8 <- cmav(y8, outplot = 1)
```

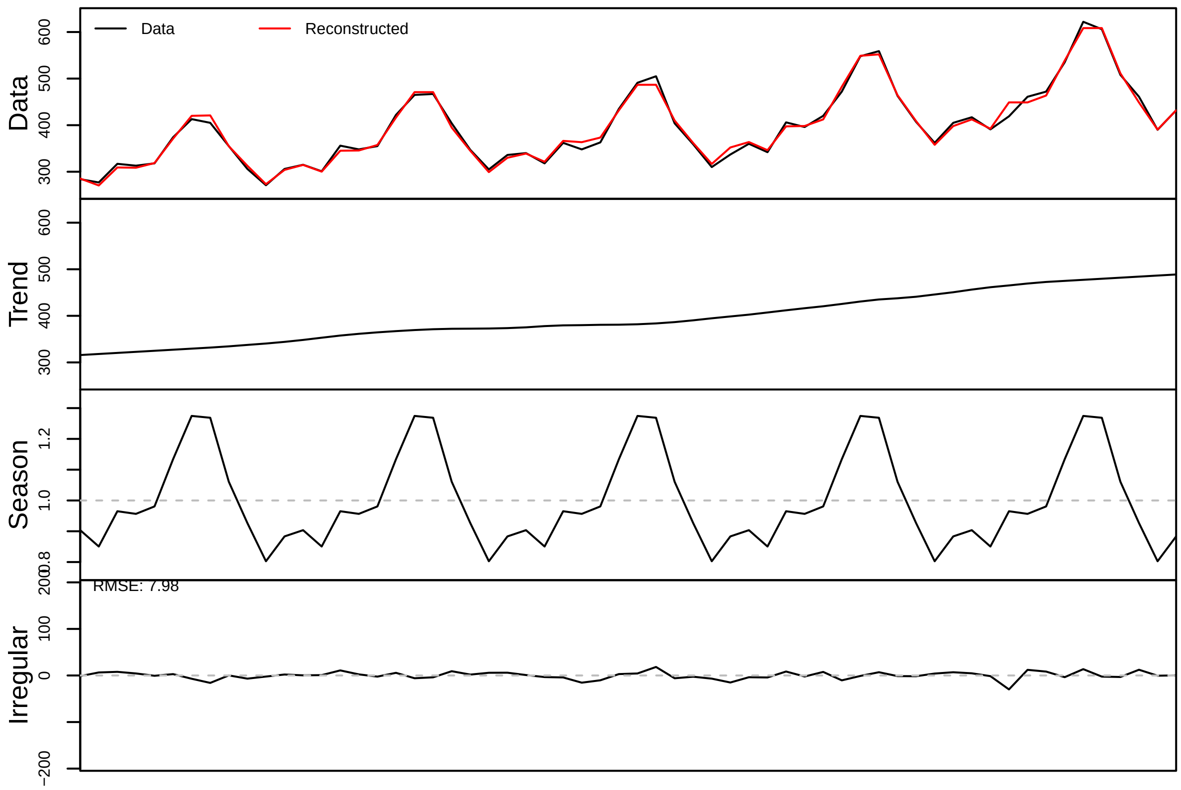



```r
seasplot(y8)
```




```
## Results of statistical testing
## Evidence of trend: TRUE  (pval: 0)
## Evidence of seasonality: TRUE  (pval: 0)
```

```r
dc8 <-decomp(y8 ,outplot=1)
```



Question: Does your understanding of the plots agree with the underlying model?

Answer: The name TrendSeason is very fitting for the time series at hand since a clear trend and a clear seasonality can be seen in the plots. The CMA plot shows a clear red line that is additive and therefore signalizes the positive trend of the time series. The seasonal plot shows that the lines from each year follow exactly the same pattern, indicating strong seasonality.

2. Does decomp() know when to remove the trend or the seasonality?

decomp() does not automatically know when to remove the trend or seasonality. The function seems to assume that a trend & seasonality exist in every time series. Even if there is no seasonality or trend the function makes it up and performs a classical decomposition based on the specified parameters. Therefore the answer to the question is no. decomp() does not know when to remove the trend or the seasonality.

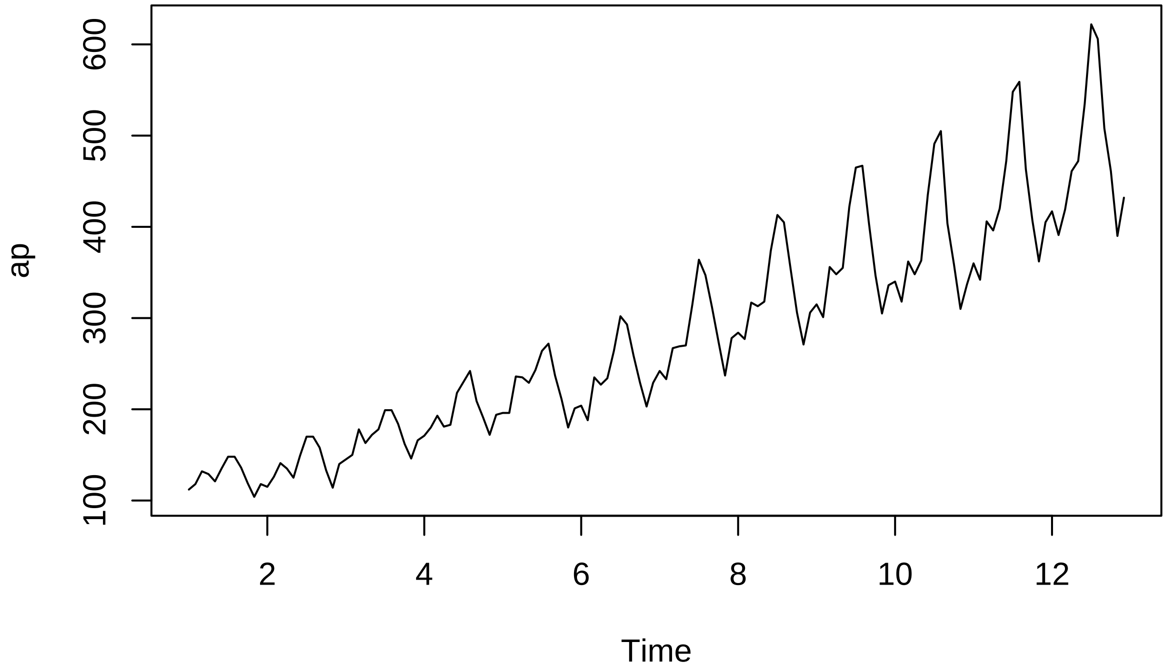
3. AirPassengers time series

```r
ap <- AirPassengers
print(ap)
```

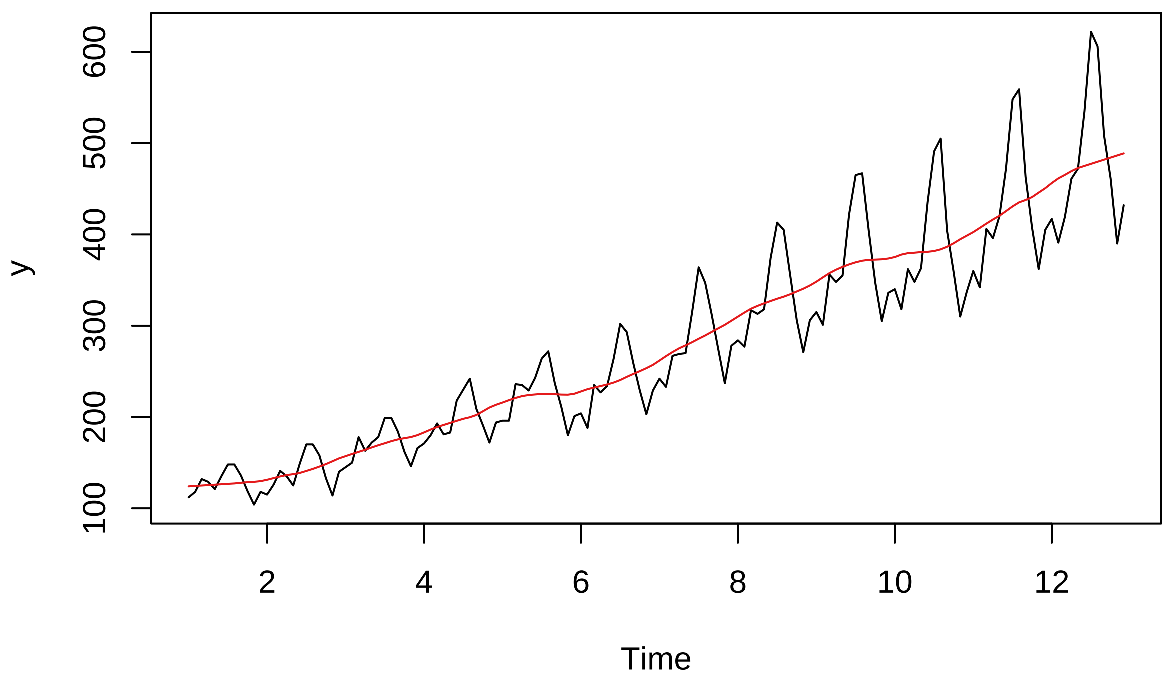
```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 1949 112 118 132 129 121 135 148 148 136 119 104 118
## 1950 115 126 141 135 125 149 170 170 158 133 114 140
## 1951 145 150 178 163 172 178 199 199 184 162 146 166
## 1952 171 180 193 181 183 218 230 242 209 191 172 194
## 1953 196 196 236 235 229 243 264 272 237 211 180 201
## 1954 204 188 235 227 234 264 302 293 259 229 203 229
## 1955 242 233 267 269 270 315 364 347 312 274 237 278
## 1956 284 277 317 313 318 374 413 405 355 306 271 306
## 1957 315 301 356 348 355 422 465 467 404 347 305 336
## 1958 340 318 362 348 363 435 491 505 404 359 310 337
```

```
## 1959 360 342 406 396 420 472 548 559 463 407 362 405
## 1960 417 391 419 461 472 535 622 606 508 461 390 432
```

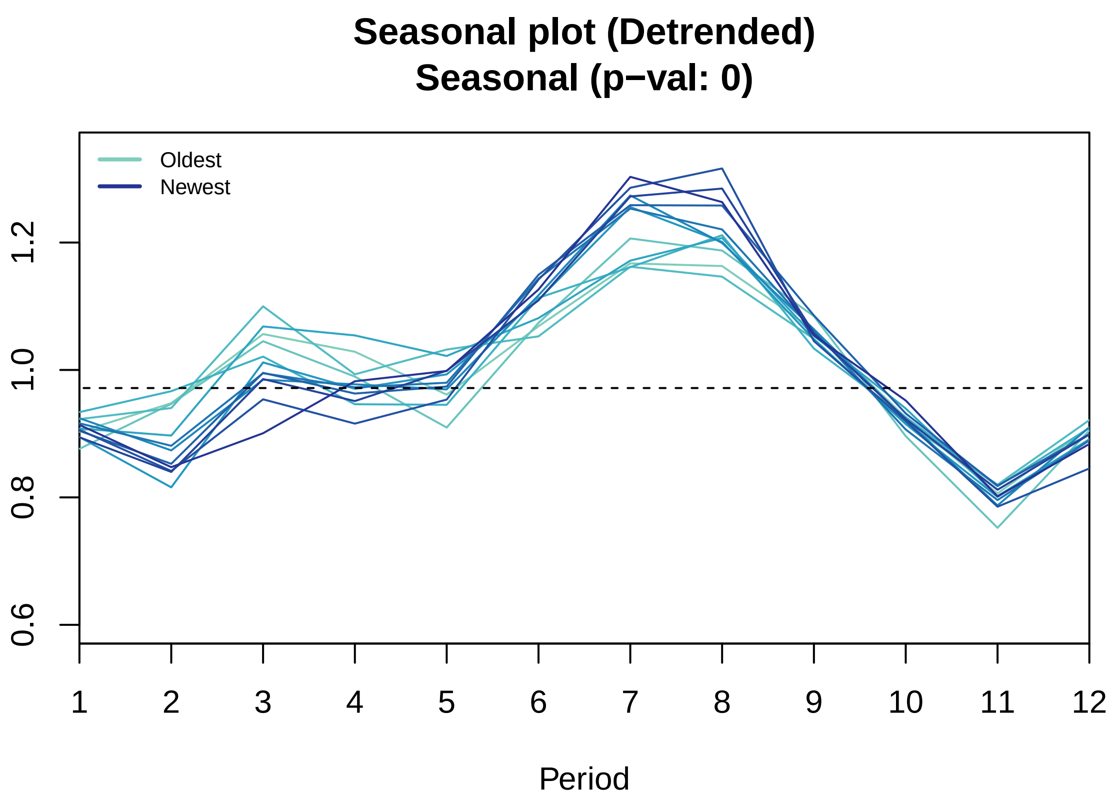
```r
ap <- ts(ap, frequency = 12)
plot(ap)
```




```r
cmaap <- cmav(ap, outplot = 1)
```

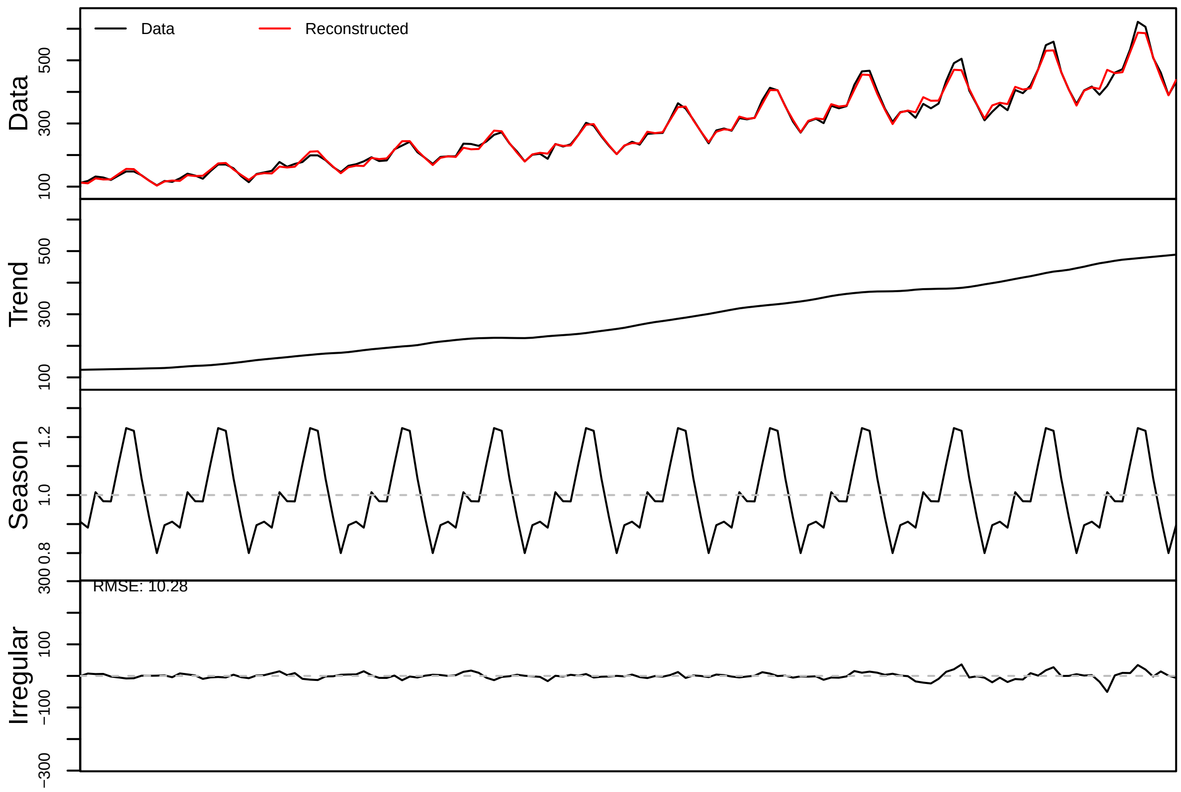



```r
seasplot(ap)
```




```
## Results of statistical testing
## Evidence of trend: TRUE  (pval: 0)
## Evidence of seasonality: TRUE  (pval: 0)
```

```
dcap <-decomp(ap ,outplot=1)
```



The time series has an additive trend with multiplicative seasonality. This can be observed on the Centered Moving Average (red line) which constantly rising linearly which suggests an additive trend and on the seasonal plot which follows the same pattern every year which indicates a seasonality.