Project Week 1

Ananda Putra Wijaya/G1401221111

2024-08-24

Library

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.3.3

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

```
library(graphics)
library(TTR)
```

```
## Warning: package 'TTR' was built under R version 4.3.3
```

```
library(TSA)
```

```
## Warning: package 'TSA' was built under R version 4.3.3

## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast

##
## Attaching package: 'TSA'

## The following objects are masked from 'package:stats':
##
##     acf, arima

## The following object is masked from 'package:utils':
##
##     tar
```

```
library(rio)
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

Import Data

```
dataa<-read.csv("C:/Users/nndap/OneDrive/Desktop/Tugas & Misc/Semester 5/Metode Peramalan Deret Waktu/Da
dataa
```

```
##     Periode Jumlah   Tanggal
```

```
## 1           1  26143 18/5/2024
## 2           2  23206 19/5/2024
## 3           3  22881 20/5/2024
## 4           4  25195 21/5/2024
## 5           5  32969 22/5/2024
## 6           6  43737 23/5/2024
## 7           7  26978 24/5/2024
## 8           8  26481 25/5/2024
## 9           9  23576 26/5/2024
## 10         10  21380 27/5/2024
## 11         11  24178 28/5/2024
## 12         12  24707 29/5/2024
## 13         13  24342 30/5/2024
## 14         14  27480 31/5/2024
## 15         15  16959  1/6/2024
## 16         16  17602  2/6/2024
## 17         17  22629  3/6/2024
## 18         18  23796  4/6/2024
## 19         19  23278  5/6/2024
## 20         20  26563  6/6/2024
## 21         21  29581  7/6/2024
## 22         22  30305  8/6/2024
## 23         23  25772  9/6/2024
## 24         24  25006 10/6/2024
## 25         25  26430 11/6/2024
## 26         26  27759 12/6/2024
## 27         27  29510 13/6/2024
## 28         28  43121 14/6/2024
## 29         29  63483 15/6/2024
## 30         30  37762 16/6/2024
## 31         31  21135 17/6/2024
## 32         32  27153 18/6/2024
## 33         33  31129 19/6/2024
## 34         34  31386 20/6/2024
## 35         35   4867 21/6/2024
## 36         36  40311 22/6/2024
## 37         37  33329 23/6/2024
## 38         38  30072 24/6/2024
## 39         39  31216 25/6/2024
## 40         40   4558 26/6/2024
## 41         41  34288 27/6/2024
## 42         42  38859 28/6/2024
## 43         43  43320 29/6/2024
## 44         44  36202 30/6/2024
## 45         45  32508  1/7/2024
## 46         46  27965  2/7/2024
## 47         47  21681  3/7/2024
## 48         48  34025  4/7/2024
## 49         49  38681  5/7/2024
## 50         50   5900  6/7/2024
## 51         51  33262  7/7/2024
## 52         52  29314  8/7/2024
## 53         53  29518  9/7/2024
## 54         54  29897 10/7/2024
```

```
## 55        55   5755 11/7/2024
## 56        56  33585 12/7/2024
## 57        57  35318 13/7/2024
## 58        58  26535 14/7/2024
## 59        59  24708 15/7/2024
## 60        60   3529 16/7/2024
## 61        61  27050 17/7/2024
## 62        62  26904 18/7/2024
## 63        63  30677 19/7/2024
## 64        64  29970 20/7/2024
## 65        65   3279 21/7/2024
## 66        66  22947 22/7/2024
## 67        67  23842 23/7/2024
## 68        68  24667 24/7/2024
## 69        69  25955 25/7/2024
## 70        70  28964 26/7/2024
## 71        71  30509 27/7/2024
## 72        72  21687 28/7/2024
## 73        73  23289 29/7/2024
## 74        74  23558 30/7/2024
## 75        75  24060 31/7/2024
## 76        76  25704  1/8/2024
## 77        77  28110  2/8/2024
## 78        78  15982  3/8/2024
## 79        79  23174  4/8/2024
## 80        80  20570  5/8/2024
## 81        81  23389  6/8/2024
## 82        82  25781  7/8/2024
## 83        83  25265  8/8/2024
## 84        84  28964  9/8/2024
## 85        85  10770 10/8/2024
## 86        86  23207 11/8/2024
## 87        87  23411 12/8/2024
## 88        88  21038 13/8/2024
## 89        89  20990 14/8/2024
## 90        90  13298 15/8/2024
## 91        91  30842 16/8/2024
## 92        92  24253 17/8/2024
## 93        93  21532 18/8/2024
## 94        94  11211 19/8/2024
## 95        95  18064 20/8/2024
## 96        96  11208 21/8/2024
## 97        97  25709 22/8/2024
## 98        98  29339 23/8/2024
## 99        99  26043 24/8/2024
## 100      100  23868 25/8/2024
```

```r
View(dataa)
str(dataa)
```

```
## 'data.frame':    100 obs. of  3 variables:
##  $ Periode: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Jumlah : int  26143 23206 22881 25195 32969 43737 26978 26481 23576 21380 ...
##  $ Tanggal: chr  "18/5/2024" "19/5/2024" "20/5/2024" "21/5/2024" ...
```

```
dim(dataa)
```

```
## [1] 100   3
```

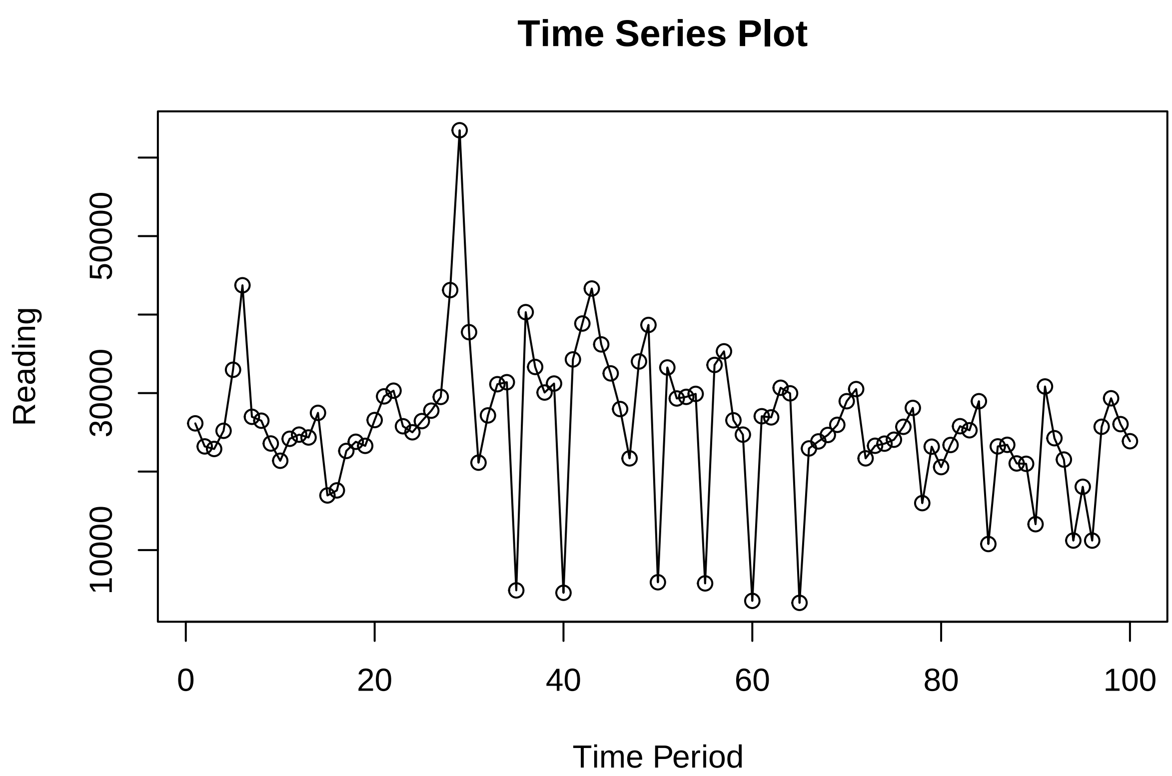
Mengubah data menjadi data deret waktu

```
datats<-ts(dataa$Jumlah)
summary(datats)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3279   23117   25777   25819   29996   63483
```

Plot awal data

```
ts.plot(datats, xlab="Time Period ", ylab="Reading",
        main = "Time Series Plot")
points(datats)
```



Single Moving Average & Double Moving Average

Membagi data latih dan data uji

```
train<-dataa[1:80,]
uji<-dataa[81:100,]
```

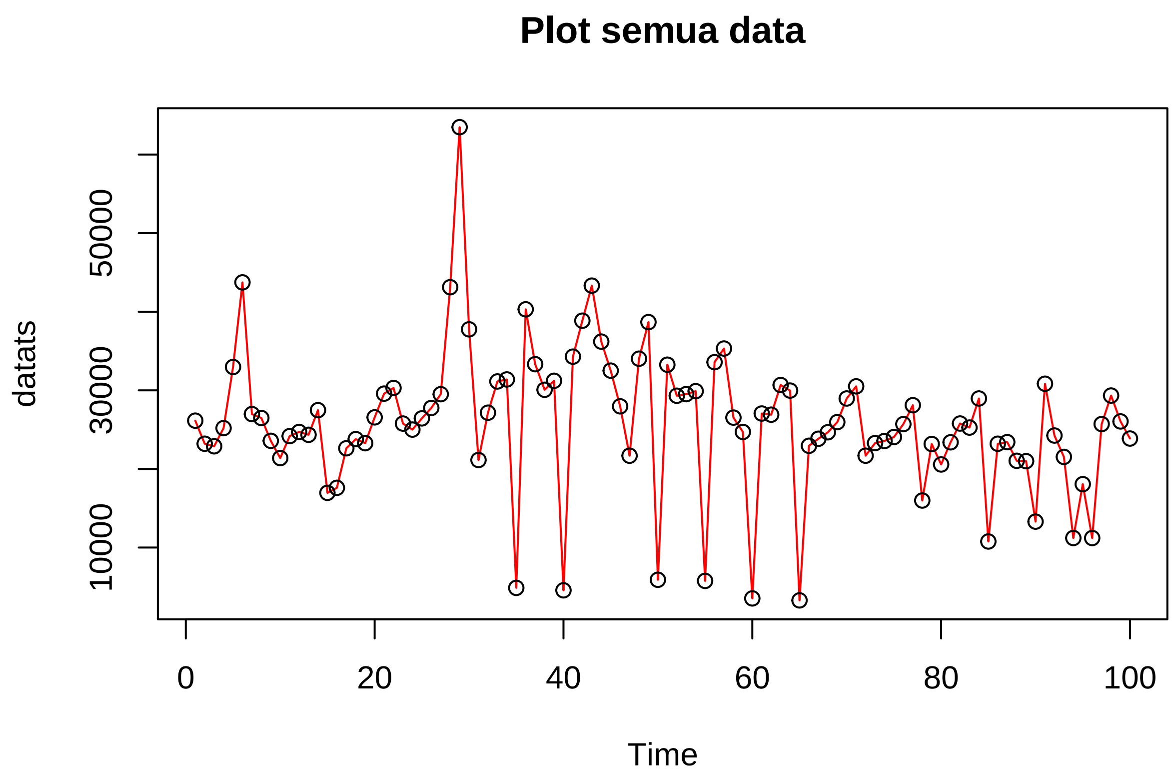
```
traints1<-ts(train$Jumlah)
testts1<-ts(uji$Jumlah)
```

Plot masing-masing data

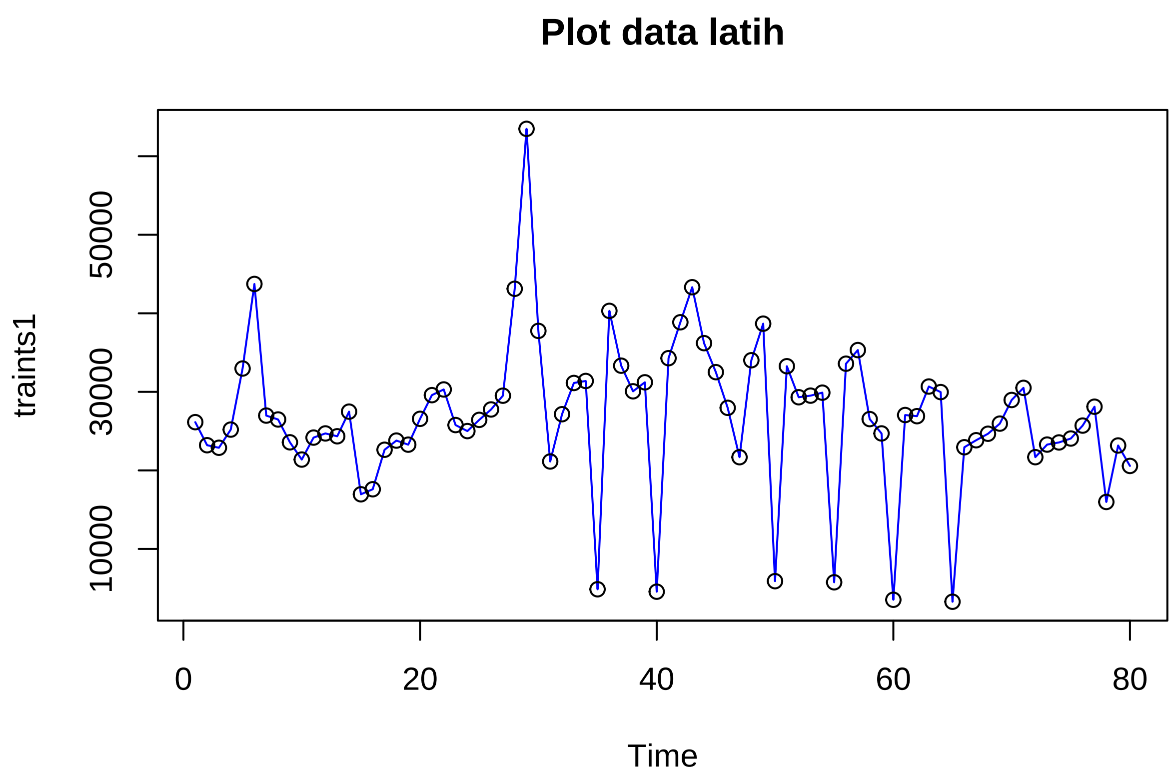
```
plot#eksplorasi keseluruhan data
```

```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x000002a67abc0dd0>
## <environment: namespace:base>
```

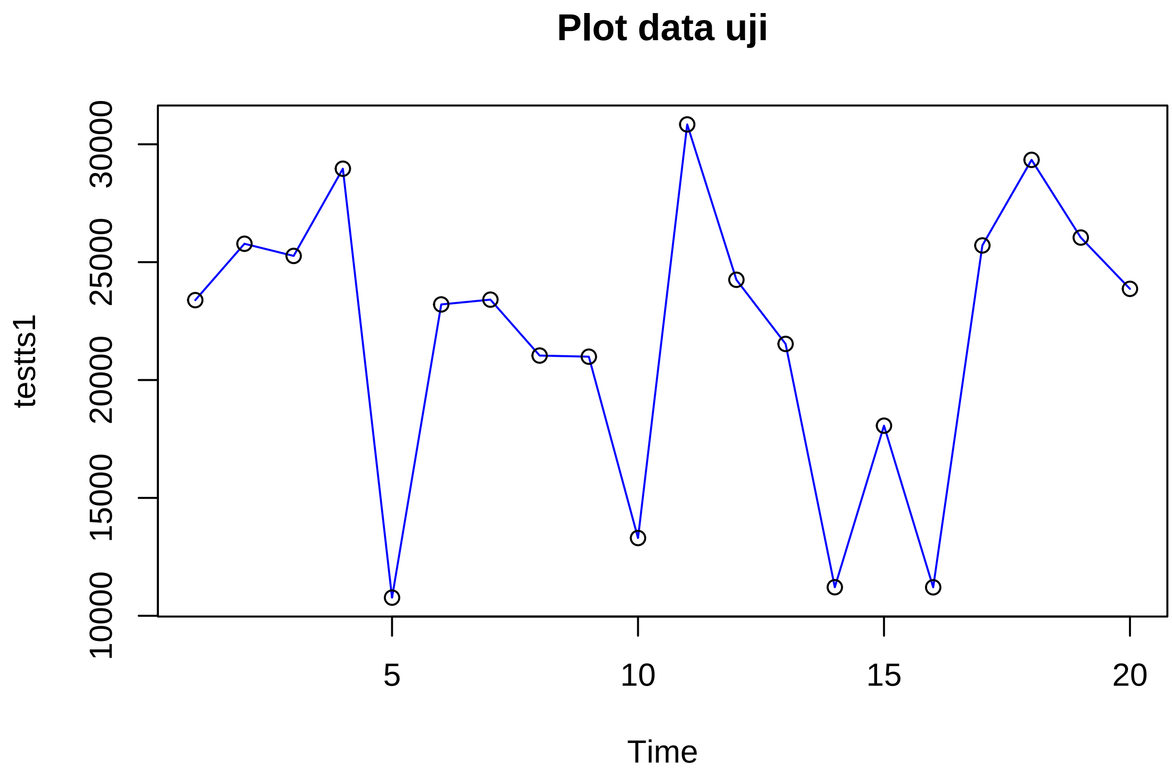
```
plot(datats, col="red",main="Plot semua data")
points(datats)
```




```
#eksplorasi data latih
plot(traints1, col="blue",main="Plot data latih")
points(traints1)
```

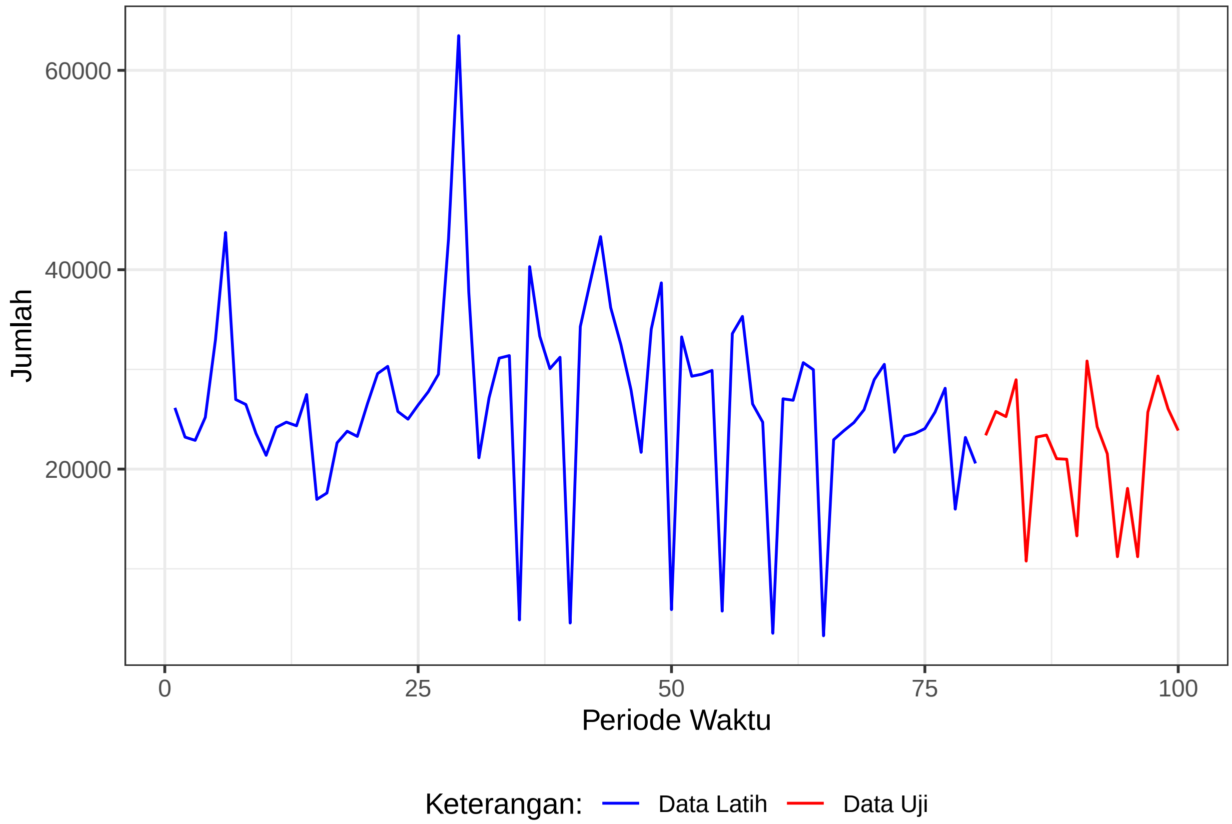



```r
#eksplorasi data uji
plot(testts1, col="blue",main="Plot data uji")
points(testts1)
```



Plot gabungan

```
plot.gabung<-ggplot() +
  geom_line(data = train, aes(x = Periode, y = Jumlah, col = "Data Latih")) +
  geom_line(data = uji, aes(x = Periode, y = Jumlah, col = "Data Uji")) +
  labs(x = "Periode Waktu", y = "Jumlah", color = "Legend") +
  scale_colour_manual(name="Keterangan:", breaks = c("Data Latih", "Data Uji"),
                      values = c("blue", "red")) +
  theme_bw() + theme(legend.position = "bottom",
                     plot.caption = element_text(hjust=0.5, size=12))
plot.gabung
```



Single Moving Average (SMA)

```r
dataSMA<-SMA(traints1,n=4)
dataSMA
```

```
## Time Series:
## Start = 1
## End = 80
## Frequency = 1
##  [1]       NA       NA       NA 24356.25 26062.75 31195.50 32219.75 32541.25
##  [9] 30193.00 24603.75 23903.75 23460.25 23651.75 25176.75 23372.00 21595.75
## [17] 21167.50 20246.50 21826.25 24066.50 25804.50 27431.75 28055.25 27666.00
## [25] 26878.25 26241.75 27176.25 31705.00 40968.25 43469.00 41375.25 37383.25
## [33] 29294.75 27700.75 23633.75 26923.25 27473.25 27144.75 33732.00 24793.75
## [41] 25033.50 27230.25 30256.25 38167.25 37722.25 34998.75 29589.00 29044.75
## [49] 30588.00 25071.75 27967.00 26789.25 24498.50 30497.75 23621.00 24688.75
## [57] 26138.75 25298.25 30036.50 22522.50 20455.50 20547.75 22040.00 28650.25
## [65] 22707.50 21718.25 20009.50 18683.75 24352.75 25857.00 27523.75 26778.75
## [73] 26112.25 24760.75 23148.50 24152.75 25358.00 23464.00 23242.50 21959.00
```

```r
ramalSMA<-c(NA,dataSMA)
ramalSMA
```

Data ramal 1 periode SMA

```
##  [1]       NA       NA       NA       NA 24356.25 26062.75 31195.50 32219.75
```

```
##  [9] 32541.25 30193.00 24603.75 23903.75 23460.25 23651.75 25176.75 23372.00
## [17] 21595.75 21167.50 20246.50 21826.25 24066.50 25804.50 27431.75 28055.25
## [25] 27666.00 26878.25 26241.75 27176.25 31705.00 40968.25 43469.00 41375.25
## [33] 37383.25 29294.75 27700.75 23633.75 26923.25 27473.25 27144.75 33732.00
## [41] 24793.75 25033.50 27230.25 30256.25 38167.25 37722.25 34998.75 29589.00
## [49] 29044.75 30588.00 25071.75 27967.00 26789.25 24498.50 30497.75 23621.00
## [57] 24688.75 26138.75 25298.25 30036.50 22522.50 20455.50 20547.75 22040.00
## [65] 28650.25 22707.50 21718.25 20009.50 18683.75 24352.75 25857.00 27523.75
## [73] 26778.75 26112.25 24760.75 23148.50 24152.75 25358.00 23464.00 23242.50
## [81] 21959.00
```

```
data.gab<-cbind(aktual=c(traints1,rep(NA,20)),pemulusan=c(dataSMA,rep(NA,20)),ramalan=c(ramalSMA,rep(ra
data.gab
```

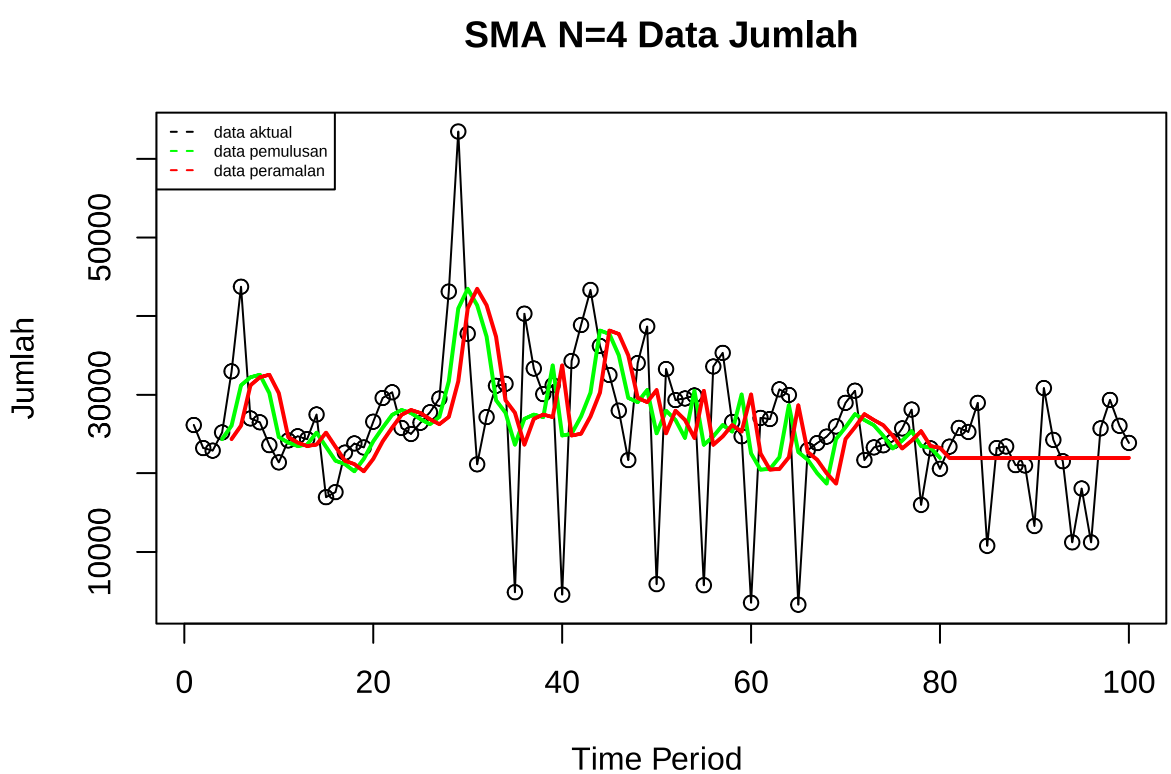
Data ramal 20 periode(sesuai jumlah data uji) SMA

```
##        aktual pemulusan  ramalan
##   [1,]  26143        NA       NA
##   [2,]  23206        NA       NA
##   [3,]  22881        NA       NA
##   [4,]  25195  24356.25       NA
##   [5,]  32969  26062.75 24356.25
##   [6,]  43737  31195.50 26062.75
##   [7,]  26978  32219.75 31195.50
##   [8,]  26481  32541.25 32219.75
##   [9,]  23576  30193.00 32541.25
##  [10,]  21380  24603.75 30193.00
##  [11,]  24178  23903.75 24603.75
##  [12,]  24707  23460.25 23903.75
##  [13,]  24342  23651.75 23460.25
##  [14,]  27480  25176.75 23651.75
##  [15,]  16959  23372.00 25176.75
##  [16,]  17602  21595.75 23372.00
##  [17,]  22629  21167.50 21595.75
##  [18,]  23796  20246.50 21167.50
##  [19,]  23278  21826.25 20246.50
##  [20,]  26563  24066.50 21826.25
##  [21,]  29581  25804.50 24066.50
##  [22,]  30305  27431.75 25804.50
##  [23,]  25772  28055.25 27431.75
##  [24,]  25006  27666.00 28055.25
##  [25,]  26430  26878.25 27666.00
##  [26,]  27759  26241.75 26878.25
##  [27,]  29510  27176.25 26241.75
##  [28,]  43121  31705.00 27176.25
##  [29,]  63483  40968.25 31705.00
##  [30,]  37762  43469.00 40968.25
##  [31,]  21135  41375.25 43469.00
##  [32,]  27153  37383.25 41375.25
##  [33,]  31129  29294.75 37383.25
##  [34,]  31386  27700.75 29294.75
##  [35,]   4867  23633.75 27700.75
##  [36,]  40311  26923.25 23633.75
##  [37,]  33329  27473.25 26923.25
```

```
## [38,] 30072 27144.75 27473.25
## [39,] 31216 33732.00 27144.75
## [40,]  4558 24793.75 33732.00
## [41,] 34288 25033.50 24793.75
## [42,] 38859 27230.25 25033.50
## [43,] 43320 30256.25 27230.25
## [44,] 36202 38167.25 30256.25
## [45,] 32508 37722.25 38167.25
## [46,] 27965 34998.75 37722.25
## [47,] 21681 29589.00 34998.75
## [48,] 34025 29044.75 29589.00
## [49,] 38681 30588.00 29044.75
## [50,]  5900 25071.75 30588.00
## [51,] 33262 27967.00 25071.75
## [52,] 29314 26789.25 27967.00
## [53,] 29518 24498.50 26789.25
## [54,] 29897 30497.75 24498.50
## [55,]  5755 23621.00 30497.75
## [56,] 33585 24688.75 23621.00
## [57,] 35318 26138.75 24688.75
## [58,] 26535 25298.25 26138.75
## [59,] 24708 30036.50 25298.25
## [60,]  3529 22522.50 30036.50
## [61,] 27050 20455.50 22522.50
## [62,] 26904 20547.75 20455.50
## [63,] 30677 22040.00 20547.75
## [64,] 29970 28650.25 22040.00
## [65,]  3279 22707.50 28650.25
## [66,] 22947 21718.25 22707.50
## [67,] 23842 20009.50 21718.25
## [68,] 24667 18683.75 20009.50
## [69,] 25955 24352.75 18683.75
## [70,] 28964 25857.00 24352.75
## [71,] 30509 27523.75 25857.00
## [72,] 21687 26778.75 27523.75
## [73,] 23289 26112.25 26778.75
## [74,] 23558 24760.75 26112.25
## [75,] 24060 23148.50 24760.75
## [76,] 25704 24152.75 23148.50
## [77,] 28110 25358.00 24152.75
## [78,] 15982 23464.00 25358.00
## [79,] 23174 23242.50 23464.00
## [80,] 20570 21959.00 23242.50
## [81,]    NA       NA 21959.00
## [82,]    NA       NA 21959.00
## [83,]    NA       NA 21959.00
## [84,]    NA       NA 21959.00
## [85,]    NA       NA 21959.00
## [86,]    NA       NA 21959.00
## [87,]    NA       NA 21959.00
## [88,]    NA       NA 21959.00
## [89,]    NA       NA 21959.00
## [90,]    NA       NA 21959.00
## [91,]    NA       NA 21959.00
```

```
##  [92,]      NA        NA 21959.00
##  [93,]      NA        NA 21959.00
##  [94,]      NA        NA 21959.00
##  [95,]      NA        NA 21959.00
##  [96,]      NA        NA 21959.00
##  [97,]      NA        NA 21959.00
##  [98,]      NA        NA 21959.00
##  [99,]      NA        NA 21959.00
## [100,]      NA        NA 21959.00
```

```
ts.plot(datats, xlab="Time Period ", ylab="Jumlah", main= "SMA N=4 Data Jumlah")
points(datats)
lines(data.gab[,2],col="green",lwd=2)
lines(data.gab[,3],col="red",lwd=2)
legend("topleft",c("data aktual","data pemulusan","data peramalan"), lty=8, col=c("black","green","red"
```



Plot ramal SMA

```
error_train.sma = traints1-ramalSMA[1:length(traints1)]
SSE_train.sma = sum(error_train.sma[5:length(traints1)]^2)
MSE_train.sma = mean(error_train.sma[5:length(traints1)]^2)
MAPE_train.sma = mean(abs((error_train.sma[5:length(traints1)]/traints1[5:length(traints1)])*100))

akurasi_train.sma <- matrix(c(SSE_train.sma, MSE_train.sma, MAPE_train.sma))
row.names(akurasi_train.sma)<- c("SSE", "MSE", "MAPE")
colnames(akurasi_train.sma) <- c("Akurasi m = 4")
```

```
akurasi_train.sma
```

Selanjutnya perhitungan akurasi *data training* dilakukan dengan ukuran akurasi *Sum Squares Error* (SSE), *Mean Square Error* (MSE) dan *Mean Absolute Percentage Error* (MAPE).

```
##      Akurasi m = 4
## SSE   8.960772e+09
## MSE   1.179049e+08
## MAPE  6.551244e+01
```

Nilai MAPE pemulusan SMA pada data latih adalah 65%.

```
error_uji.sma = testts1-data.gab[81:100,3]
SSE_uji.sma = sum(error_uji.sma^2)
MSE_uji.sma = mean(error_uji.sma^2)
MAPE_uji.sma = mean(abs((error_uji.sma/testts1*100)))

akurasi_test.sma <- matrix(c(SSE_uji.sma, MSE_uji.sma, MAPE_uji.sma))
row.names(akurasi_test.sma)<- c("SSE", "MSE", "MAPE")
colnames(akurasi_test.sma) <- c("Akurasi m = 4")
akurasi_test.sma
```

Selanjutnya perhitungan akurasi *data uji* dilakukan dengan ukuran akurasi *Sum Squares Error* (SSE), *Mean Square Error* (MSE) dan *Mean Absolute Percentage Error* (MAPE).

```
##      Akurasi m = 4
## SSE   7.017891e+08
## MSE   3.508946e+07
## MAPE  2.824114e+01
```

Nilai MAPE pemulusan SMA pada data uji adalah 32%.

Double Moving Average

```
dma <- SMA(dataSMA, n = 4)
At <- 2*dataSMA - dma
Bt <- 2/(4-1)*(dataSMA - dma)
dataDMA<- At+Bt
ramalDMA<- c(NA, dataDMA)

t = 1:20
f = c()

for (i in t) {
  f[i] = At[length(At)] + Bt[length(Bt)]*(i)
}

data.gab2 <- cbind(aktual = c(traints1,rep(NA,20)), pemulusan1 = c(dataSMA,rep(NA,20)),pemulusan2 = c(da
data.gab2
```

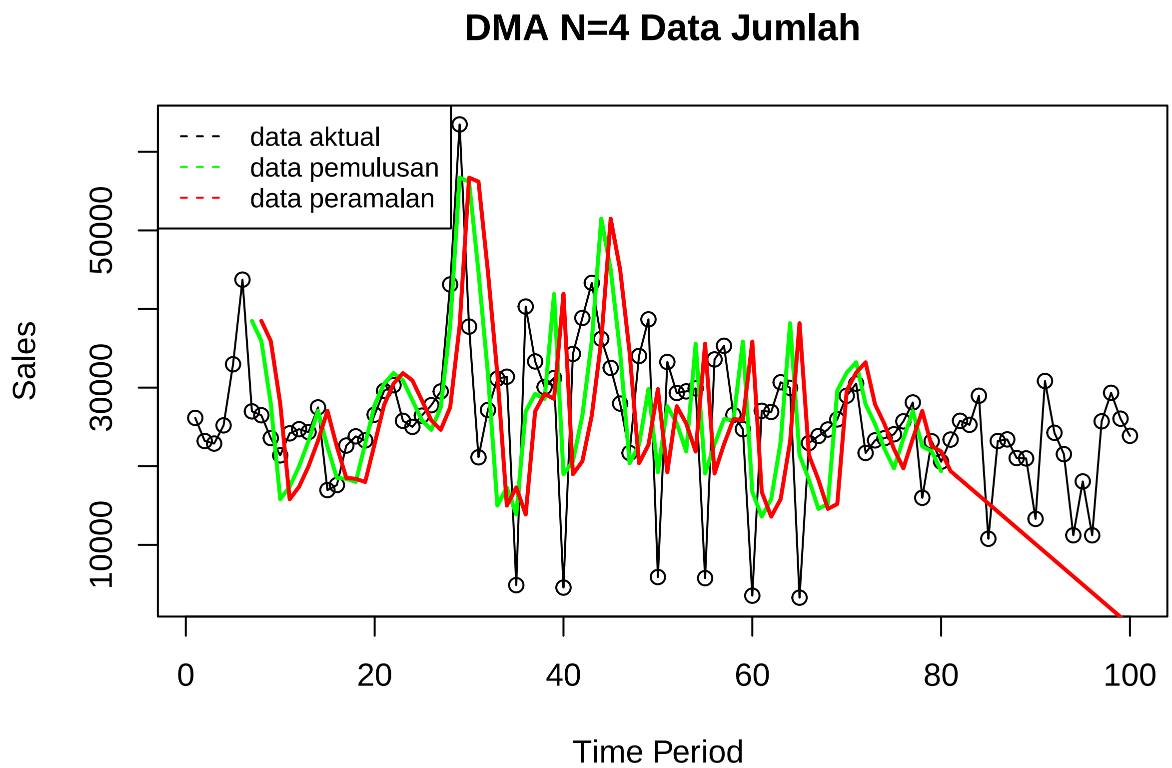
```
##        aktual pemulusan1 pemulusan2       At          Bt   ramalan
##   [1,]  26143         NA         NA       NA          NA        NA
##   [2,]  23206         NA         NA       NA          NA        NA
##   [3,]  22881         NA         NA       NA          NA        NA
##   [4,]  25195   24356.25         NA       NA          NA        NA
```

```
##   [5,] 32969  26062.75        NA       NA          NA        NA
##   [6,] 43737  31195.50        NA       NA          NA        NA
##   [7,] 26978  32219.75  38488.40 35980.94  2507.45833        NA
##   [8,] 26481  32541.25  35935.31 34577.69  1357.62500 38488.396
##   [9,] 23576  30193.00  27952.38 28848.62  -896.25000 35935.312
##  [10,] 21380  24603.75  15794.27 19318.06 -3523.79167 27952.375
##  [11,] 24178  23903.75  17392.60 19997.06 -2604.45833 15794.271
##  [12,] 24707  23460.25  19993.69 21380.31 -1386.62500 17392.604
##  [13,] 24342  23651.75  23229.88 23398.62  -168.75000 19993.688
##  [14,] 27480  25176.75  27057.79 26305.38   752.41667 23229.875
##  [15,] 16959  23372.00  22466.69 22828.81  -362.12500 27057.792
##  [16,] 17602  21595.75  18506.90 19742.44 -1235.54167 22466.688
##  [17,] 22629  21167.50  18400.00 19507.00 -1107.00000 18506.896
##  [18,] 23796  20246.50  17998.27 18897.56  -899.29167 18400.000
##  [19,] 23278  21826.25  22855.00 22443.50   411.50000 17998.271
##  [20,] 26563  24066.50  27799.52 26306.31  1493.20833 22855.000
##  [21,] 29581  25804.50  30502.10 28623.06  1879.04167 27799.521
##  [22,] 30305  27431.75  31847.58 30081.25  1766.33333 30502.104
##  [23,] 25772  28055.25  30914.83 29771.00  1143.83333 31847.583
##  [24,] 25006  27666.00  28377.04 28092.62   284.41667 30914.833
##  [25,] 26430  26878.25  25828.98 26248.69  -419.70833 28377.042
##  [26,] 27759  26241.75  24627.48 25273.19  -645.70833 25828.979
##  [27,] 29510  27176.25  27485.73 27361.94   123.79167 24627.479
##  [28,] 43121  31705.00  37879.48 35409.69  2469.79167 27485.729
##  [29,] 63483  40968.25  56710.65 50413.69  6296.95833 37879.479
##  [30,] 37762  43469.00  56201.29 51108.38  5092.91667 56710.646
##  [31,] 21135  41375.25  44701.71 43371.12  1330.58333 56201.292
##  [32,] 27153  37383.25  31690.44 33967.56 -2277.12500 44701.708
##  [33,] 31129  29294.75  14985.06 20708.94 -5723.87500 31690.438
##  [34,] 31386  27700.75  17304.50 21463.00 -4158.50000 14985.062
##  [35,]  4867  23633.75  13851.46 17764.38 -3912.91667 17304.500
##  [36,] 40311  26923.25  26981.79 26958.38    23.41667 13851.458
##  [37,] 33329  27473.25  29207.42 28513.75   693.66667 26981.792
##  [38,] 30072  27144.75  28563.08 27995.75   567.33333 29207.417
##  [39,] 31216  33732.00  41921.48 38645.69  3275.79167 28563.083
##  [40,]  4558  24793.75  18973.44 21301.56 -2328.12500 41921.479
##  [41,] 34288  25033.50  20629.33 22391.00 -1761.66667 18973.438
##  [42,] 38859  27230.25  26451.71 26763.12  -311.41667 20629.333
##  [43,] 43320  30256.25  35969.27 33684.06  2285.20833 26451.708
##  [44,] 36202  38167.25  51492.98 46162.69  5330.29167 35969.271
##  [45,] 32508  37722.25  45019.33 42100.50  2918.83333 51492.979
##  [46,] 27965  34998.75  34519.79 34711.38  -191.58333 45019.333
##  [47,] 21681  29589.00  20371.81 24058.69 -3686.87500 34519.792
##  [48,] 34025  29044.75  22721.52 25250.81 -2529.29167 20371.812
##  [49,] 38681  30588.00  29809.46 30120.88  -311.41667 22721.521
##  [50,]  5900  25071.75  19235.71 21570.12 -2334.41667 29809.458
##  [51,] 33262  27967.00  27632.21 27766.12  -133.91667 19235.708
##  [52,] 29314  26789.25  25431.33 25974.50  -543.16667 27632.208
##  [53,] 29518  24498.50  21859.96 22915.38 -1055.41667 25431.333
##  [54,] 29897  30497.75  35597.12 33557.38  2039.75000 21859.958
##  [55,]  5755  23621.00  19069.96 20890.38 -1820.41667 35597.125
##  [56,] 33585  24688.75  22792.50 23551.00  -758.50000 19069.958
##  [57,] 35318  26138.75  25975.73 26040.94   -65.20833 22792.500
##  [58,] 26535  25298.25  25900.85 25659.81   241.04167 25975.729
```

```
## [59,] 24708 30036.50 35863.06 33532.44  2330.62500 25900.854
## [60,]  3529 22522.50 16728.33 19046.00 -2317.66667 35863.062
## [61,] 27050 20455.50 13584.35 16332.81 -2748.45833 16728.333
## [62,] 26904 20547.75 15809.73 17704.94 -1895.20833 13584.354
## [63,] 30677 22040.00 23120.94 22688.56   432.37500 15809.729
## [64,] 29970 28650.25 38195.04 34377.12  3817.91667 23120.938
## [65,]  3279 22707.50 21409.38 21928.62  -519.25000 38195.042
## [66,] 22947 21718.25 18283.67 19657.50 -1373.83333 21409.375
## [67,] 23842 20009.50 14573.04 16747.62 -2174.58333 18283.667
## [68,] 24667 18683.75 15190.42 16587.75 -1397.33333 14573.042
## [69,] 25955 24352.75 29622.23 27514.44  2107.79167 15190.417
## [70,] 28964 25857.00 31909.08 29488.25  2420.83333 29622.229
## [71,] 30509 27523.75 33222.81 30943.19  2279.62500 31909.083
## [72,] 21687 26778.75 27863.23 27429.44   433.79167 33222.812
## [73,] 23289 26112.25 25352.77 25656.56  -303.79167 27863.229
## [74,] 23558 24760.75 22205.54 23227.62 -1022.08333 25352.771
## [75,] 24060 23148.50 19729.23 21096.94 -1367.70833 22205.542
## [76,] 25704 24152.75 23501.40 23761.94  -260.54167 19729.229
## [77,] 28110 25358.00 27029.67 26361.00   668.66667 23501.396
## [78,] 15982 23464.00 22519.31 22897.19  -377.87500 27029.667
## [79,] 23174 23242.50 21889.48 22430.69  -541.20833 22519.312
## [80,] 20570 21959.00 19380.88 20412.12 -1031.25000 21889.479
## [81,]    NA       NA       NA       NA          NA 19380.875
## [82,]    NA       NA       NA       NA          NA 18349.625
## [83,]    NA       NA       NA       NA          NA 17318.375
## [84,]    NA       NA       NA       NA          NA 16287.125
## [85,]    NA       NA       NA       NA          NA 15255.875
## [86,]    NA       NA       NA       NA          NA 14224.625
## [87,]    NA       NA       NA       NA          NA 13193.375
## [88,]    NA       NA       NA       NA          NA 12162.125
## [89,]    NA       NA       NA       NA          NA 11130.875
## [90,]    NA       NA       NA       NA          NA 10099.625
## [91,]    NA       NA       NA       NA          NA  9068.375
## [92,]    NA       NA       NA       NA          NA  8037.125
## [93,]    NA       NA       NA       NA          NA  7005.875
## [94,]    NA       NA       NA       NA          NA  5974.625
## [95,]    NA       NA       NA       NA          NA  4943.375
## [96,]    NA       NA       NA       NA          NA  3912.125
## [97,]    NA       NA       NA       NA          NA  2880.875
## [98,]    NA       NA       NA       NA          NA  1849.625
## [99,]    NA       NA       NA       NA          NA   818.375
## [100,]   NA       NA       NA       NA          NA  -212.875
```

Hasil Pemulusan DMA

```r
ts.plot(datats, xlab="Time Period ", ylab="Sales", main= "DMA N=4 Data Jumlah")
points(datats)
lines(data.gab2[,3],col="green",lwd=2)
lines(data.gab2[,6],col="red",lwd=2)
legend("topleft",c("data aktual","data pemulusan","data peramalan"), lty=8, col=c("black","green","red")
```



Perhitungan nilai keakuratan *data latih* DMA

```
error_train.dma = traints1-ramalDMA[1:length(traints1)]
SSE_train.dma = sum(error_train.dma[8:length(traints1)]^2)
MSE_train.dma = mean(error_train.dma[8:length(traints1)]^2)
MAPE_train.dma = mean(abs((error_train.dma[8:length(traints1)]/traints1[8:length(traints1)])*100))

akurasi_train.dma <- matrix(c(SSE_train.dma, MSE_train.dma, MAPE_train.dma))
row.names(akurasi_train.dma)<- c("SSE", "MSE", "MAPE")
colnames(akurasi_train.dma) <- c("Akurasi m = 4")
akurasi_train.dma
```

```
##      Akurasi m = 4
## SSE   1.381834e+10
## MSE   1.892923e+08
## MAPE  8.236944e+01
```

Nilai MAPE keakuratan *data latih* yang didapat sebesar 65%.

Perhitungan nilai keakuratan *data uji* DMA

```
error_test.dma = testts1-data.gab2[81:100,6]
SSE_test.dma = sum(error_test.dma^2)
MSE_test.dma = mean(error_test.dma^2)
MAPE_test.dma = mean(abs((error_test.dma/testts1*100)))

akurasi_test.dma <- matrix(c(SSE_test.dma, MSE_test.dma, MAPE_test.dma))
row.names(akurasi_test.dma)<- c("SSE", "MSE", "MAPE")
colnames(akurasi_test.dma) <- c("Akurasi m = 4")
```

```
akurasi_test.dma
```

```
##      Akurasi m = 4
## SSE   4.380375e+09
## MSE   2.190187e+08
## MAPE  5.639970e+01
```

Nilai MAPE keakuratan *data uji* didapat sebesar 32%.

Pada data Training, metode SMA lebih baik karena nilai MAPE SMA didapat sebesar 65% dibandingkan dengan nilai MAPE DMA yakni 65%. Sedangkan pada data uji, metode DMA sama baiknya karena nilai MAPE yang didapat sebesar 32% sama sepert nilai MAPE SMA yakni 32%

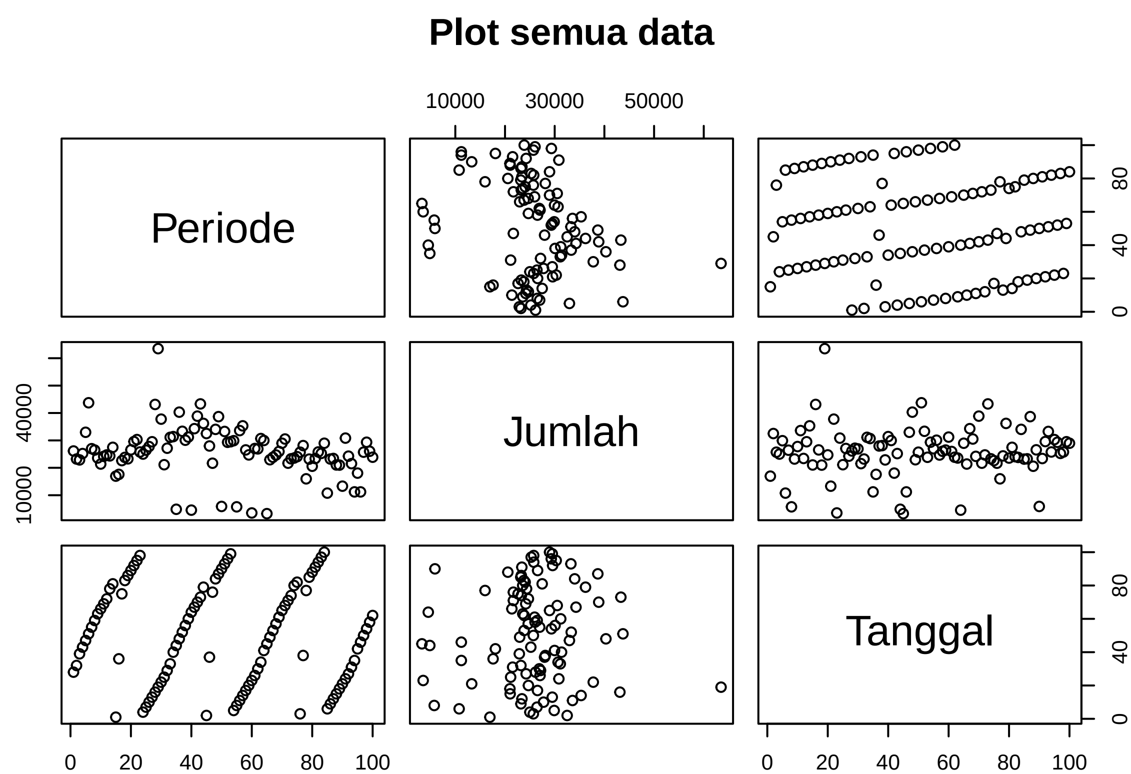
Single Exponential Smoothing & Double Exponential Smoothing

Membagi data latih dan data uji

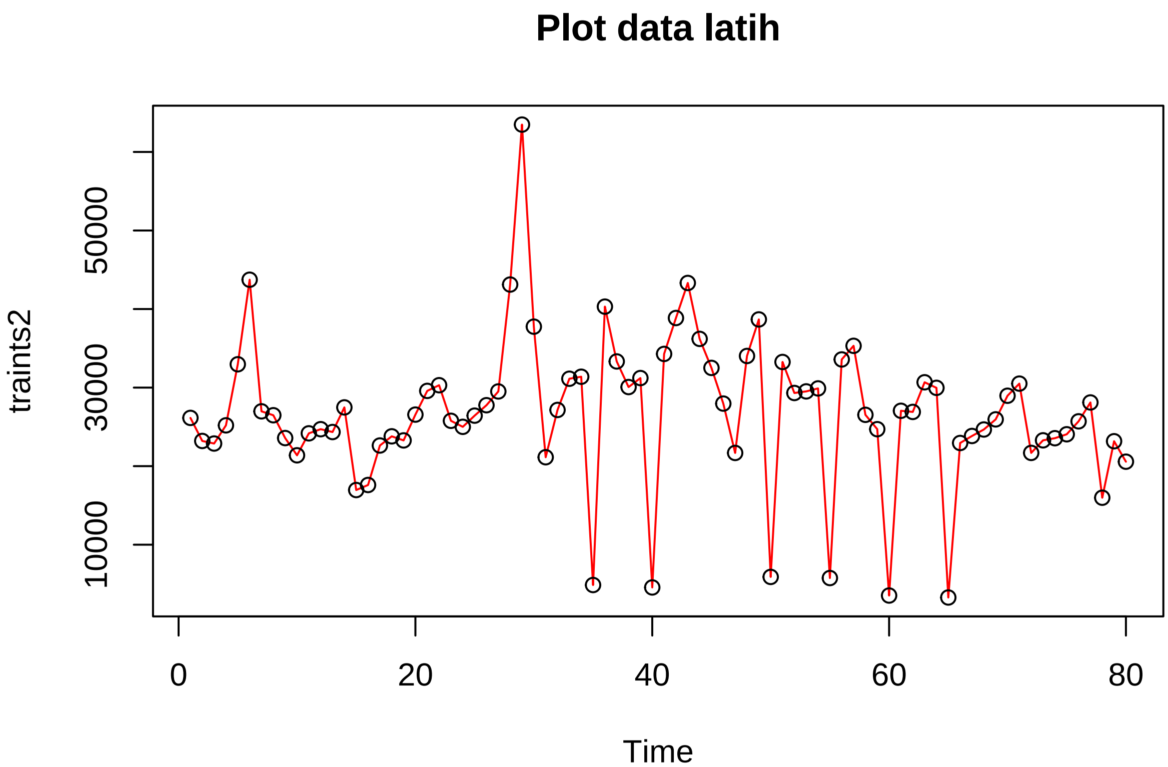
```
train2<-dataa[1:80,]
test2<-dataa[81:100,]
traints2 <- ts(train2$Jumlah)
testts2 <- ts(test2$Jumlah)
```

Plot masing-masing data

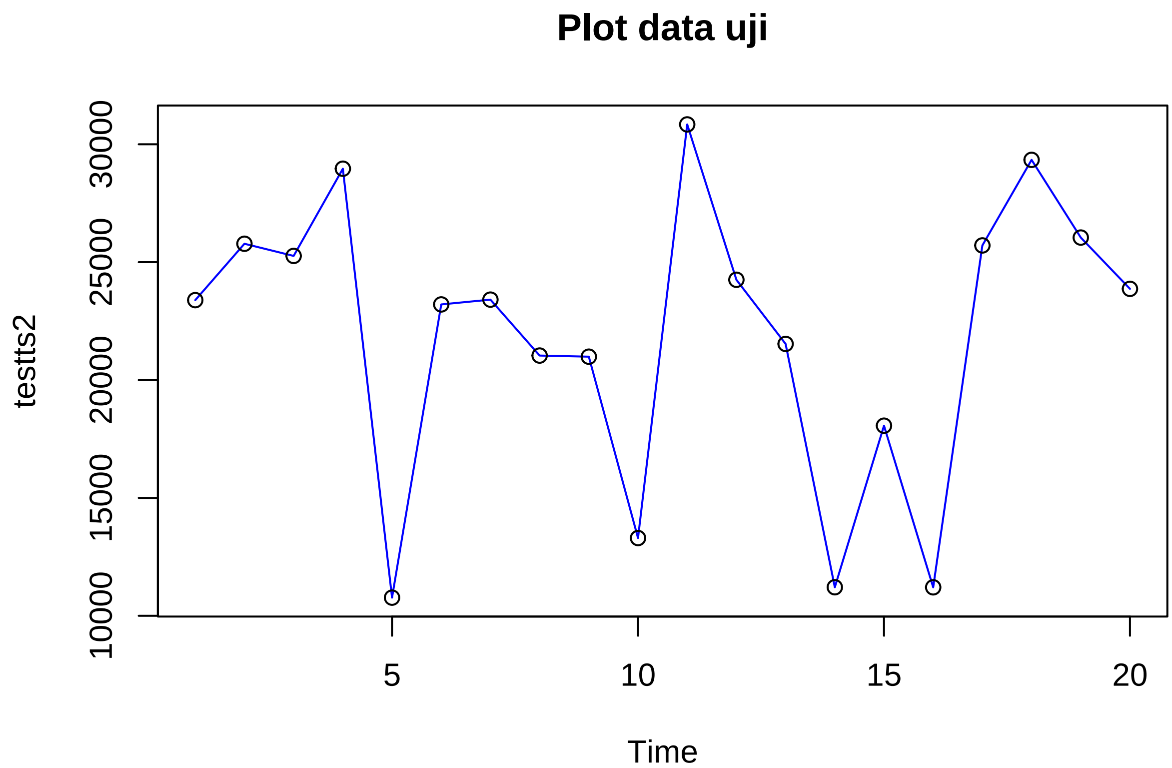
```
plot(dataa, col="black",main="Plot semua data")
points(datats)
```



```r
plot(traints2, col="red",main="Plot data latih")
points(traints2)
```

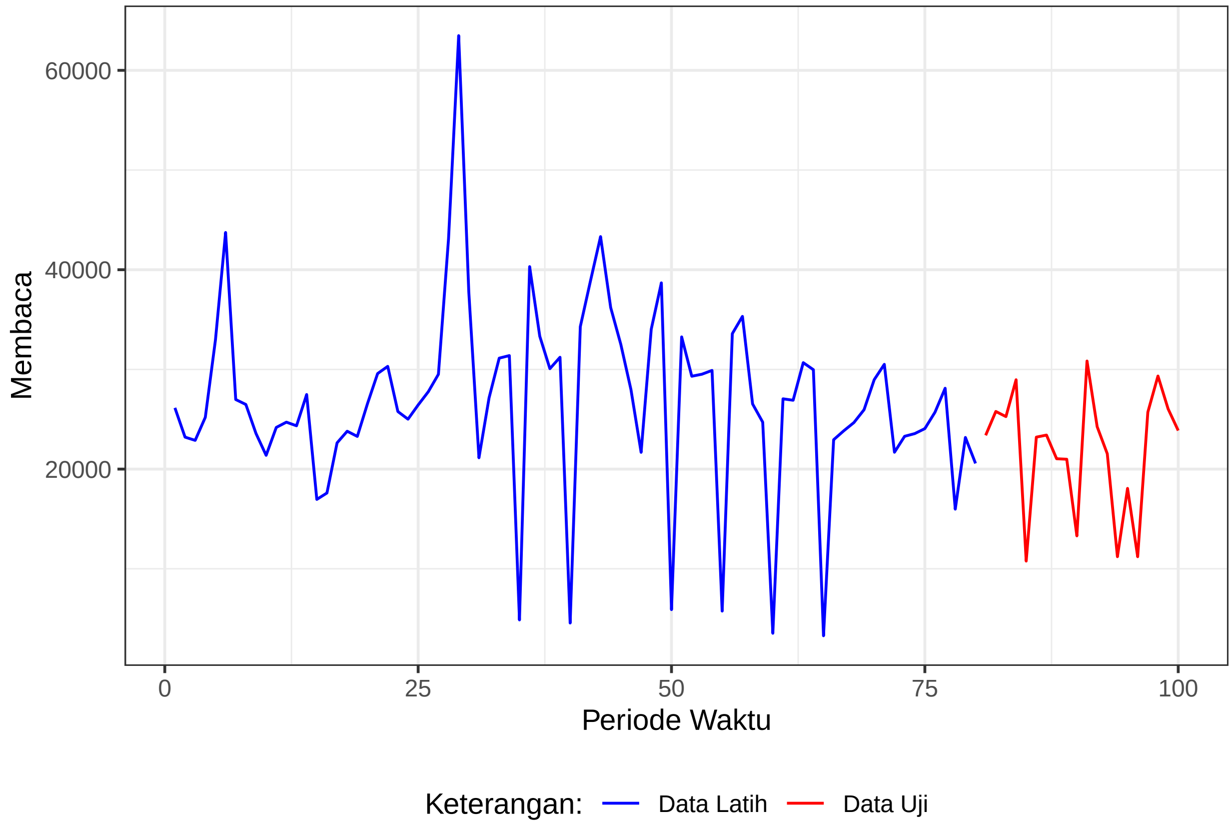



```r
plot(testts2, col="blue",main="Plot data uji")
points(testts2)
```



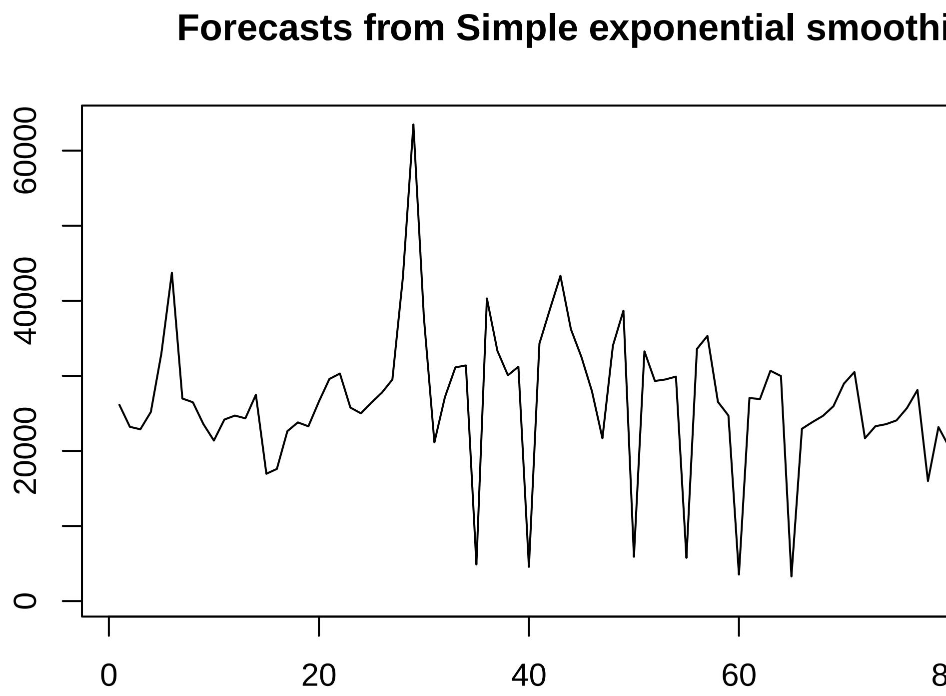
Plot Gabungan

```
ggplot() +
  geom_line(data = train2, aes(x = Periode, y = Jumlah, col = "Data Latih")) +
  geom_line(data = test2, aes(x = Periode, y = Jumlah, col = "Data Uji")) +
  labs(x = "Periode Waktu", y = "Membaca", color = "Legend") +
  scale_colour_manual(name="Keterangan:", breaks = c("Data Latih", "Data Uji"),
                      values = c("blue", "red")) +
  theme_bw() + theme(legend.position = "bottom",
                     plot.caption = element_text(hjust=0.5, size=12))
```



Single Exponential Smoothing

```
#Cara 1 (fungsi ses)
ses.1 <- ses(traints2, h = 10, alpha = 0.2)
plot(ses.1)
```

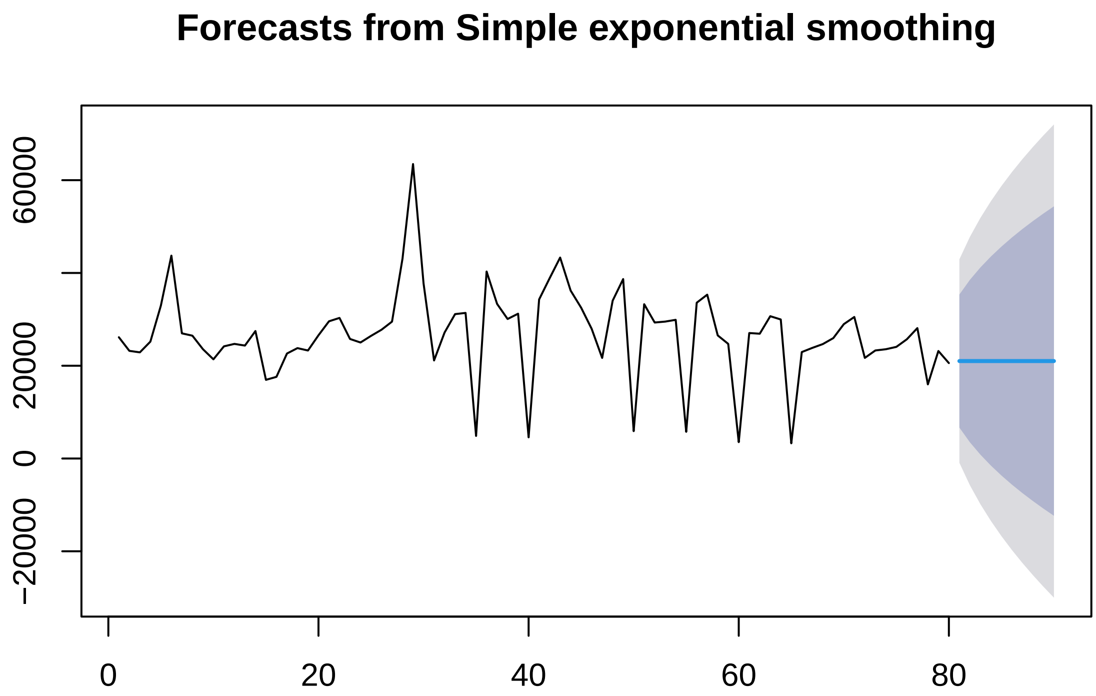


SES menggunakan fungsi ses()

```
ses.1
```

```
##    Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## 81       22820.87 10279.523 35362.22 3640.5328 42001.21
## 82       22820.87 10031.155 35610.59 3260.6872 42381.06
## 83       22820.87  9787.520 35854.23 2888.0788 42753.67
## 84       22820.87  9548.355 36093.39 2522.3089 43119.44
## 85       22820.87  9313.425 36328.32 2163.0144 43478.73
## 86       22820.87  9082.512 36559.23 1809.8630 43831.88
## 87       22820.87  8855.416 36786.33 1462.5501 44179.20
## 88       22820.87  8631.955 37009.79 1120.7952 44520.95
## 89       22820.87  8411.959 37229.79  784.3398 44857.41
## 90       22820.87  8195.271 37446.48  452.9448 45188.80
```

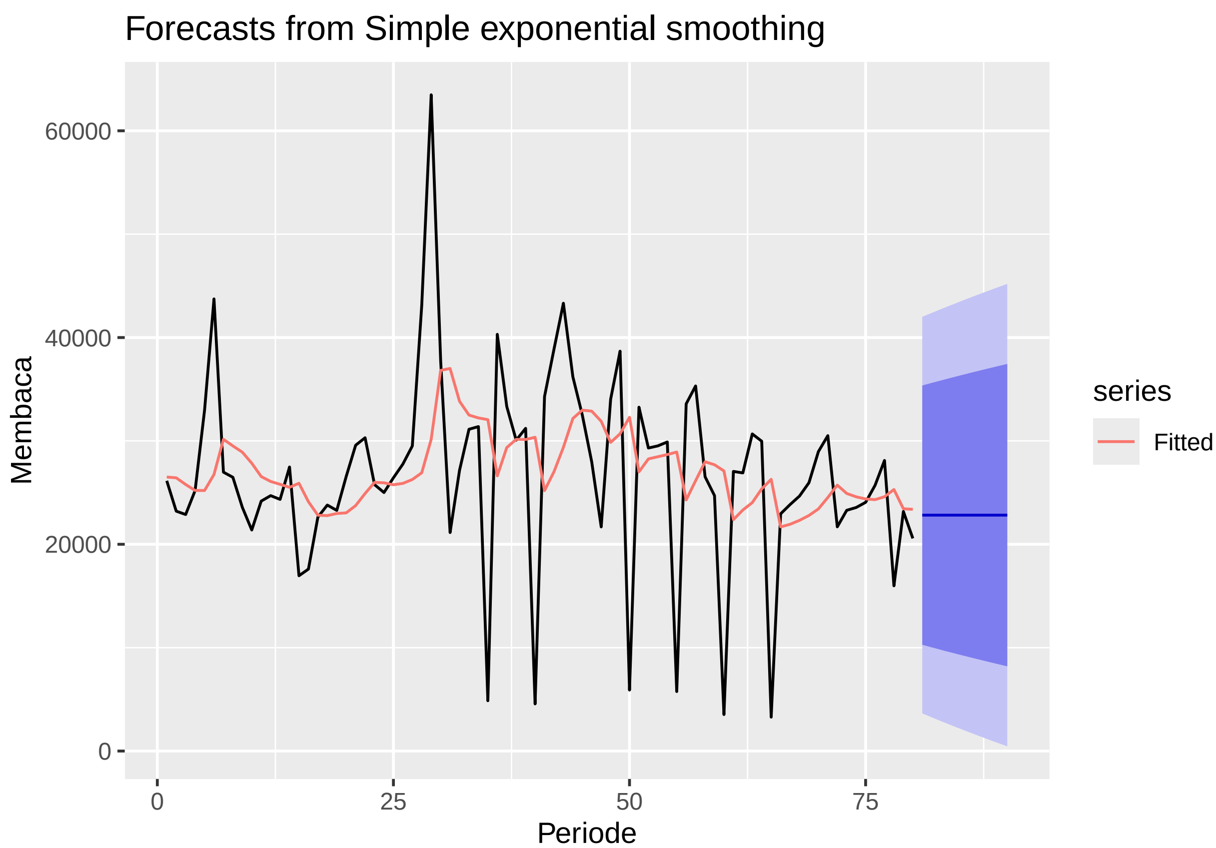
```
ses.2<- ses(traints2, h = 10, alpha = 0.7)
plot(ses.2)
```



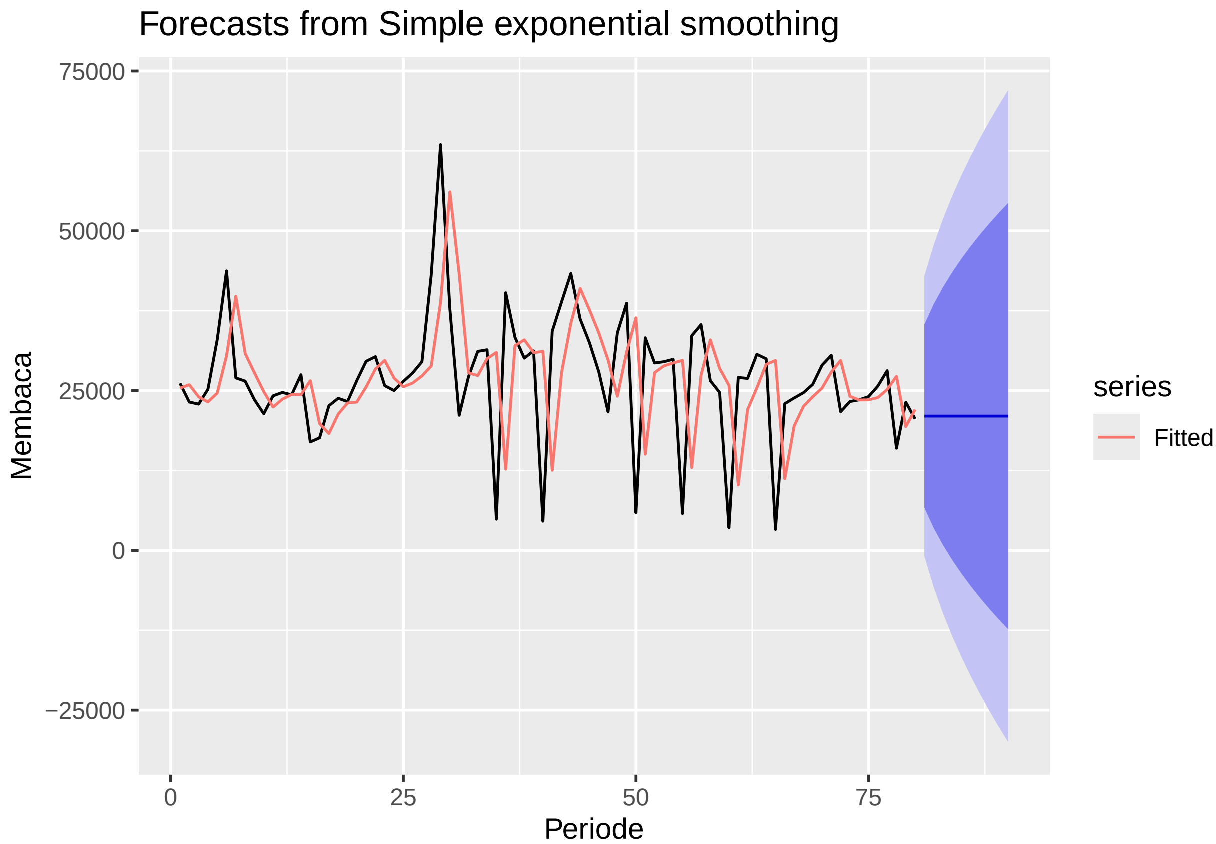

```r
ses.2
```

```
##    Point Forecast        Lo 80    Hi 80        Lo 95    Hi 95
## 81       21007.52   6668.2484 35346.79    -922.5038 42937.55
## 82       21007.52   3504.2082 38510.83   -5761.4857 47776.53
## 83       21007.52    830.3761 41184.67   -9850.7591 51865.80
## 84       21007.52  -1528.4147 43543.46  -13458.2183 55473.26
## 85       21007.52  -3662.6962 45677.74  -16722.3202 58737.36
## 86       21007.52  -5626.4955 47641.54  -19725.6920 61740.73
## 87       21007.52  -7455.1221 49470.16  -22522.3351 64537.38
## 88       21007.52  -9173.1561 51188.20  -25149.8414 67164.88
## 89       21007.52 -10798.5239 52813.57  -27635.6269 69650.67
## 90       21007.52 -12344.7759 54359.82  -30000.4153 72015.46
```

```r
autoplot(ses.1) +
  autolayer(fitted(ses.1), series="Fitted") +
  ylab("Membaca") + xlab("Periode")
```

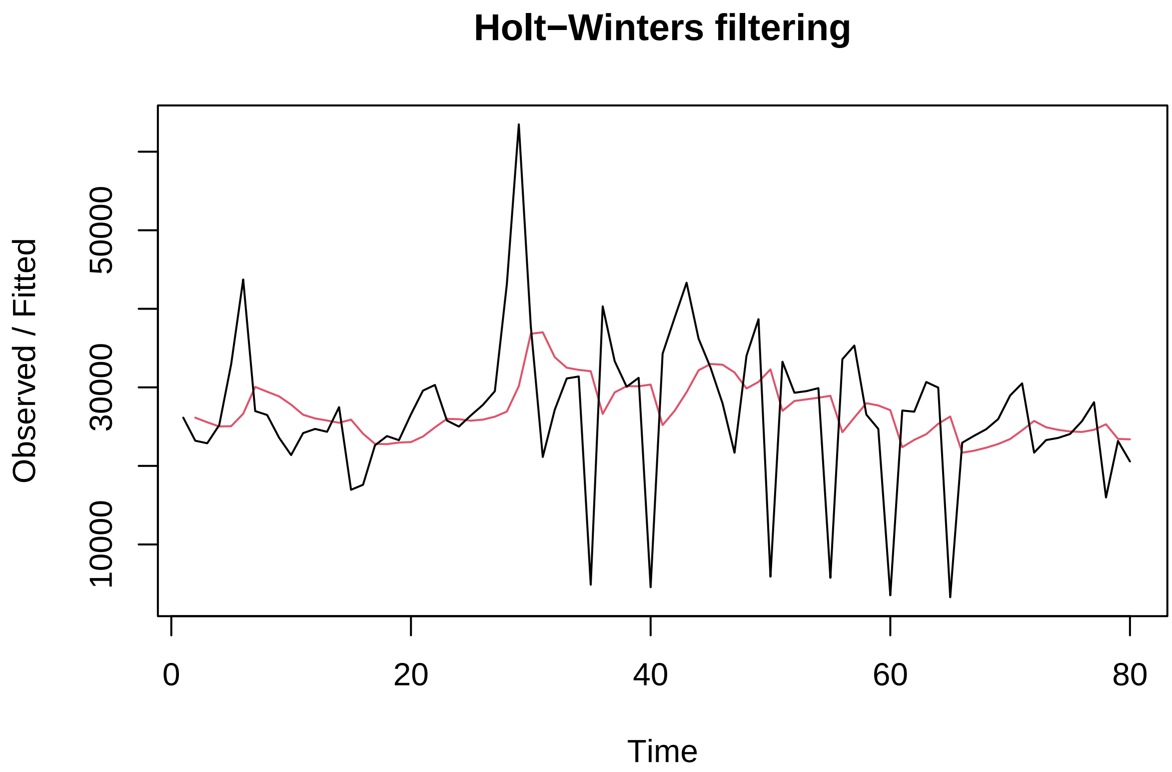



```r
autoplot(ses.2) +
  autolayer(fitted(ses.2), series="Fitted") +
  ylab("Membaca") + xlab("Periode")
```



SES menggunakan fungsi Holtwinter

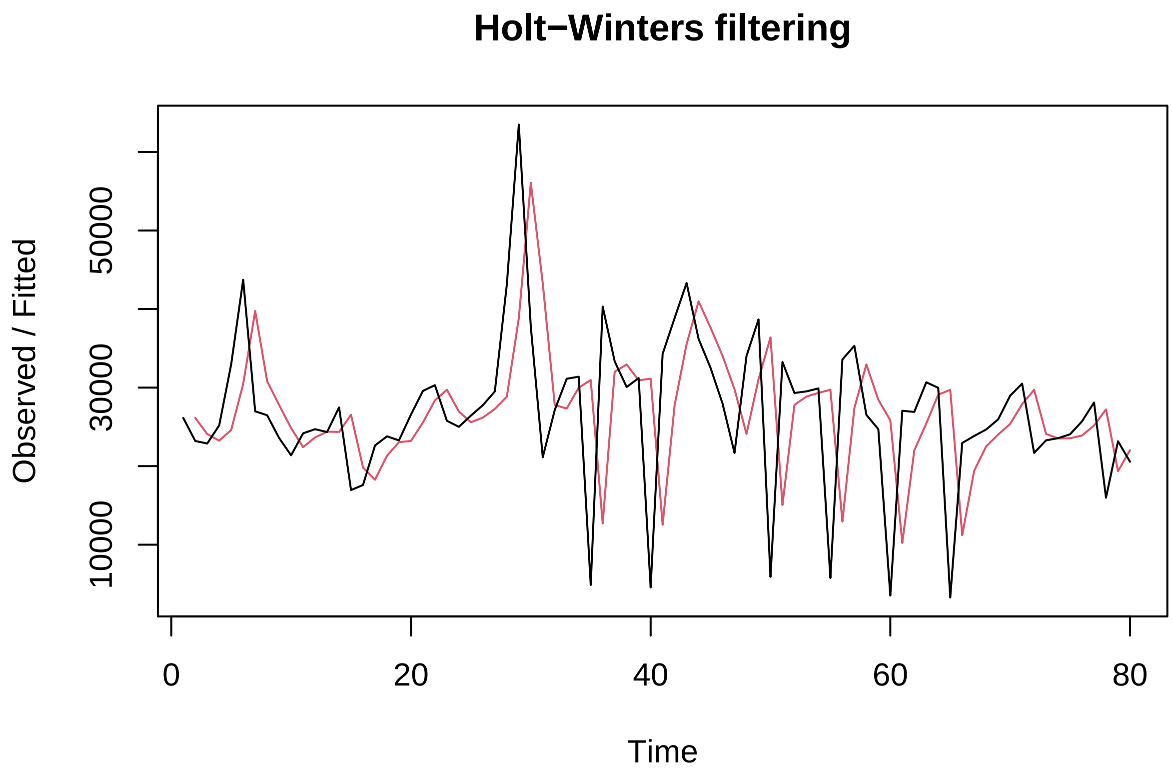
```
#Cara 2 (fungsi Holtwinter)
ses1<- HoltWinters(traints2, gamma = FALSE, beta = FALSE, alpha = 0.2)
plot(ses1)
```




```r
#ramalan
ramalan1<- forecast(ses1, h=20)
ramalan1
```

```
##     Point Forecast      Lo 80    Hi 80      Lo 95    Hi 95
##  81       22820.87 10282.155 35359.59  3644.5589 41997.19
##  82       22820.87 10033.840 35607.91  3264.7931 42376.95
##  83       22820.87  9790.255 35851.49  2892.2628 42749.48
##  84       22820.87  9551.141 36090.61  2526.5698 43115.18
##  85       22820.87  9316.261 36325.49  2167.3507 43474.40
##  86       22820.87  9085.396 36556.35  1814.2734 43827.47
##  87       22820.87  8858.348 36783.40  1467.0334 44174.71
##  88       22820.87  8634.933 37006.81  1125.3502 44516.40
##  89       22820.87  8414.983 37226.76   788.9655 44852.78
##  90       22820.87  8198.341 37443.41   457.6400 45184.11
##  91       22820.87  7984.862 37656.88   131.1522 45510.59
##  92       22820.87  7774.412 37867.34  -190.7039 45832.45
##  93       22820.87  7566.865 38074.88  -508.1200 46149.87
##  94       22820.87  7362.104 38279.64  -821.2749 46463.02
##  95       22820.87  7160.020 38481.73 -1130.3357 46772.08
##  96       22820.87  6960.510 38681.24 -1435.4589 47077.21
##  97       22820.87  6763.479 38878.27 -1736.7914 47378.54
##  98       22820.87  6568.837 39072.91 -2034.4709 47676.22
##  99       22820.87  6376.499 39265.25 -2328.6272 47970.37
## 100       22820.87  6186.384 39455.36 -2619.3826 48261.13
```

```r
ses2<- HoltWinters(traints2, gamma = FALSE, beta = FALSE, alpha = 0.7)
plot(ses2)
```

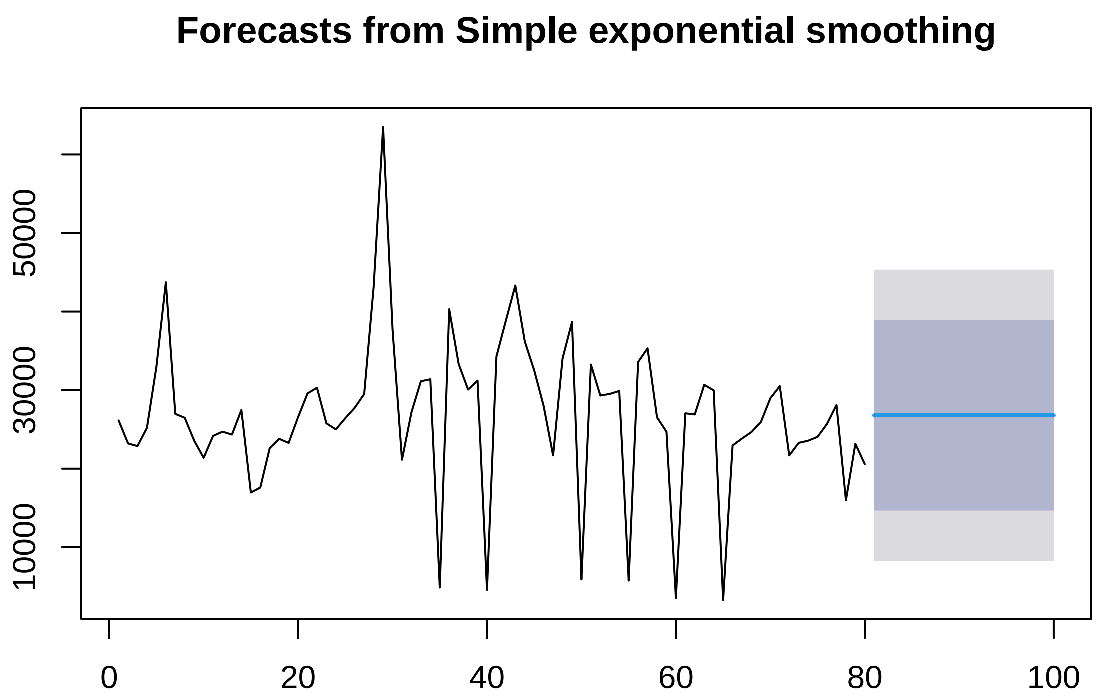



```r
#ramalan
ramalan2<- forecast(ses2, h=20)
ramalan2
```

```
##    Point Forecast       Lo 80    Hi 80       Lo 95    Hi 95
## 81       21007.52   6668.2687 35346.77   -922.4727 42937.51
## 82       21007.52   3504.2330 38510.81  -5761.4478 47776.49
## 83       21007.52    830.4047 41184.64  -9850.7154 51865.76
## 84       21007.52  -1528.3827 43543.42 -13458.1694 55473.21
## 85       21007.52  -3662.6612 45677.70 -16722.2667 58737.31
## 86       21007.52  -5626.4577 47641.50 -19725.6343 61740.68
## 87       21007.52  -7455.0818 49470.12 -22522.2734 64537.31
## 88       21007.52  -9173.1134 51188.15 -25149.7760 67164.82
## 89       21007.52 -10798.4788 52813.52 -27635.5579 69650.60
## 90       21007.52 -12344.7287 54359.77 -30000.3430 72015.38
## 91       21007.52 -13822.4013 55837.44 -32260.2483 74275.29
## 92       21007.52 -15239.8847 57254.93 -34428.1021 76443.14
## 93       21007.52 -16603.9846 58619.03 -36514.3129 78529.35
## 94       21007.52 -17920.3134 59935.35 -38527.4640 80542.51
## 95       21007.52 -19193.5638 61208.61 -40474.7325 82489.77
## 96       21007.52 -20427.7074 62442.75 -42362.1923 84377.23
## 97       21007.52 -21626.1405 63641.18 -44195.0374 86210.08
## 98       21007.52 -22791.7943 64806.84 -45977.7512 87992.79
```

```
## 99        21007.52 -23927.2201 65942.26 -47714.2350 89729.28
## 100       21007.52 -25034.6541 67049.70 -49407.9092 91422.95
```

```
#SES
ses.opt <- ses(traints2, h = 20, alpha = NULL)
plot(ses.opt)
```




```
ses.opt
```

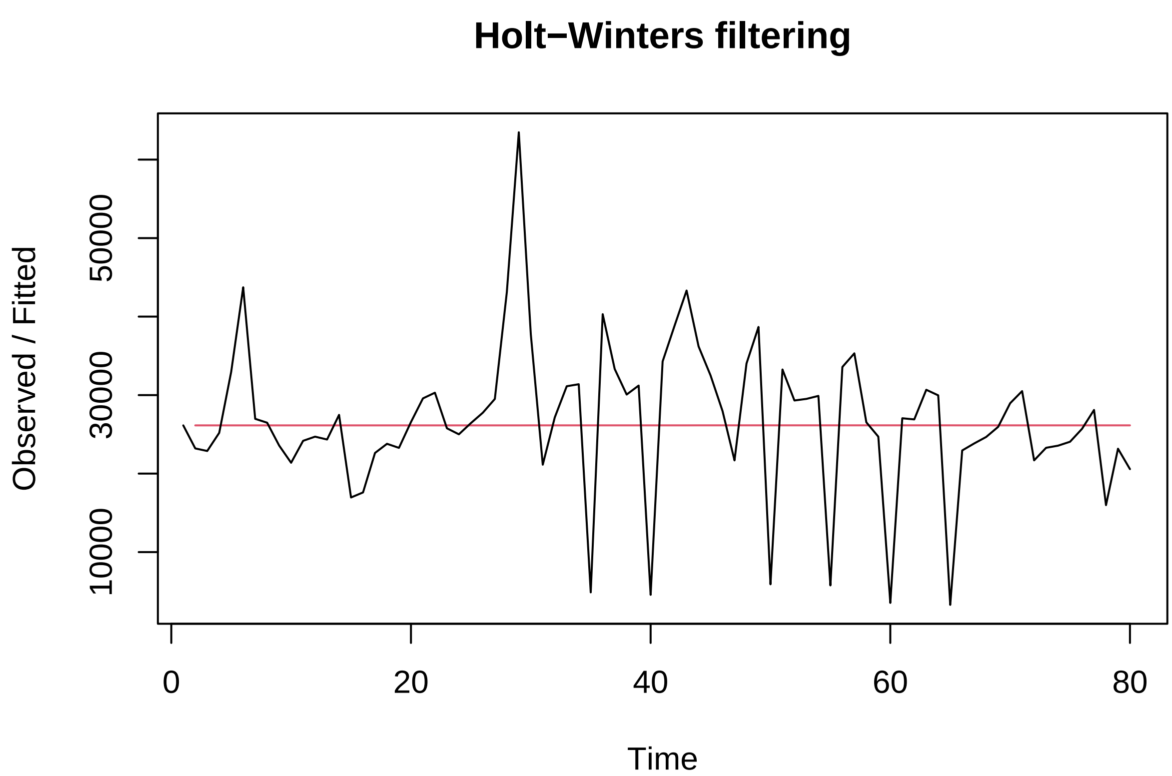
```
##     Point Forecast    Lo 80   Hi 80    Lo 95    Hi 95
##  81       26794.84 14669.38 38920.3 8250.552 45339.12
##  82       26794.84 14669.38 38920.3 8250.552 45339.12
##  83       26794.84 14669.38 38920.3 8250.552 45339.12
##  84       26794.84 14669.38 38920.3 8250.552 45339.12
##  85       26794.84 14669.38 38920.3 8250.552 45339.12
##  86       26794.84 14669.38 38920.3 8250.552 45339.12
##  87       26794.84 14669.38 38920.3 8250.552 45339.12
##  88       26794.84 14669.38 38920.3 8250.551 45339.12
##  89       26794.84 14669.38 38920.3 8250.551 45339.13
##  90       26794.84 14669.38 38920.3 8250.551 45339.13
##  91       26794.84 14669.38 38920.3 8250.551 45339.13
##  92       26794.84 14669.38 38920.3 8250.551 45339.13
##  93       26794.84 14669.38 38920.3 8250.551 45339.13
##  94       26794.84 14669.38 38920.3 8250.551 45339.13
##  95       26794.84 14669.38 38920.3 8250.551 45339.13
##  96       26794.84 14669.38 38920.3 8250.551 45339.13
##  97       26794.84 14669.38 38920.3 8250.551 45339.13
```

```
##  98       26794.84 14669.38 38920.3 8250.551 45339.13
##  99       26794.84 14669.38 38920.3 8250.550 45339.13
## 100       26794.84 14669.38 38920.3 8250.550 45339.13
```

```r
#Lamda Optimum Holt Winter
sesopt<- HoltWinters(traints2, gamma = FALSE, beta = FALSE,alpha = NULL)
sesopt
```

```
## Holt-Winters exponential smoothing without trend and without seasonal component.
##
## Call:
## HoltWinters(x = traints2, alpha = NULL, beta = FALSE, gamma = FALSE)
##
## Smoothing parameters:
##  alpha: 5.809839e-05
##  beta : FALSE
##  gamma: FALSE
##
## Coefficients:
##       [,1]
## a 26146.02
```

```r
plot(sesopt)
```




```r
#ramalan
ramalanopt<- forecast(sesopt, h=20)
ramalanopt
```

```
##     Point Forecast    Lo 80 Hi 80    Lo 95    Hi 95
##  81       26146.02 14021.05 38271 7602.475 44689.57
##  82       26146.02 14021.05 38271 7602.475 44689.57
##  83       26146.02 14021.05 38271 7602.474 44689.57
##  84       26146.02 14021.05 38271 7602.474 44689.57
##  85       26146.02 14021.05 38271 7602.474 44689.57
##  86       26146.02 14021.05 38271 7602.474 44689.57
##  87       26146.02 14021.05 38271 7602.474 44689.57
##  88       26146.02 14021.05 38271 7602.474 44689.57
##  89       26146.02 14021.05 38271 7602.474 44689.57
##  90       26146.02 14021.05 38271 7602.474 44689.57
##  91       26146.02 14021.05 38271 7602.474 44689.57
##  92       26146.02 14021.05 38271 7602.474 44689.57
##  93       26146.02 14021.05 38271 7602.474 44689.57
##  94       26146.02 14021.05 38271 7602.474 44689.57
##  95       26146.02 14021.05 38271 7602.474 44689.57
##  96       26146.02 14021.05 38271 7602.474 44689.57
##  97       26146.02 14021.05 38271 7602.474 44689.57
##  98       26146.02 14021.05 38271 7602.474 44689.57
##  99       26146.02 14021.05 38271 7602.474 44689.57
## 100       26146.02 14021.05 38271 7602.474 44689.57
```

```r
#Keakuratan Metode
#Pada data training
SSE1<-ses1$SSE
MSE1<-ses1$SSE/length(traints2)
RMSE1<-sqrt(MSE1)

akurasi1 <- matrix(c(SSE1,MSE1,RMSE1))
row.names(akurasi1)<- c("SSE", "MSE", "RMSE")
colnames(akurasi1) <- c("Akurasi lamda=0.2")
akurasi1
```

Akurasi Data Latih

```
##      Akurasi lamda=0.2
## SSE       7.470193e+09
## MSE       9.337741e+07
## RMSE      9.663199e+03
```

```r
SSE2<-ses2$SSE
MSE2<-ses2$SSE/length(traints2)
RMSE2<-sqrt(MSE2)

akurasi2 <- matrix(c(SSE2,MSE2,RMSE2))
row.names(akurasi2)<- c("SSE", "MSE", "RMSE")
colnames(akurasi2) <- c("Akurasi lamda=0.7")
akurasi2
```

```
##      Akurasi lamda=0.7
## SSE       9.765753e+09
## MSE       1.220719e+08
## RMSE      1.104862e+04
```

```r
#Cara Manual
fitted1<-ramalan1$fitted
sisaan1<-ramalan1$residuals
head(sisaan1)
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
## [1]        NA -2937.000 -2674.600   174.320  7913.456 17098.765
```

```r
resid1<-train2$Jumlah-ramalan1$fitted
head(resid1)
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
## [1]        NA -2937.000 -2674.600   174.320  7913.456 17098.765
```

```r
#Cara Manual
SSE.1=sum(sisaan1[2:length(traints2)]^2)
SSE.1
```

```
## [1] 7470192612
```

```r
MSE.1 = SSE.1/length(traints2)
MSE.1
```

```
## [1] 93377408
```

```r
MAPE.1 = sum(abs(sisaan1[2:length(traints2)]/traints2[2:length(traints2)])*
               100)/length(traints2)
MAPE.1
```

```
## [1] 56.5032
```

```r
akurasi.1 <- matrix(c(SSE.1,MSE.1,MAPE.1))
row.names(akurasi.1)<- c("SSE", "MSE", "MAPE")
colnames(akurasi.1) <- c("Akurasi lamda=0.2")
akurasi.1
```

```
##      Akurasi lamda=0.2
## SSE       7.470193e+09
## MSE       9.337741e+07
## MAPE      5.650320e+01
```

```r
fitted2<-ramalan2$fitted
sisaan2<-ramalan2$residuals
head(sisaan2)
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
## [1]        NA -2937.000 -1206.100  1952.170  8359.651 13275.895
```

```r
resid2<-train2$Jumlah-ramalan2$fitted
head(resid2)
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
## [1]        NA -2937.000 -1206.100  1952.170  8359.651 13275.895
```

```
SSE.2=sum(sisaan2[2:length(traints2)]^2)
SSE.2
```

```
## [1] 9765752766
```

```
MSE.2 = SSE.2/length(traints2)
MSE.2
```

```
## [1] 122071910
```

```
MAPE.2 = sum(abs(sisaan2[2:length(traints2)]/traints2[2:length(traints2)])*
               100)/length(traints2)
MAPE.2
```

```
## [1] 62.34285
```

```
akurasi.2 <- matrix(c(SSE.2,MSE.2,MAPE.2))
row.names(akurasi.2)<- c("SSE", "MSE", "MAPE")
colnames(akurasi.2) <- c("Akurasi lamda=0.7")
akurasi.2
```

```
##      Akurasi lamda=0.7
## SSE       9.765753e+09
## MSE       1.220719e+08
## MAPE      6.234285e+01
```

Dengan menggunakan parameter $\lambda = 0,2$ menghasilkan nilai akurasi yang paling baik dibandingkan $\lambda = 0,7$, yakni sebesar 54%.

```
selisih1<-ramalan1$mean-test2$Jumlah
SSEtesting1<-sum(selisih1^2)
MSEtesting1<-SSEtesting1/length(test2)

selisih2<-ramalan2$mean-test2$Jumlah
SSEtesting2<-sum(selisih2^2)
MSEtesting2<-SSEtesting2/length(test2)

selisihopt<-ramalanopt$mean-test2$Jumlah
SSEtestingopt<-sum(selisihopt^2)
MSEtestingopt<-SSEtestingopt/length(test2)

akurasitesting1 <- matrix(c(SSEtesting1,SSEtesting2,SSEtestingopt))
row.names(akurasitesting1)<- c("SSE1", "SSE2", "SSEopt")
akurasitesting1
```

Akurasi Data Uji

```
##              [,1]
## SSE1    718365931
## SSE2    717996221
## SSEopt 1060769597
```

```r
akurasitesting2 <- matrix(c(MSEtesting1,MSEtesting2,MSEtestingopt))
row.names(akurasitesting2)<- c("MSE1", "MSE2", "MSEopt")
akurasitesting2
```

```
##             [,1]
## MSE1   239455310
## MSE2   239332074
## MSEopt 353589866
```

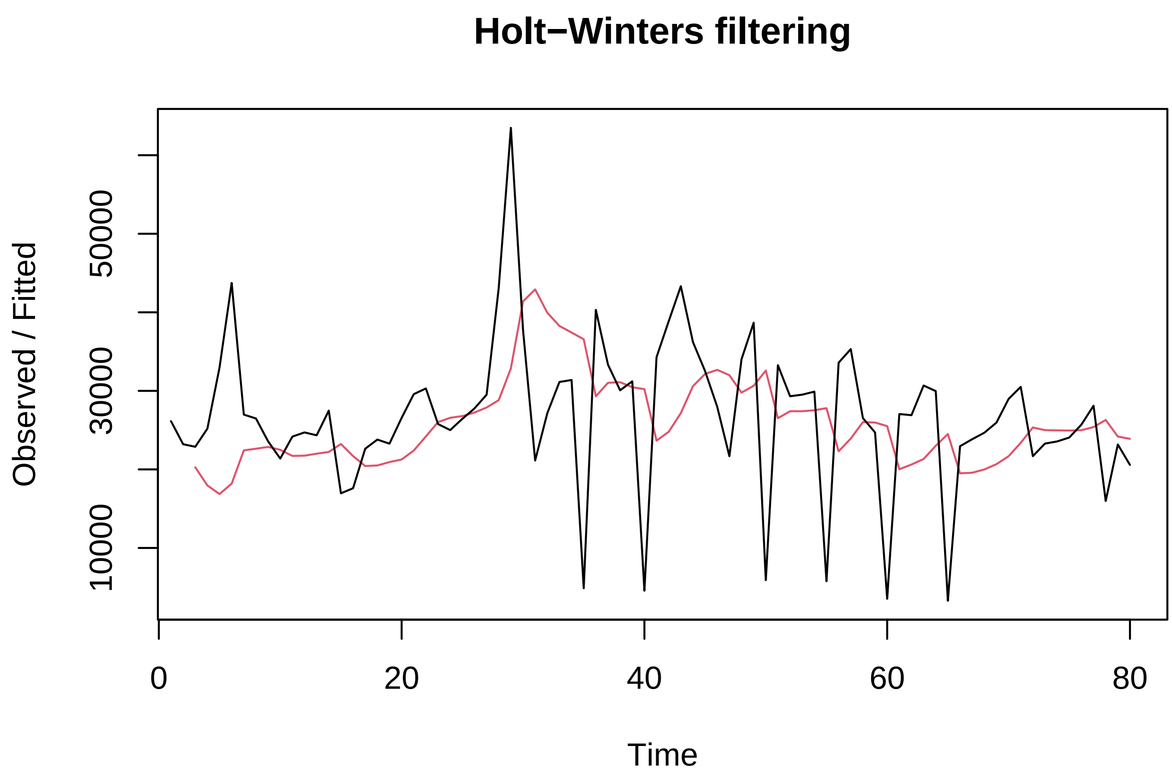
```r
accuracy(ramalanopt,test2$Jumlah)
```

Cara lain

```
##                       ME      RMSE      MAE       MPE     MAPE      MASE     ACF1
## Training set    658.5725 9424.135 6377.459 -33.85073 52.65036 0.8107900 0.141926
## Test set      -4236.9227 7282.752 5307.616 -32.27697 35.86079 0.6747769       NA
```

Double Exponential Smoothing

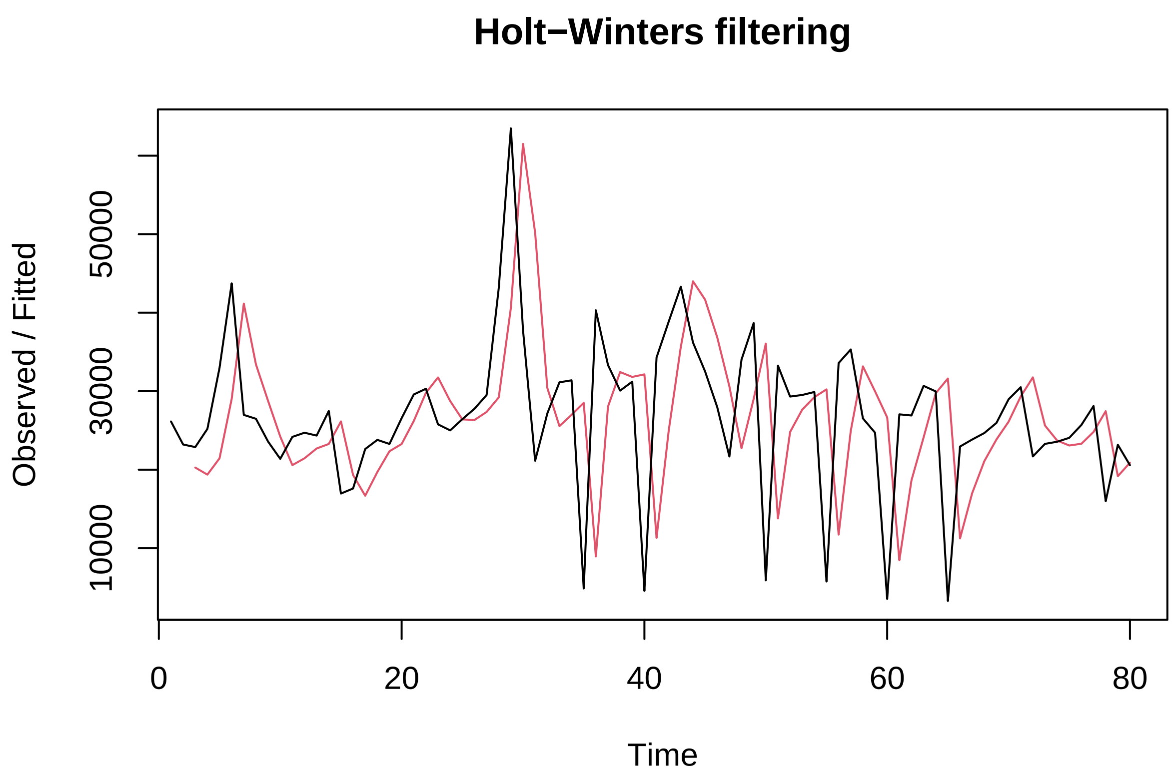
```r
#Lamda=0.2 dan gamma=0.2
des.1<- HoltWinters(traints2, gamma = FALSE, beta = 0.2, alpha = 0.2)
plot(des.1)
```




```r
#ramalan
ramalandes1<- forecast(des.1, h=20)
ramalandes1
```

```
##     Point Forecast        Lo 80    Hi 80        Lo 95    Hi 95
## 81        23003.12   9389.8375 36616.41   2183.3989 43822.85
## 82        22779.53   8779.6647 36779.39   1368.5863 44190.46
## 83        22555.93   8046.4361 37065.42    365.5761 44746.28
## 84        22332.33   7182.9963 37481.66   -836.5751 45501.23
## 85        22108.73   6186.4200 38031.04  -2242.3409 46459.80
## 86        21885.13   5057.4335 38712.83  -3850.6108 47620.87
## 87        21661.53   3799.5708 39523.49  -5655.9796 48979.04
## 88        21437.93   2418.2657 40457.60  -7650.1374 50526.00
## 89        21214.33    920.0319 41508.63  -9823.1220 52251.79
## 90        20990.73   -688.1842 42669.65 -12164.3103 54145.78
## 91        20767.14  -2399.4628 43933.73 -14663.1189 56197.39
## 92        20543.54  -4207.2005 45294.27 -17309.4492 58396.52
## 93        20319.94  -6105.2715 46745.15 -20093.9321 60733.81
## 94        20096.34  -8088.0963 48280.77 -23008.0349 63200.71
## 95        19872.74 -10150.6530 49896.13 -26044.0770 65789.56
## 96        19649.14 -12288.4532 51586.73 -29195.1942 68493.48
## 97        19425.54 -14497.5005 53348.58 -32455.2744 71306.36
## 98        19201.94 -16774.2411 55178.13 -35818.8827 74222.77
## 99        18978.34 -19115.5137 57072.20 -39281.1839 77237.87
## 100       18754.75 -21518.5009 59027.99 -42837.8697 80347.36
```

```r
#Lamda=0.6 dan gamma=0.3
des.2<- HoltWinters(traints2, gamma = FALSE, beta = 0.3, alpha = 0.6)
plot(des.2)
```

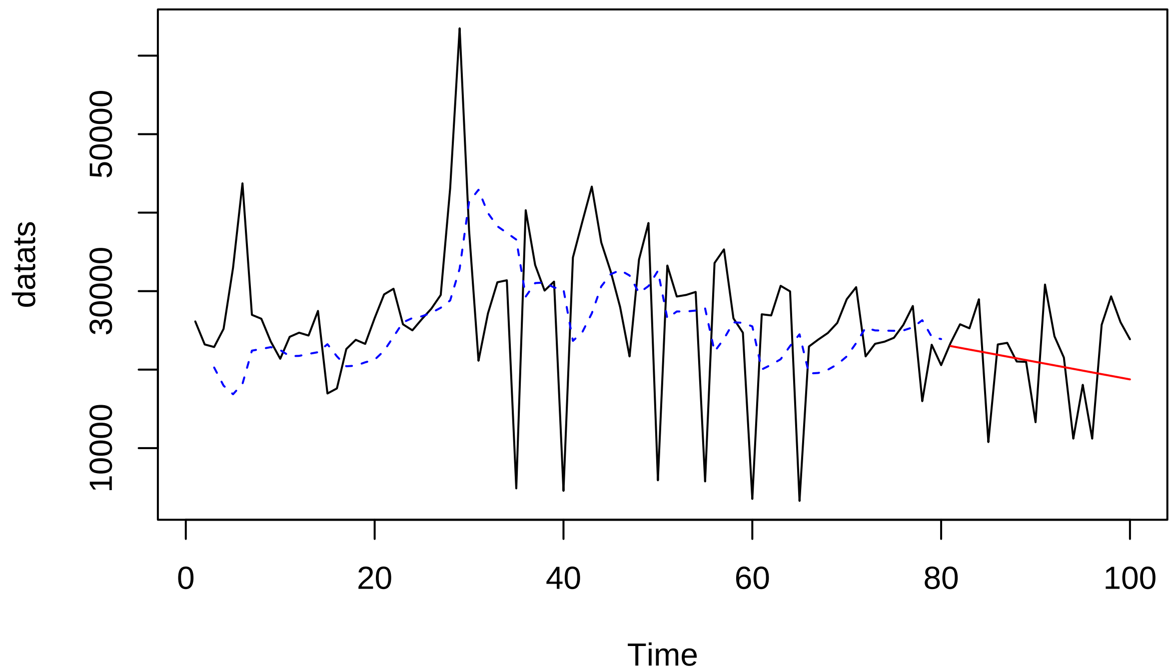


```r
#ramalan
ramalandes2<- forecast(des.2, h=20)
ramalandes2
```

```
##     Point Forecast         Lo 80     Hi 80       Lo 95     Hi 95
##  81      19958.856    4223.8595  35693.85   -4105.744  44023.46
##  82      19219.697    -735.8516  39175.25  -11299.681  49739.07
##  83      18480.538   -6547.5051  43508.58  -19796.551  56757.63
##  84      17741.379  -13051.0066  48533.77  -29351.511  64834.27
##  85      17002.221  -20140.3697  54144.81  -39802.469  73806.91
##  86      16263.062  -27744.3208  60270.45  -51040.422  83566.55
##  87      15523.903  -35812.7116  66860.52  -62988.674  94036.48
##  88      14784.745  -44308.4930  73877.98  -75590.563 105160.05
##  89      14045.586  -53203.0697  81294.24  -88802.358 116893.53
##  90      13306.427  -62473.5485  89086.40 -102589.044 129201.90
##  91      12567.269  -72101.0471  97235.58 -116921.746 142056.28
##  92      11828.110  -82069.6132 105725.83 -131776.065 155432.29
##  93      11088.951  -92365.5061 114543.41 -147130.988 169308.89
##  94      10349.793 -102976.7027 123676.29 -162968.125 183667.71
##  95       9610.634 -113892.5465 133113.81 -179271.181 198492.45
##  96       8871.475 -125103.4916 142846.44 -196025.555 213768.51
##  97       8132.317 -136600.9100 152865.54 -213218.052 229482.69
##  98       7393.158 -148376.9445 163163.26 -230836.656 245622.97
##  99       6653.999 -160424.3936 173732.39 -248870.352 262178.35
## 100       5914.841 -172736.6201 184566.30 -267308.991 279138.67
```

Membandingkan plot data latih dan data uji.

```r
#Visually evaluate the prediction
plot(datats)
lines(des.1$fitted[,1], lty=2, col="blue")
lines(ramalandes1$mean, col="red")
```

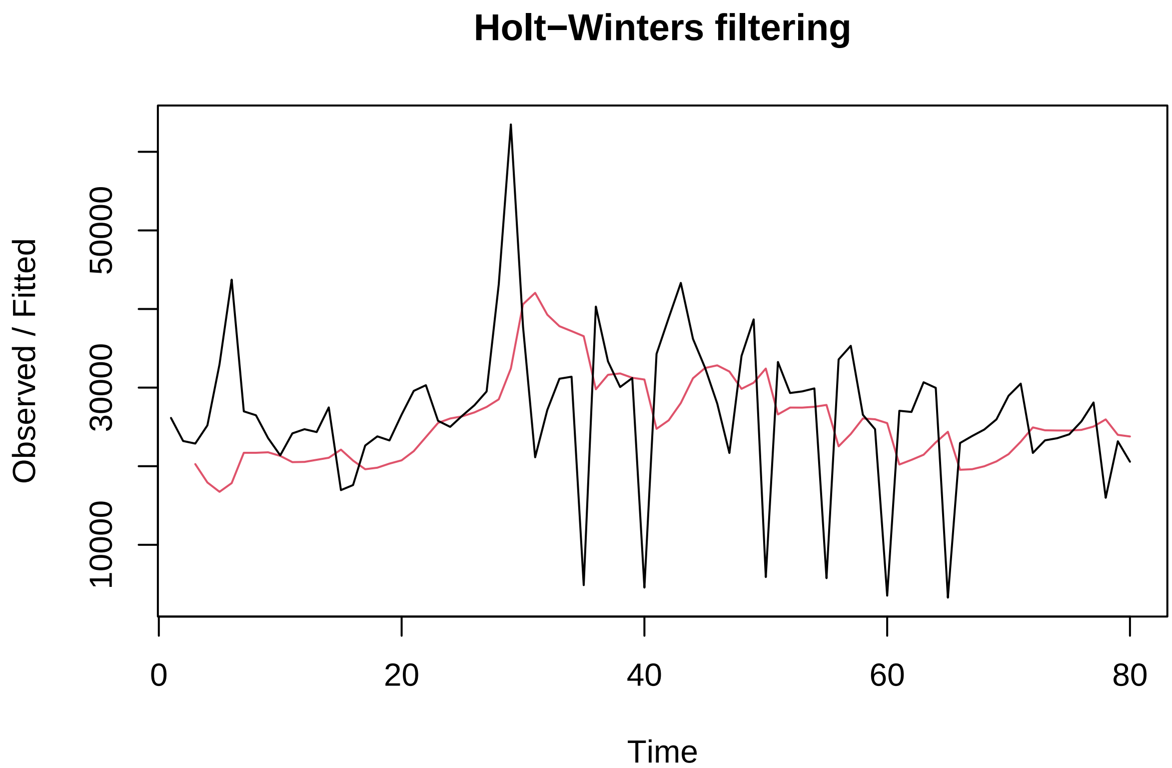


Mencari nilai parameter optimum

```
#Lamda dan gamma optimum
des.opt<- HoltWinters(traints2, gamma = FALSE)
des.opt
```

```
## Holt-Winters exponential smoothing with trend and without seasonal component.
##
## Call:
## HoltWinters(x = traints2, gamma = FALSE)
##
## Smoothing parameters:
##  alpha: 0.1955582
##  beta : 0.1703259
##  gamma: FALSE
##
## Coefficients:
##          [,1]
## a 23154.3443
## b  -153.1015
```

```
plot(des.opt)
```




```r
#ramalan
ramalandesopt<- forecast(des.opt, h=20)
ramalandesopt
```

```
##     Point Forecast         Lo 80    Hi 80       Lo 95    Hi 95
##  81       23001.24    9440.79259 36561.69   2262.3239 43740.16
##  82       22848.14    8937.07542 36759.21   1573.0020 44123.28
##  83       22695.04    8336.86008 37053.22    736.0988 44653.98
##  84       22541.94    7635.14278 37448.73   -256.0381 45339.91
##  85       22388.84    6829.54804 37948.13  -1407.0420 46184.72
##  86       22235.74    5920.03435 38551.44  -2716.9761 47188.45
##  87       22082.63    4908.45335 39256.81  -4183.0087 48348.28
##  88       21929.53    3798.05065 40061.01  -5800.1761 49659.24
##  89       21776.43    2592.98520 40959.88  -7562.1176 51114.98
##  90       21623.33    1297.91716 41948.74  -9461.7062 52708.37
##  91       21470.23     -82.31352 43022.77 -11491.5399 54432.00
##  92       21317.13   -1542.91726 44177.17 -13644.2935 56278.55
##  93       21164.03   -3079.30563 45407.36 -15912.9497 58241.00
##  94       21010.92   -4687.17981 46709.03 -18290.9341 60312.78
##  95       20857.82   -6362.57352 48078.22 -20772.1806 62487.83
##  96       20704.72   -8101.86494 49511.31 -23351.1503 64760.59
##  97       20551.62   -9901.76900 51005.01 -26022.8190 67126.06
##  98       20398.52  -11759.31811 52556.35 -28782.6482 69579.68
##  99       20245.42  -13671.83691 54162.67 -31626.5462 72117.38
## 100       20092.31  -15636.91488 55821.54 -34550.8266 74735.46
```

```r
#Akurasi Data Training
ssedes.train1<-des.1$SSE
msedes.train1<-ssedes.train1/length(traints2)
sisaandes1<-ramalandes1$residuals
head(sisaandes1)
```

Akurasi Data Latih

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
## [1]       NA       NA  2612.00  7236.12 16105.97 25551.61
```

```r
mapedes.train1 <- sum(abs(sisaandes1[3:length(traints2)]/traints2[3:length(traints2)])
                      *100)/length(traints2)

akurasides.1 <- matrix(c(ssedes.train1,msedes.train1,mapedes.train1))
row.names(akurasides.1)<- c("SSE", "MSE", "MAPE")
colnames(akurasides.1) <- c("Akurasi lamda=0.2 dan gamma=0.2")
akurasides.1
```

```
##      Akurasi lamda=0.2 dan gamma=0.2
## SSE                     8.747489e+09
## MSE                     1.093436e+08
## MAPE                    5.929683e+01
```

```r
ssedes.train2<-des.2$SSE
msedes.train2<-ssedes.train2/length(traints2)
sisaandes2<-ramalandes2$residuals
head(sisaandes2)
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
## [1]       NA       NA  2612.00  5825.64 11522.48 14721.17
```

```r
mapedes.train2 <- sum(abs(sisaandes2[3:length(traints2)]/traints2[3:length(traints2)])
                      *100)/length(traints2)

akurasides.2 <- matrix(c(ssedes.train2,msedes.train2,mapedes.train2))
row.names(akurasides.2)<- c("SSE", "MSE", "MAPE")
colnames(akurasides.2) <- c("Akurasi lamda=0.6 dan gamma=0.3")
akurasides.2
```

```
##      Akurasi lamda=0.6 dan gamma=0.3
## SSE                     1.160976e+10
## MSE                     1.451220e+08
## MAPE                    6.691419e+01
```

Dengan menggunakan lambda = 0.2 dan gamma = 0.2, didapat nilai MAPE yang lebih kecil, artinya parameter lambda dan gamma = 0.2 memberikan hasil yang lebih baik daripada lambda = 0.6 dan gamma = 0.3

```r
#Akurasi Data Testing
selisihdes1<-ramalandes1$mean-test2$Jumlah
selisihdes1
```

Akurasi Data Uji

```
## Time Series:
## Start = 81
## End = 100
## Frequency = 1
##  [1]   -385.8761  -3001.4749  -2709.0738  -6631.6727  11338.7284  -1321.8704
##  [7]  -1749.4693    399.9318    224.3329   7692.7341 -10074.8648  -3709.4637
## [13]  -1212.0626   8885.3386   1808.7397   8441.1408  -6283.4581 -10137.0569
## [19]  -7064.6558  -5113.2547
```

```r
SSEtestingdes1<-sum(selisihdes1^2)
MSEtestingdes1<-SSEtestingdes1/length(test2$Jumlah)
MAPEtestingdes1<-sum(abs(selisihdes1/test2$Jumlah)*100)/length(test2$Jumlah)

selisihdes2<-ramalandes2$mean-test2$Jumlah
selisihdes2
```

```
## Time Series:
## Start = 81
## End = 100
## Frequency = 1
##  [1]  -3430.144482  -6561.303161  -6784.461841 -11222.620520   6232.220800
##  [6]  -6943.937879  -7887.096559  -6253.255238  -6944.413918      8.427403
## [11] -18274.731276 -12424.889956 -10443.048635   -861.207315  -8453.365994
## [16]  -2336.524674 -17576.683353 -21945.842033 -19389.000712 -17953.159391
```

```r
SSEtestingdes2<-sum(selisihdes2^2)
MSEtestingdes2<-SSEtestingdes2/length(test2$Jumlah)
MAPEtestingdes2<-sum(abs(selisihdes2/test2$Jumlah)*100)/length(test2$Jumlah)

selisihdesopt<-ramalandesopt$mean-test2$Jumlah
selisihdesopt
```

```
## Time Series:
## Start = 81
## End = 100
## Frequency = 1
##  [1]  -387.7572 -2932.8587 -2569.9601 -6422.0616 11618.8369  -971.2645
##  [7] -1328.3660   891.5325   786.4311  8325.3296 -9371.7719 -2935.8734
## [13]  -367.9748  9799.9237  2793.8222  9496.7208 -5157.3807 -8940.4822
## [19] -5797.5836 -3775.6851
```

```r
SSEtestingdesopt<-sum(selisihdesopt^2)
MSEtestingdesopt<-SSEtestingdesopt/length(test2$Jumlah)
MAPEtestingdesopt<-sum(abs(selisihdesopt/test2$Jumlah)*100)/length(test2$Jumlah)

akurasitestingdes <-
  matrix(c(SSEtestingdes1,MSEtestingdes1,MAPEtestingdes1,SSEtestingdes2,MSEtestingdes2,
           MAPEtestingdes2,SSEtestingdesopt,MSEtestingdesopt,MAPEtestingdesopt),
         nrow=3,ncol=3)
```

```
row.names(akurasitestingdes)<- c("SSE", "MSE", "MAPE")
colnames(akurasitestingdes) <- c("des ske1","des ske2","des opt")
akurasitestingdes
```

```
##          des ske1     des ske2      des opt
## SSE  7.417414e+08 2.627256e+09 7.100444e+08
## MSE  3.708707e+07 1.313628e+08 3.550222e+07
## MAPE 2.759466e+01 4.086055e+01 2.770679e+01
```

```
MSEfull <-
  matrix(c(MSEtesting1,MSEtesting2,MSEtestingopt,MSEtestingdes1,MSEtestingdes2,
           MSEtestingdesopt),nrow=3,ncol=2)
row.names(MSEfull)<- c("ske 1", "ske 2", "ske opt")
colnames(MSEfull) <- c("SES","DES")
MSEfull
```

Perbandingan SES dengan DES

```
##               SES       DES
## ske 1   239455310  37087068
## ske 2   239332074 131362823
## ske opt 353589866  35502219
```

Berdasarkan nilai akurasi MSE, metode DES lebih baik dibandingkan metode SES. Hal ini dikarenakan nilai MES pada metode DES lebih kecil dibandingkan metode SES.

```
accuracy(ramalandesopt,test2$Jumlah)
```

Akurasi DES

```
##                     ME      RMSE      MAE        MPE     MAPE      MASE
## Training set 1071.5242 10567.693 7371.936 -35.466737 60.87888 0.9372216
## Test set      362.3212  5958.374 4733.581  -8.901525 27.70679 0.6017977
##                   ACF1
## Training set 0.1260325
## Test set            NA
```

Pemulusan Data Musiman

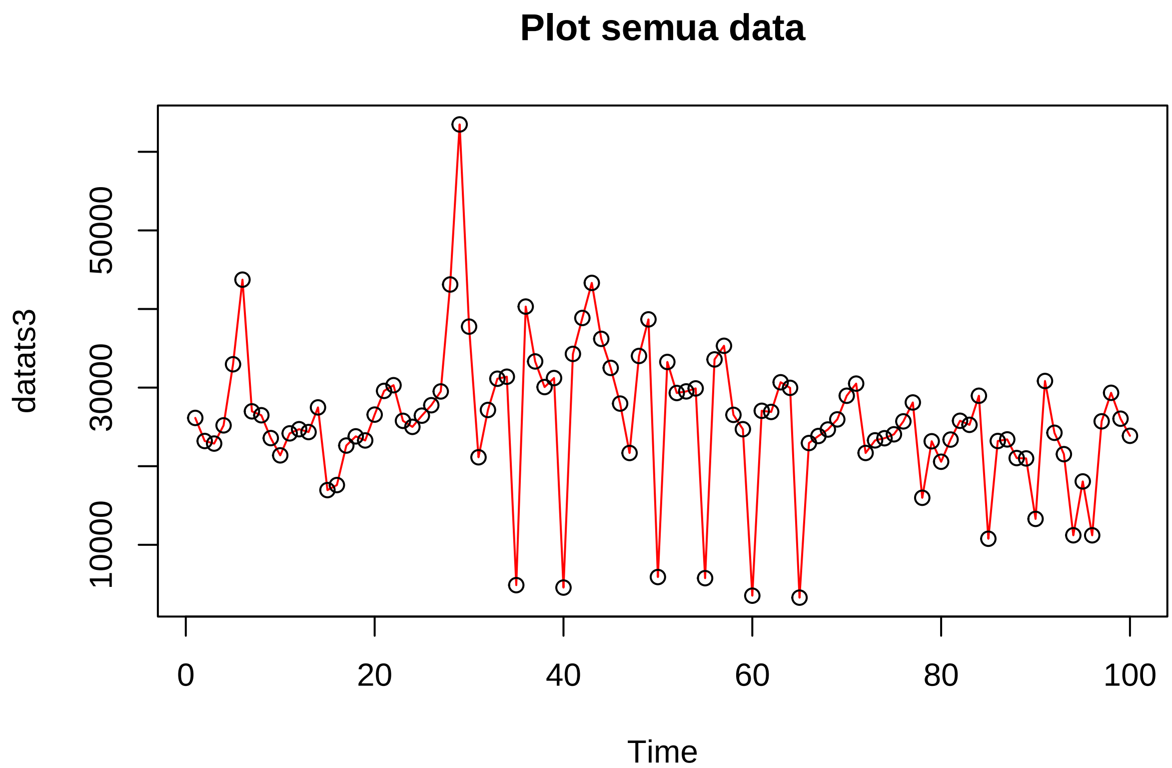
```
datats3<-ts(dataa$Jumlah)
```

Pembagian data latih dan data uji

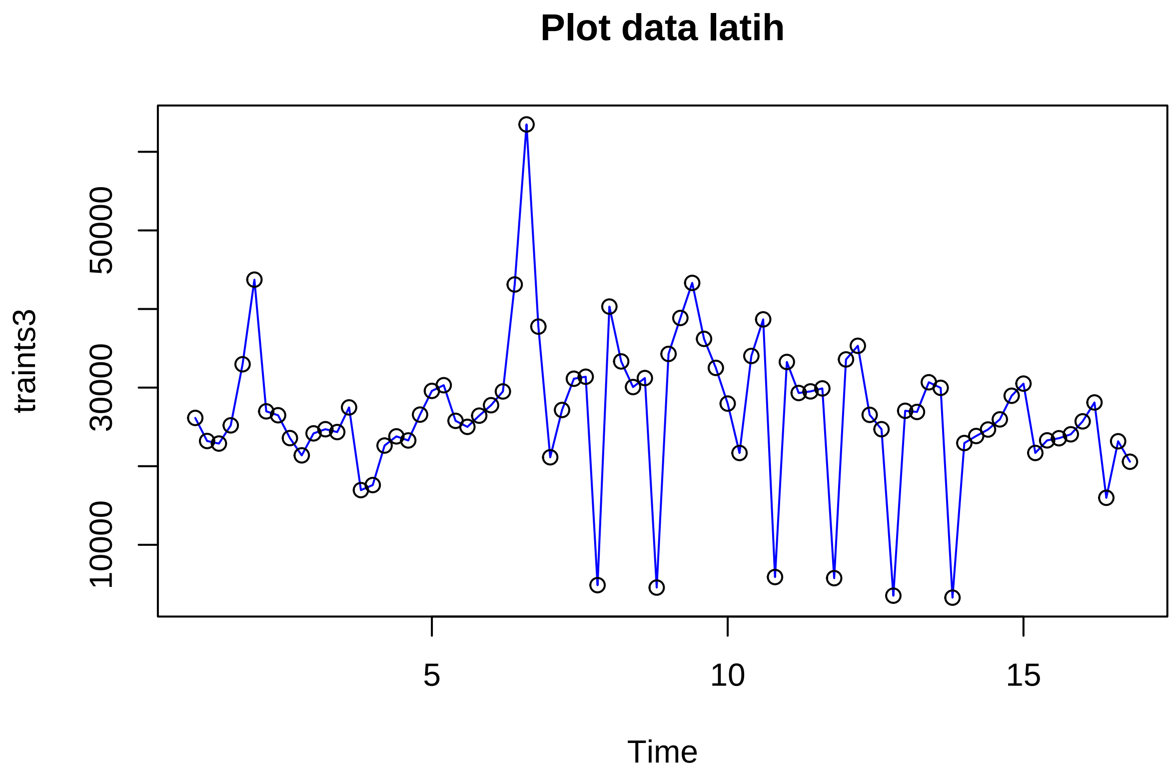
```
train3<-dataa[1:80,2]
uji3<-dataa[81:100,2]
traints3<-ts(train3,frequency = 5)
ujits3<-ts(uji3,frequency = 5)
```

Eksplorasi data

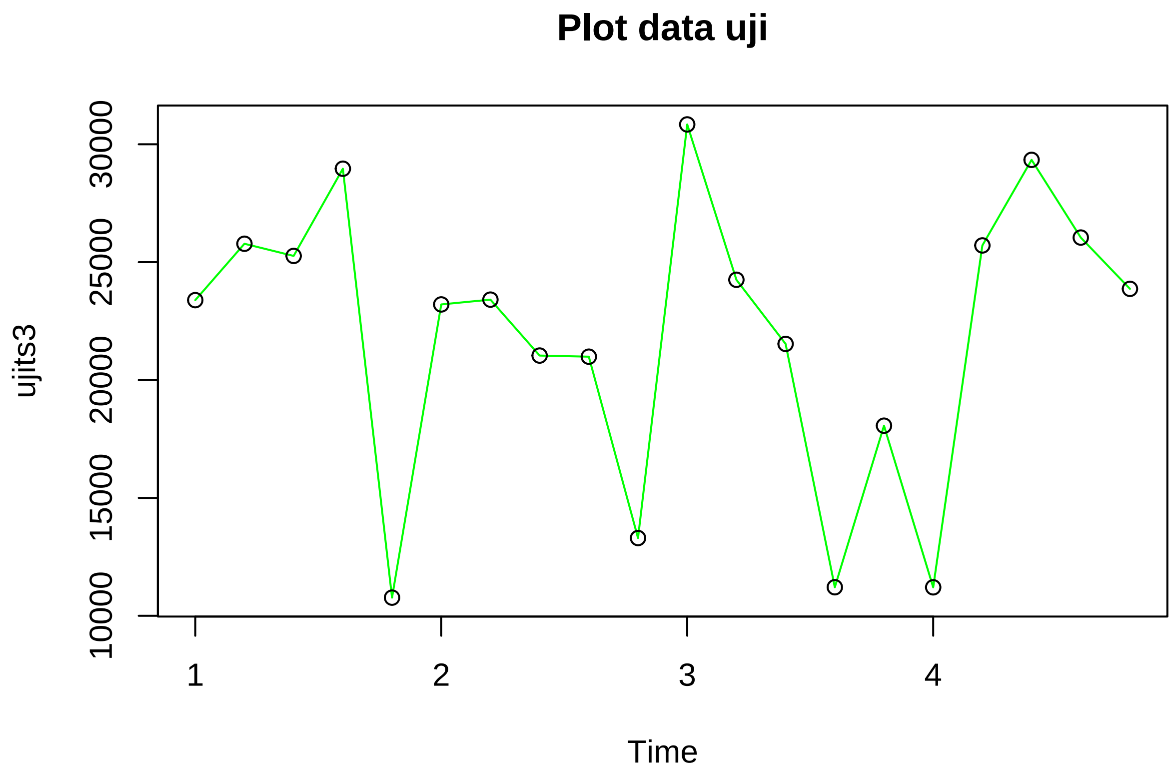
```
plot(datats3, col="red",main="Plot semua data")
points(datats3)
```




```
plot(traints3, col="blue",main="Plot data latih")
points(traints3)
```




```
plot(ujits3, col="green",main="Plot data uji")
points(ujits3)
```



Winter Aditif

```
winter1 <- HoltWinters(traints3,alpha=0.2,beta=0.1,gamma=0.1,seasonal = "additive")
winter1$fitted
```

Pemulusan

```
## Time Series:
## Start = c(2, 1)
## End = c(16, 5)
## Frequency = 5
##          xhat    level        trend       season
##   2.0 39999.87 27786.85   455.137143 11757.880000
##   2.2 24841.98 28989.42   529.879733 -4677.320000
##   2.4 27038.38 29946.50   572.600211 -3480.720000
##   2.6 25659.36 30407.62   561.452589 -5309.720000
##   2.8 32782.07 30552.41   519.785440  1709.880000
##   3.0 41140.37 28791.78   291.744011 12056.850362
##   3.2 21137.11 25691.05   -47.503419 -4506.438089
##   3.4 22856.11 26357.52    23.894474 -3525.310488
##   3.6 21255.82 26678.60    53.612348 -5476.388598
##   3.8 28952.85 27977.04   178.095961   797.714287
##   4.0 36394.45 25756.37   -61.781125 10699.860641
##   4.2 17277.62 21936.10 -437.630098 -4220.846516
##   4.4 18831.70 22568.74 -330.602532 -3406.438996
##   4.6 18021.23 23231.00 -231.316578 -4978.454143
```

```
##   4.8 23763.06 24051.04 -126.181181  -161.794057
##   5.0 33611.13 24484.84  -70.182441  9196.464748
##   5.2 19665.12 23608.64 -150.784977 -3792.736250
##   5.4 22638.55 25585.83   62.012714 -3009.295182
##   5.6 21841.30 26274.53  124.681791 -4557.912554
##   5.8 27282.33 27032.15  187.975765    62.200902
##   6.0 36094.65 27049.66  170.929159  8874.054606
##   6.2 22616.13 25553.46    4.216217 -2941.545486
##   6.4 24319.93 26936.45  142.093541 -2758.618873
##   6.6 27052.14 30838.76  518.114997 -4304.736660
##   6.8 39883.79 38643.05 1246.732218    -5.985521
##   7.0 48876.92 39465.42 1204.296328  8207.202839
##   7.2 33380.76 35121.33  649.457921 -2390.036190
##   7.4 33795.61 34525.24  524.902819 -1254.533049
##   7.6 33598.12 34516.82  471.570617 -1390.267778
##   7.8 34797.57 34545.97  427.328138  -175.729083
##   8.0 34803.75 28987.18 -171.283182  5987.849214
##   8.2 26967.95 29917.35  -61.138140 -2888.256601
##   8.4 29726.64 31128.42   66.082773 -1467.861855
##   8.6 29769.33 31263.57   72.989952 -1567.237693
##   8.8 29157.65 31625.90  101.923414 -2570.174363
##   9.0 32846.25 26807.89 -390.069552  6428.429381
##   9.2 23965.57 26706.17 -361.234608 -2379.372951
##   9.4 27820.03 29323.63  -63.365915 -1440.233136
##   9.6 31155.38 32360.25  246.633562 -1451.503847
##   9.8 29425.63 33616.21  347.565887 -4538.146226
##  10.0 41533.23 34580.25  409.213276  6543.769155
##  10.2 31225.77 32275.82  137.848615 -1187.898176
##  10.4 30251.43 30504.71  -53.046740  -200.235228
##  10.6 30181.03 31206.38   22.424652 -1047.774549
##  10.8 28829.67 32928.80  192.424059 -4291.556668
##  11.0 33727.43 28535.29 -266.169255  5458.310509
##  11.2 25949.08 28176.03 -275.477864 -1951.479596
##  11.4 28467.01 28573.54 -208.179392   101.650339
##  11.6 28020.62 28575.56 -187.159625  -367.776922
##  11.8 22488.11 28763.67 -149.632074 -6125.929921
##  12.0 30204.20 25267.42 -484.294331  5421.076073
##  12.2 23360.32 25459.29 -416.678371 -1682.285707
##  12.4 27442.35 27434.14 -177.524799   185.729407
##  12.6 26661.81 27075.15 -195.671748  -217.666717
##  12.8 18789.39 26488.71 -234.747950 -7464.578951
##  13.0 28353.47 23201.89 -539.955708  5691.539915
##  13.2 21109.54 22401.24 -566.025177  -725.671421
##  13.4 22657.11 22994.11 -450.136022   113.141610
##  13.6 23484.24 24147.95 -289.738239  -373.971524
##  13.8 16309.93 25155.36 -160.022984 -8685.409982
##  14.0 27555.77 22389.15 -420.641552  5587.262038
##  14.2 20271.82 21046.76 -512.817015  -262.114802
##  14.4 21561.29 21247.97 -441.413509   754.732745
##  14.6 21193.29 21427.70 -379.299385   144.889493
##  14.8 11988.79 22000.74 -284.065233 -9727.884253
##  15.0 30385.72 25111.72   55.438869  5218.560184
##  15.2 25273.22 25191.82   57.904484    23.499223
##  15.4 25521.85 24532.48  -13.819895  1003.189242
```

```
## 15.6 24539.44 24072.09  -58.476800   525.826102
## 15.8 15369.35 23817.32  -78.105526 -8369.867848
## 16.0 30801.48 25477.35   95.707484  5228.422644
## 16.2 24283.92 24553.56   -6.242069  -263.398291
## 16.4 26207.37 25312.53   70.279550   824.561621
## 16.6 23650.82 23337.74 -134.227945   447.311200
## 16.8 15289.77 23108.15 -143.764373 -7674.615811
```

```r
xhat1 <- winter1$fitted[,2]

winter1.opt<- HoltWinters(traints3, alpha= NULL,  beta = NULL, gamma = NULL, seasonal = "additive")
winter1.opt
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = traints3, alpha = NULL, beta = NULL, gamma = NULL,     seasonal = "additive")
##
## Smoothing parameters:
##  alpha: 0.07511462
##  beta : 0.07011048
##  gamma: 0.3165856
##
## Coefficients:
##           [,1]
## a   29375.5675
## b    -101.6719
## s1  -3198.7136
## s2  -4406.2841
## s3  -7005.2449
## s4  -4244.6107
## s5 -12103.5040
```

```r
winter1.opt$fitted
```

```
## Time Series:
## Start = c(2, 1)
## End = c(16, 5)
## Frequency = 5
##           xhat    level      trend       season
##  2.0 39999.870 27786.85 455.137143  11757.88000
##  2.2 24320.202 28522.70 474.818069  -4677.32000
##  2.4 26205.256 29197.16 488.814891  -3480.72000
##  2.6 24887.235 29706.69 490.267047  -5309.72000
##  2.8 32291.704 30098.46 483.361660   1709.88000
##  3.0 43040.224 29762.20 425.897115  12852.13161
##  3.2 25198.724 28771.26 326.562573  -3899.10234
##  3.4 25984.883 29060.89 323.972995  -3399.98067
##  3.6 23883.124 29261.46 315.321043  -5693.65678
##  3.8 28696.096 29846.96 334.263351  -1485.12562
##  4.0 36901.217 29299.59 272.452027   7329.17094
##  4.2 24250.127 28122.39 170.816140  -4043.08186
##  4.4 24452.692 28171.44 162.278762  -3881.02572
##  4.6 23802.739 28284.39 158.820411  -4640.47206
##  4.8 23638.042 28403.80 156.056968  -4921.81044
```

```
##   5.0 30629.277 28779.56  171.460739   1678.25641
##   5.2 24520.463 28872.28  165.940178  -4517.75665
##   5.4 25595.817 29472.72  196.403412  -4073.30863
##   5.6 25085.573 29682.36  197.331246  -4794.11839
##   5.8 26005.259 29873.71  196.912190  -4065.36697
##   6.0 31672.995 30102.53  199.149010   1371.31542
##   6.2 27362.205 30007.68  178.536654  -2824.01305
##   6.4 26515.675 30347.55  189.847633  -4021.72144
##   6.6 27244.578 31784.70  277.296617  -4817.41775
##   6.8 31311.170 34784.03  468.139806  -3941.00063
##   7.0 36464.111 35736.72  502.111952    225.27665
##   7.2 33313.651 35087.39  421.383923  -2195.12712
##   7.4 36275.370 35046.02  388.939954    840.40696
##   7.6 41203.620 35048.40  361.837515   5793.38698
##   7.8 32930.758 34672.79  310.134770  -2052.16294
##   8.0 28774.095 32874.92  162.341993  -4263.16939
##   8.2 30127.955 33903.85  223.099046  -3998.99876
##   8.4 33940.878 34367.40  239.956779   -666.47779
##   8.6 37455.064 34316.75  219.582024   2918.73515
##   8.8 23985.027 34067.68  186.725107 -10269.38206
##   9.0 31994.470 32795.16   84.416135   -885.10161
##   9.2 30086.628 33051.85   96.494604  -3061.71555
##   9.4 32150.664 33807.28  142.692738  -1799.30603
##   9.6 36082.368 34788.95  201.514057   1091.90380
##   9.8 19243.874 34999.45  202.144079 -15957.71986
##  10.0 36256.378 36197.92  271.997232   -213.54358
##  10.2 35582.332 35847.12  228.332170   -493.11786
##  10.4 36657.515 35031.26  155.123284   1471.13571
##  10.6 36256.832 34988.64  141.259611   1126.93274
##  10.8 23392.103 35311.99  154.026061 -12073.91259
##  11.0 31572.706 34152.10   61.907019  -2641.30349
##  11.2 29848.201 34340.90   70.803383  -4563.50252
##  11.4 35139.889 34371.58   67.990106    700.32103
##  11.6 35892.407 34017.28   38.383431   1836.74224
##  11.8 16416.438 33605.32    6.809687 -17195.69433
##  12.0 30615.296 32811.30  -49.336874  -2146.66926
##  12.2 28231.417 32985.03  -33.697458  -4719.91957
##  12.4 32541.467 33483.64    3.622770   -945.79821
##  12.6 33089.337 33036.09  -28.009216     81.25474
##  12.8 11988.953 32378.52  -72.148033 -20317.42060
##  13.0 30277.082 31670.91 -116.700865  -1277.12402
##  13.2 28533.180 31311.81 -133.695719  -2644.92986
##  13.4 28208.935 31055.73 -142.275503  -2704.52393
##  13.6 28596.723 31098.85 -129.277879  -2372.84577
##  13.8  8156.136 31072.72 -122.045760 -22794.54022
##  14.0 28214.571 30584.33 -147.730327  -2222.03106
##  14.2 26743.497 30040.93 -175.471048  -3121.96243
##  14.4 27474.902 29647.51 -190.751263  -1981.86121
##  14.6 27069.567 29245.85 -205.538577  -1970.74287
##  14.8  4522.590 28956.59 -211.408245 -24222.59183
##  15.0 26733.993 30581.09  -82.691915  -3764.40403
##  15.2 26747.607 30781.95  -62.811512  -3971.53623
##  15.4 27445.526 30339.02  -89.462297  -2804.02994
##  15.6 27528.894 29937.34 -111.351898  -2297.09410
```

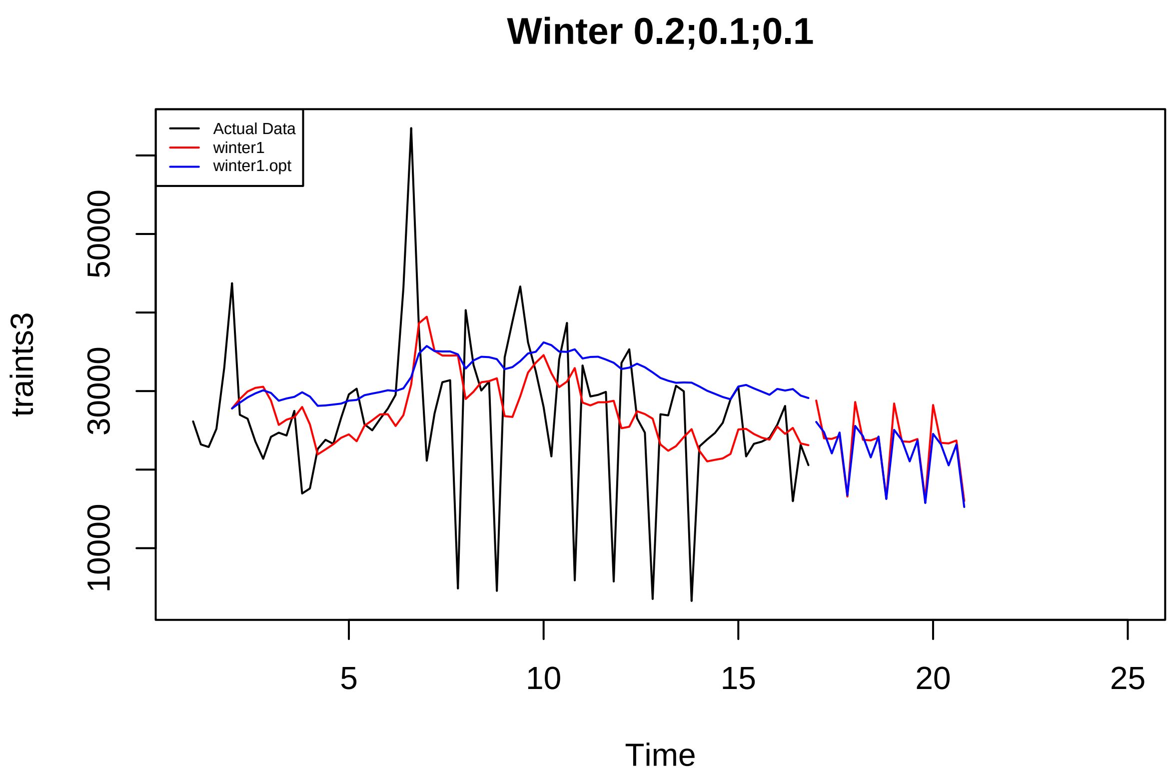
```
## 15.8 12329.436 29527.72 -132.263902 -17066.01551
## 16.0 27547.040 30276.59  -70.486979  -2659.06159
## 16.2 24534.160 30067.66  -80.193020  -5453.30920
## 16.4 26173.623 30256.07  -61.361497  -4021.08296
## 16.6 25854.339 29429.17 -115.033861  -3459.79311
## 16.8 15352.407 29112.80 -129.149388 -13631.24340
```

```
xhat1.opt <- winter1.opt$fitted[,2]
```

```
forecast1 <- predict(winter1, n.ahead = 20)
forecast1.opt <- predict(winter1.opt, n.ahead = 20)
```

Peramalan

```
plot(traints3,main="Winter 0.2;0.1;0.1",type="l",col="black",
     xlim=c(1,25),pch=12)
lines(xhat1,type="l",col="red")
lines(xhat1.opt,type="l",col="blue")
lines(forecast1,type="l",col="red")
lines(forecast1.opt,type="l",col="blue")
legend("topleft",c("Actual Data",expression(paste(winter1)),
                   expression(paste(winter1.opt))),cex=0.5,
       col=c("black","red","blue"),lty=1)
```



Plot DW

```
SSE1<-winter1$SSE
MSE1<-winter1$SSE/length(traints3)
RMSE1<-sqrt(MSE1)
akurasi1 <- matrix(c(SSE1,MSE1,RMSE1))
row.names(akurasi1)<- c("SSE", "MSE", "RMSE")
colnames(akurasi1) <- c("Akurasi")
akurasi1
```

Akurasi Data Latih

```
##          Akurasi
## SSE  8.43520e+09
## MSE  1.05440e+08
## RMSE 1.02684e+04
```

```
SSE1.opt<-winter1.opt$SSE
MSE1.opt<-winter1.opt$SSE/length(traints3)
RMSE1.opt<-sqrt(MSE1.opt)
akurasi1.opt <- matrix(c(SSE1.opt,MSE1.opt,RMSE1.opt))
row.names(akurasi1.opt)<- c("SSE1.opt", "MSE1.opt", "RMSE1.opt")
colnames(akurasi1.opt) <- c("Akurasi")
akurasi1.opt
```

```
##                Akurasi
## SSE1.opt  6.916439e+09
## MSE1.opt  8.645549e+07
## RMSE1.opt 9.298144e+03
```

```
akurasi1.train = data.frame(Model_Winter.Latih = c("Winter 1","Winter1 optimal"),
                            Nilai_SSE=c(SSE1,SSE1.opt),
                            Nilai_MSE=c(MSE1,MSE1.opt),Nilai_RMSE=c(RMSE1,RMSE1.opt))
akurasi1.train
```

```
##   Model_Winter.Latih  Nilai_SSE Nilai_MSE Nilai_RMSE
## 1           Winter 1 8435199678 105439996  10268.398
## 2    Winter1 optimal 6916438840  86455485   9298.144
```

Berdasarkan nilai RMSE, metode Winter Aditif dengan parameter optimal menghasilkan nilai RMSE yang lebih kecil, sehingga dapat disimpulkan bahwa metode Winter Aditif Optimal lebih baik.

```
forecast1<-data.frame(forecast1)
ujits3df<-data.frame(ujits3)
selisih1<-forecast1-ujits3df
SSEtesting1<-sum(selisih1^2)
MSEtesting1<-SSEtesting1/length(ujits3df)
RMSEtesting1<-sqrt(MSEtesting1)


forecast1.opt<-data.frame(forecast1.opt)
selisih1.opt<-forecast1.opt-ujits3df
SSEtesting1.opt<-sum(selisih1.opt^2)
MSEtesting1.opt<-SSEtesting1.opt/length(ujits3df)
RMSEtesting1.opt<-sqrt(MSEtesting1.opt)
```

```
akurasi1.uji<-data.frame("Nilai RMSE Uji",RMSEtesting1,RMSEtesting1.opt)
akurasi1.uji
```

Akurasi Data Uji

```
##   X.Nilai.RMSE.Uji. RMSEtesting1 RMSEtesting1.opt
## 1    Nilai RMSE Uji     26808.85         25251.44
```

Pada pengujian akurasi RMSE pada data uji, metode Winter dengan parameter optimal menghasilkan nilai RMSE yang lebih besar, sehingga dapat disimpulkan bahwa metode Winter dengan parameter alpha=0.2,beta=0.1,gamma=0.1 lebih baik.

Winter Multiplikatif

```
winter2 <- HoltWinters(traints3,alpha=0.2,beta=0.1,gamma=0.3,seasonal = "multiplicative")
winter2$fitted
```

Pemulusan

```
## Time Series:
## Start = c(2, 1)
## End = c(16, 5)
## Frequency = 5
##          xhat    level        trend    season
##  2.0 38769.765 27786.85  455.1371429 1.3727703
##  2.2 25236.532 28965.67  527.5051863 0.8556736
##  2.4 26873.950 29900.22  568.2092012 0.8820262
##  2.6 25684.844 30379.32  559.2990400 0.8301870
##  2.8 32775.091 30430.58  508.4949751 1.0593429
##  3.0 41118.537 28787.73  293.3598939 1.4139271
##  3.2 23253.258 26684.85   53.7359869 0.8696518
##  3.4 23871.587 27072.91   87.1687187 0.8789219
##  3.6 22262.929 27267.12   97.8730281 0.8135550
##  3.8 27844.119 28647.53  226.1267039 0.9643433
##  4.0 33578.494 26616.14    0.3747492 1.2615662
##  4.2 21031.620 24083.71 -252.9055662 0.8825392
##  4.4 21172.387 24192.80 -216.7059195 0.8830623
##  4.6 20928.353 24570.30 -157.2851409 0.8572620
##  4.8 21532.410 24961.19 -102.4676556 0.8661912
##  5.0 28698.701 26020.27   13.6865907 1.1023566
##  5.2 23559.018 26194.03   29.6941007 0.8983857
##  5.4 25357.343 27725.52  179.8741762 0.9086895
##  5.6 24799.260 27996.66  189.0006605 0.8798537
##  5.8 25941.647 28232.66  193.7000746 0.9125913
##  6.0 31911.603 28533.38  204.4026310 1.1104405
##  6.2 26904.177 27989.87  129.6106335 0.9567808
##  6.4 26316.660 28664.18  184.0812746 0.9122441
##  6.6 29168.051 32532.44  552.4988734 0.8816111
##  6.8 38685.151 40869.54 1330.9588595 0.9166989
##  7.0 46550.959 41999.09 1310.8180824 1.0748339
##  7.2 38574.918 38580.63  837.8899673 0.9785989
##  7.4 39053.487 37084.18  604.4558528 1.0362140
##  7.6 39653.716 36159.12  451.5050591 1.0831203
##  7.8 32248.741 35083.98  298.8402739 0.9114237
```

```
##  8.0 26651.334 29374.26 -302.0162185 0.9167279
##  8.2 28993.446 32052.33   -4.0070293 0.9046790
##  8.4 32556.369 33006.80   91.8403197 0.9836165
##  8.6 33501.697 32593.49   41.3253229 1.0265631
##  8.8 22134.625 32189.50   -3.2057322 0.6877033
##  9.0 27065.096 27074.61 -514.3745495 1.0190081
##  9.2 25844.148 27977.87 -372.6111147 0.9362038
##  9.4 29240.628 30385.61  -94.5765051 0.9653230
##  9.6 33723.244 33208.06  197.1262930 1.0095213
##  9.8 18160.297 33896.26  246.2338504 0.5318972
## 10.0 43588.004 39537.41  785.7254081 1.0809677
## 10.2 39408.533 37432.57  496.6695132 1.0390013
## 10.4 36997.963 34516.83  155.4277197 1.0670769
## 10.6 35141.004 34115.04   99.7061007 1.0270720
## 10.8 21709.687 34904.08  168.6398573 0.6189907
## 11.0 29053.586 29964.51 -342.1816409 0.9808004
## 11.2 27677.416 30480.48 -256.3657315 0.9157394
## 11.4 31762.454 30581.55 -220.6222949 1.0461621
## 11.6 31193.651 29931.85 -263.5306364 1.0514129
## 11.8 14344.253 29421.67 -288.1955541 0.4923633
## 12.0 25355.913 25644.48 -637.0945312 1.0139369
## 12.2 24287.841 26630.58 -474.7750355 0.9285831
## 12.4 29091.223 28531.50 -237.2053622 1.0281656
## 12.6 28633.528 27797.06 -286.9293257 1.0408358
## 12.8 10873.548 26755.83 -362.3596216 0.4119788
## 13.0 24056.847 22827.97 -718.9094471 1.0880991
## 13.2 22465.282 22659.22 -663.8932545 1.0213661
## 13.4 22423.370 22864.50 -576.9759817 1.0060951
## 13.6 23647.592 23928.25 -412.9034165 1.0056238
## 13.8  8196.858 24772.76 -287.1623986 0.3347625
## 14.0 23478.325 21547.48 -580.9741839 1.1198017
## 14.2 21659.401 20871.61 -590.4638105 1.0679576
## 14.4 21930.343 20689.88 -549.5895491 1.0888789
## 14.6 21490.746 20642.95 -499.3239599 1.0668757
## 14.8  5757.885 20980.51 -415.6355931 0.2799864
## 15.0 42747.401 37141.48 1242.0245407 1.1136921
## 15.2 40678.528 36185.69 1022.2438487 1.0932755
## 15.4 38561.483 33733.69  674.8194806 1.1206960
## 15.6 35869.478 31682.98  402.2659399 1.1179431
## 15.8 12926.007 29882.72  182.0136223 0.4299392
## 16.0 37112.966 35244.07  699.9470112 1.0325214
## 16.2 32781.552 33734.09  478.9546724 0.9581594
## 16.4 33787.687 33237.94  381.4437174 1.0050062
## 16.6 30676.994 30075.98   27.1038771 1.0190647
## 16.8 14419.365 28630.56 -120.1486783 0.5057578
```

```r
xhat2 <- winter2$fitted[,2]

winter2.opt<- HoltWinters(traints3, alpha= NULL,  beta = NULL, gamma = NULL, seasonal = "multiplicative
winter2.opt$fitted
```

```
## Time Series:
## Start = c(2, 1)
## End = c(16, 5)
## Frequency = 5
```

```
##           xhat    level      trend    season
##   2.0 38769.765 27786.85  455.13714 1.3727703
##   2.2 24666.194 28349.79  476.84350 0.8556736
##   2.4 25931.732 28907.13  493.05091 0.8820262
##   2.6 24835.480 29418.74  496.78662 0.8301870
##   2.8 32159.540 29870.32  487.68569 1.0593429
##   3.0 43228.977 30054.84  426.64297 1.4182045
##   3.2 26666.856 30081.26  346.05901 0.8764115
##   3.4 27220.934 30360.70  332.64415 0.8868677
##   3.6 25322.954 30596.63  313.17069 0.8192533
##   3.8 30263.020 30988.24  328.96538 0.9663384
##   4.0 39065.731 30907.03  246.37607 1.2539795
##   4.2 26466.859 30643.45  143.69628 0.8596725
##   4.4 26539.061 30654.14  116.91531 0.8624684
##   4.6 25767.285 30676.29   97.83601 0.8373034
##   4.8 26295.873 30685.56   80.00147 0.8547179
##   5.0 33089.237 30774.87   81.87632 1.0723502
##   5.2 25495.789 30759.27   62.25074 0.8272073
##   5.4 26094.815 30994.74   97.12698 0.8392811
##   5.6 25447.327 31080.40   94.81962 0.8162676
##   5.8 26780.867 31159.12   91.57623 0.8569687
##   6.0 32667.647 31238.49   89.12013 1.0427748
##   6.2 27106.052 31187.37   60.88164 0.8674423
##   6.4 26275.820 31330.82   77.50637 0.8365878
##   6.6 26170.859 32008.23  198.29729 0.8125949
##   6.8 29079.168 33574.56  473.74952 0.8540562
##   7.0 34954.373 34351.20  534.73762 1.0019616
##   7.2 30992.091 34475.02  451.99915 0.8873385
##   7.4 34275.093 34798.12  426.04486 0.9730562
##   7.6 39115.331 35127.84  406.64923 1.1007710
##   7.8 32820.382 35325.28  364.52664 0.9196009
##   8.0 31400.375 34784.17  182.17689 0.8980170
##   8.2 30488.083 35261.98  241.70113 0.8587303
##   8.4 34064.580 35602.24  261.54708 0.9498322
##   8.6 37558.965 35738.55  236.33104 1.0440330
##   8.8 25599.260 35793.88  199.88517 0.7112138
##   9.0 33853.856 35112.32   22.40830 0.9635439
##   9.2 30932.151 35148.16   25.11121 0.8794221
##   9.4 32709.977 35441.82   79.18333 0.9208631
##   9.6 35943.476 35864.27  148.30137 0.9980812
##   9.8 20104.135 36020.29  149.85521 0.5558212
##  10.0 35884.436 36835.03  283.72793 0.9667468
##  10.2 34786.933 36874.69  234.58606 0.9374188
##  10.4 36754.172 36692.74  150.71647 0.9975767
##  10.6 36894.125 36761.95  134.30473 0.9999423
##  10.8 23857.016 36949.49  145.02460 0.6431413
##  11.0 33016.843 36262.66  -22.46890 0.9110559
##  11.2 30604.816 36248.21  -20.85465 0.8447985
##  11.4 35368.250 36181.83  -30.02069 0.9783258
##  11.6 36355.978 35973.65  -65.89314 1.0124825
##  11.8 18331.466 35717.70 -104.16208 0.5147331
##  12.0 31615.037 34885.59 -250.73262 0.9128097
##  12.2 28794.111 34699.16 -237.78625 0.8355475
##  12.4 32300.214 34694.00 -190.94747 0.9361554
```

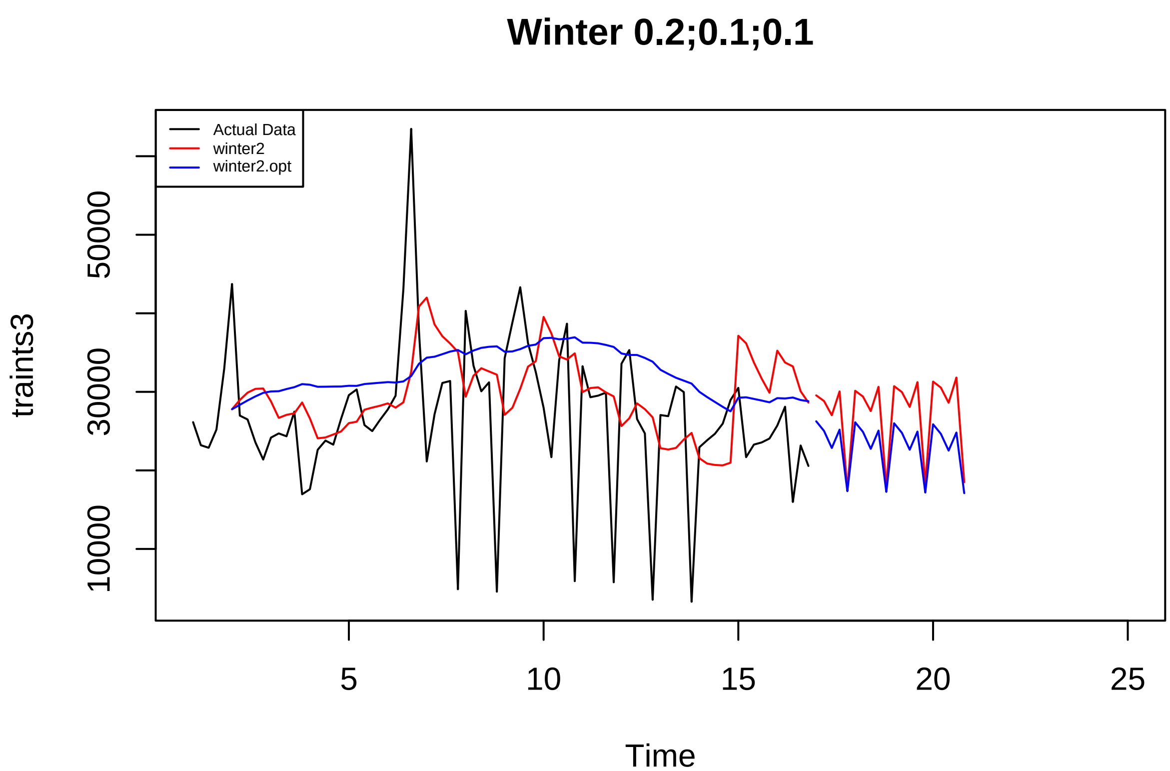
```
## 12.6 32918.603 34319.57 -227.89094 0.9655906
## 12.8 14136.933 33838.34 -278.90058 0.4212506
## 13.0 30032.741 32809.18 -429.96439 0.9275314
## 13.2 28151.202 32283.41 -449.25549 0.8843082
## 13.4 27968.943 31792.13 -457.71614 0.8925950
## 13.6 27969.531 31424.81 -439.51605 0.9026712
## 13.8 10333.218 31051.32 -426.22154 0.3374101
## 14.0 26610.744 30002.21 -551.63985 0.9035732
## 14.2 25134.718 29329.76 -575.96368 0.8741355
## 14.4 25732.633 28709.74 -584.83515 0.9149412
## 14.6 25281.384 28090.20 -591.82205 0.9193771
## 14.8  7445.312 27520.21 -587.42675 0.2764405
## 15.0 25378.822 29251.96 -120.46173 0.8711814
## 15.2 25202.630 29306.95  -85.13577 0.8624596
## 15.4 26239.666 29100.36 -109.58891 0.9051040
## 15.6 26627.999 28893.65 -129.14544 0.9257243
## 15.8 13322.892 28665.70 -149.03965 0.4671968
## 16.0 26754.960 29201.37  -11.17349 0.9165735
## 16.2 24217.518 29156.03  -18.05192 0.8311323
## 16.4 25732.719 29277.52   10.04293 0.8786229
## 16.6 25951.195 28956.92  -56.53102 0.8979531
## 16.8 16163.630 28808.24  -75.08439 0.5625427
```

```r
xhat2.opt <- winter2.opt$fitted[,2]
```

```r
forecast2 <- predict(winter2, n.ahead = 20)
forecast2.opt <- predict(winter2.opt, n.ahead = 20)
```

Peramalan

```r
plot(traints3,main="Winter 0.2;0.1;0.1",type="l",col="black",
     xlim=c(1,25),pch=12)
lines(xhat2,type="l",col="red")
lines(xhat2.opt,type="l",col="blue")
lines(forecast2,type="l",col="red")
lines(forecast2.opt,type="l",col="blue")
legend("topleft",c("Actual Data",expression(paste(winter2)),
                   expression(paste(winter2.opt))),cex=0.5,
       col=c("black","red","blue"),lty=1)
```



Plot DW

```
SSE2<-winter2$SSE
MSE2<-winter2$SSE/length(traints3)
RMSE2<-sqrt(MSE2)
akurasi1 <- matrix(c(SSE2,MSE2,RMSE2))
row.names(akurasi1)<- c("SSE2", "MSE2", "RMSE2")
colnames(akurasi1) <- c("Akurasi lamda=0.2")
akurasi1
```

Akurasi Data Latih

```
##        Akurasi lamda=0.2
## SSE2       8.749204e+09
## MSE2       1.093651e+08
## RMSE2      1.045777e+04
```

```
SSE2.opt<-winter2.opt$SSE
MSE2.opt<-winter2.opt$SSE/length(traints3)
RMSE2.opt<-sqrt(MSE2.opt)
akurasi1.opt <- matrix(c(SSE2.opt,MSE2.opt,RMSE2.opt))
row.names(akurasi1.opt)<- c("SSE2.opt", "MSE2.opt", "RMSE2.opt")
colnames(akurasi1.opt) <- c("Akurasi")
akurasi1.opt
```

```
##                  Akurasi
## SSE2.opt   6.908641e+09
## MSE2.opt   8.635801e+07
```

```
## RMSE2.opt 9.292901e+03
```

```
akurasi2.train = data.frame(Model_Winter = c("Winter 2","winter2 optimal"),
                            Nilai_SSE=c(SSE2,SSE2.opt),
                            Nilai_MSE=c(MSE2,MSE2.opt),Nilai_RMSE=c(RMSE2,RMSE2.opt))
akurasi2.train
```

```
##      Model_Winter  Nilai_SSE Nilai_MSE Nilai_RMSE
## 1        Winter 2 8749204235 109365053  10457.775
## 2 winter2 optimal 6908640767  86358010   9292.901
```

Pada data latih, metode Winter Multiplikatif dengan parameter optimal menghasilkan nilai RMSE lebih kecil, sehingga dapat dikatakan bahwa metode Winter Multiplikatif adalah metode yang lebih baik.

```

forecast2<-data.frame(forecast2)
ujits3df<-data.frame(ujits3)
selisih2<-forecast2-ujits3df
SSEtesting2<-sum(selisih2^2)
MSEtesting2<-SSEtesting2/length(ujits3df)
RMSEtesting2<-sqrt(MSEtesting2)


forecast2.opt<-data.frame(forecast2.opt)
selisih2.opt<-forecast2.opt-ujits3df
SSEtesting2.opt<-sum(selisih2.opt^2)
MSEtesting2.opt<-SSEtesting2.opt/length(ujits3df)
RMSEtesting2.opt<-sqrt(MSEtesting2.opt)


akurasiwin2.uji<-data.frame("Nilai RMSE",RMSEtesting2,RMSEtesting2.opt)
akurasiwin2.uji
```

Akurasi Data Uji

```
##   X.Nilai.RMSE. RMSEtesting2 RMSEtesting2.opt
## 1    Nilai RMSE     36099.67         25345.85
```

Pada data uji, metode Winter Multiplikatif dengan parameter optimal menghasilkan nilai RMSE lebih kecil, sehingga dapat disimpulkan bahwa metode Winter Multiplikatif merupakan metode yang lebih baik.

Nilai Akurasi RMSE data latih metode SMA dan DMA

```
akurasi_train.sma
```

```
##      Akurasi m = 4
## SSE   8.960772e+09
## MSE   1.179049e+08
## MAPE  6.551244e+01
```

```
RMSE.dmalat<-sqrt(MSE_train.dma)
RMSE.dmalat
```

```
## [1] 13758.35
```

Nilai Akurasi data uji metode SMA dan DMA

```r
RMSE.smauji<-sqrt(MSE_uji.sma)
RMSE.smauji
```

```
## [1] 5923.635
```

```r
RMSE.dmauji<-sqrt(MSE_test.dma)
RMSE.dmauji
```

```
## [1] 14799.28
```

Nilai Akurasi SES dan DES

```r
accuracy(ramalanopt,test2$Jumlah)
```

```
##                      ME     RMSE      MAE       MPE     MAPE      MASE     ACF1
## Training set   658.5725 9424.135 6377.459 -33.85073 52.65036 0.8107900 0.141926
## Test set     -4236.9227 7282.752 5307.616 -32.27697 35.86079 0.6747769       NA
```

```r
accuracy(ramalandesopt,test2$Jumlah)
```

```
##                     ME      RMSE      MAE        MPE     MAPE      MASE
## Training set 1071.5242 10567.693 7371.936 -35.466737 60.87888 0.9372216
## Test set      362.3212  5958.374 4733.581  -8.901525 27.70679 0.6017977
##                   ACF1
## Training set 0.1260325
## Test set            NA
```

Nilai Akurasi Winter aditif

```r
akurasi1.train
```

```
##   Model_Winter.Latih  Nilai_SSE Nilai_MSE Nilai_RMSE
## 1           Winter 1 8435199678 105439996  10268.398
## 2    Winter1 optimal 6916438840  86455485   9298.144
```

```r
akurasi1.uji
```

```
##   X.Nilai.RMSE.Uji. RMSEtesting1 RMSEtesting1.opt
## 1    Nilai RMSE Uji     26808.85         25251.44
```

Nilai Akurasi Winter multiplikatif

```r
akurasi2.train
```

```
##      Model_Winter  Nilai_SSE Nilai_MSE Nilai_RMSE
## 1        Winter 2 8749204235 109365053  10457.775
## 2 winter2 optimal 6908640767  86358010   9292.901
```

```r
akurasiwin2.uji
```

```
##   X.Nilai.RMSE. RMSEtesting2 RMSEtesting2.opt
## 1    Nilai RMSE     36099.67         25345.85
```

Kesimpulan

Metode yang tepat digunakan adalah metode Winter, karena plot awal data menggambarkan bentuk musiman. Secara umum, data yang digunakan memiliki puncak(atas dan bawah) yang relatif sama sehingga lebih tepat menggunakan metode Winter Aditif. Selanjutnya nilai akurasi metode Winter Aditif pada data uji menghasilkan RMSE sebesar 26808.85 untuk parameter alpha=0.2, beta=0.1, gamma=0.1. Sedangkan jika menggunakan parameter optimal menghasilkan nilai RMSE sebesar 25251.44.

Sehingga metode pemulusan yang paling tepat digunakan adalah metode Winter Aditif dengan parameter optimal.