153_project

2022-04-05

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(TSA)
```

```
##
## Attaching package: 'TSA'

## The following object is masked from 'package:readr':
##
##     spec

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

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

```r
library(forecast)
```

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

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

```r
library(astsa)
```

```
##
## Attaching package: 'astsa'

## The following object is masked from 'package:forecast':
##
##     gas
```

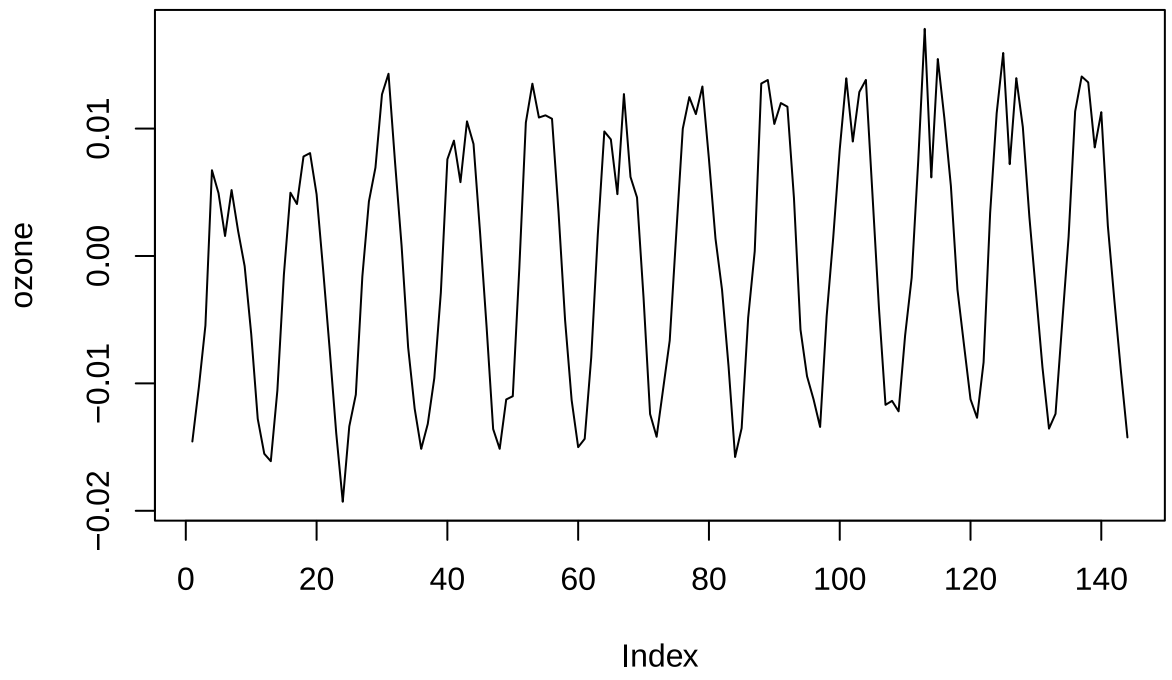
```r
source('cleaning.R')
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

## New names:
## * `` -> ...1

## Rows: 1746661 Columns: 29
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (8): Address, State, County, City, NO2 Units, O3 Units, SO2 Units, CO ...
## dbl  (20): ...1, State Code, County Code, Site Num, NO2 Mean, NO2 1st Max Va...
## date  (1): Date Local
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
## `summarise()` has grouped output by 'year'. You can override using the `.groups` argument.
```

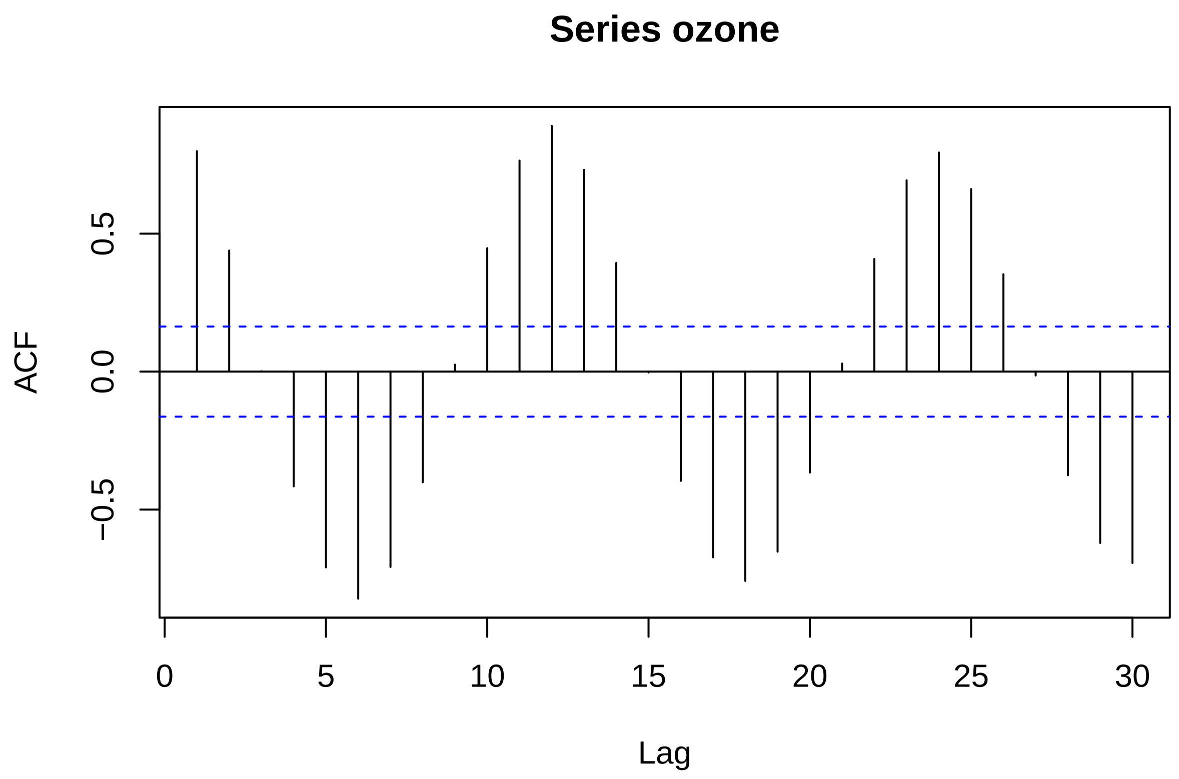
Original data

1. The original data have clear sign of seasonality, but there seems to be linear upward trend.
2. The ACF PACF plot shows the strong sign of seasonality

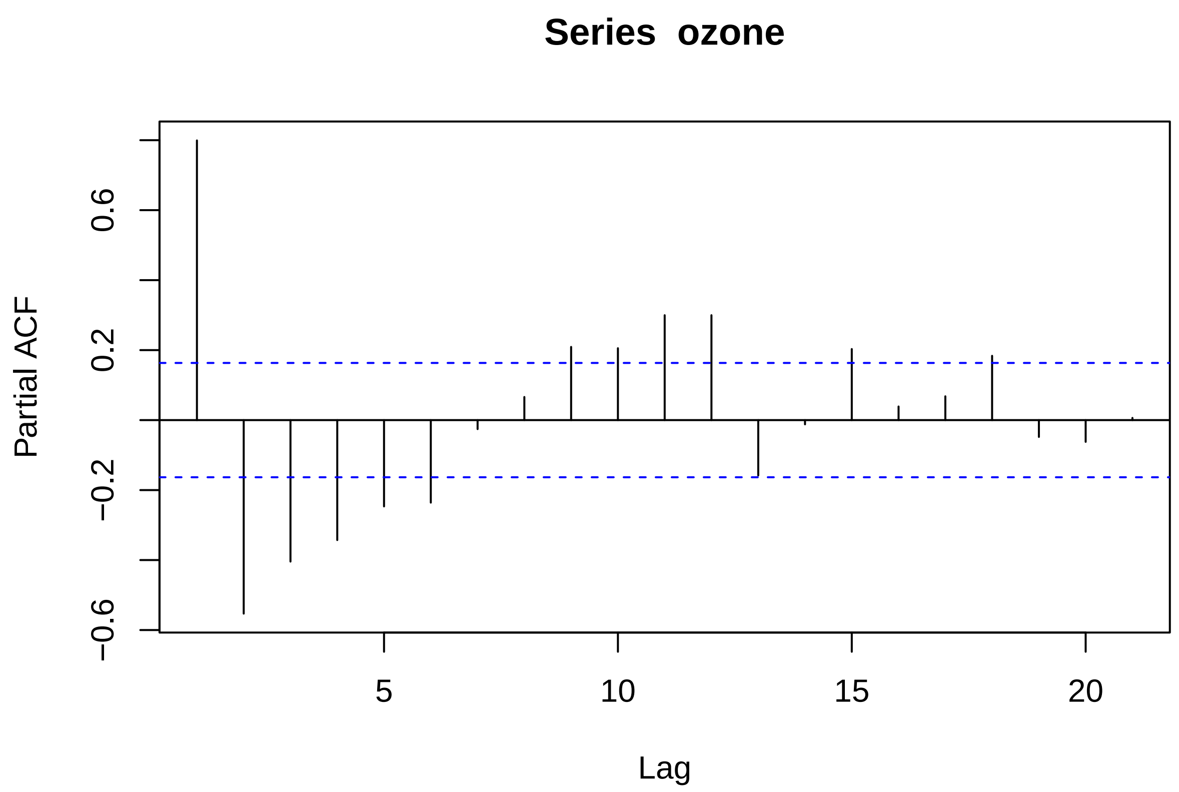
```r
ozone <- phoenix$o3
ozone = ozone - mean(ozone) # mean centered
plot(ozone,type ="l")
```



```
acf(ozone,lag.max = 30)
```



```
pacf(ozone)
```



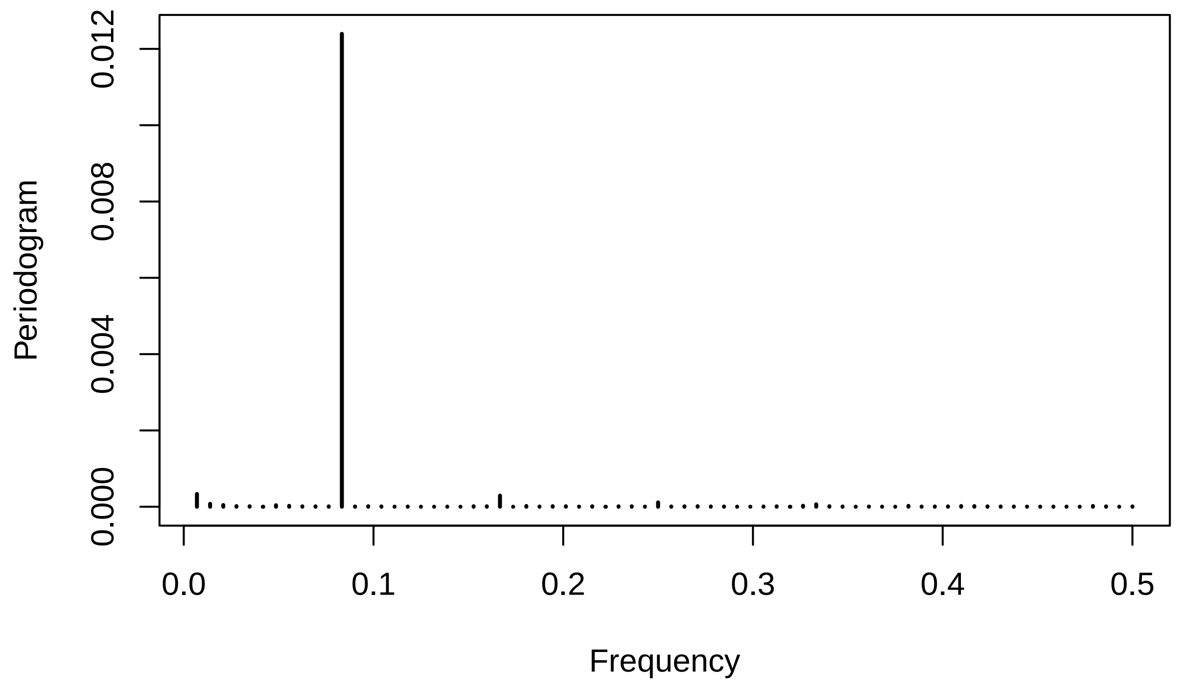
Sinusoidal fitting

Vt = ozone (1+B)Vt = Xt (Vt - f(t)) = Xt

$f(t) = -0.003 + 0.000042 * t - 0.0012 * sin(t) - 0.0123 * cos(t) + 0.000002 * t * sin(t) - 0.000011 * t * cos(t)$

1. There is one significant peak in the periodogram 2. The residual plot shows some seasonality but the plot seems to be AR process and possible seasonal ARMA

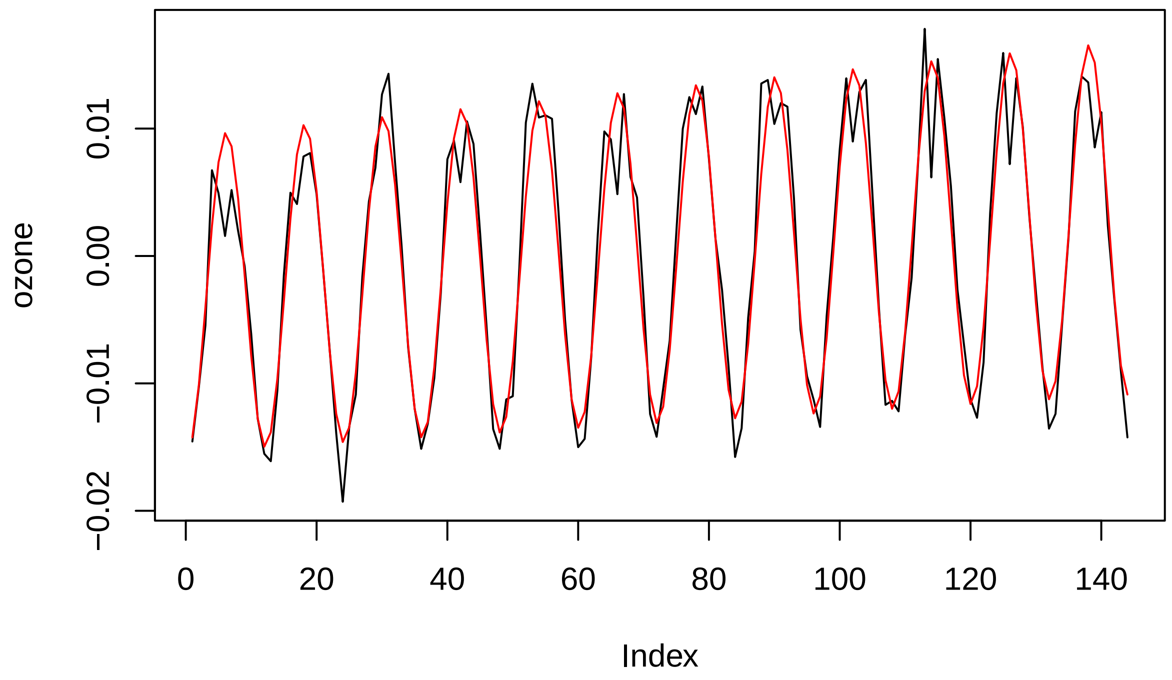
```
t = 1:length(ozone)
# Check the periodogram
periodo = periodogram(ozone,plot=TRUE,ylab="Periodogram", xlab="Frequency") # There is one significant
```



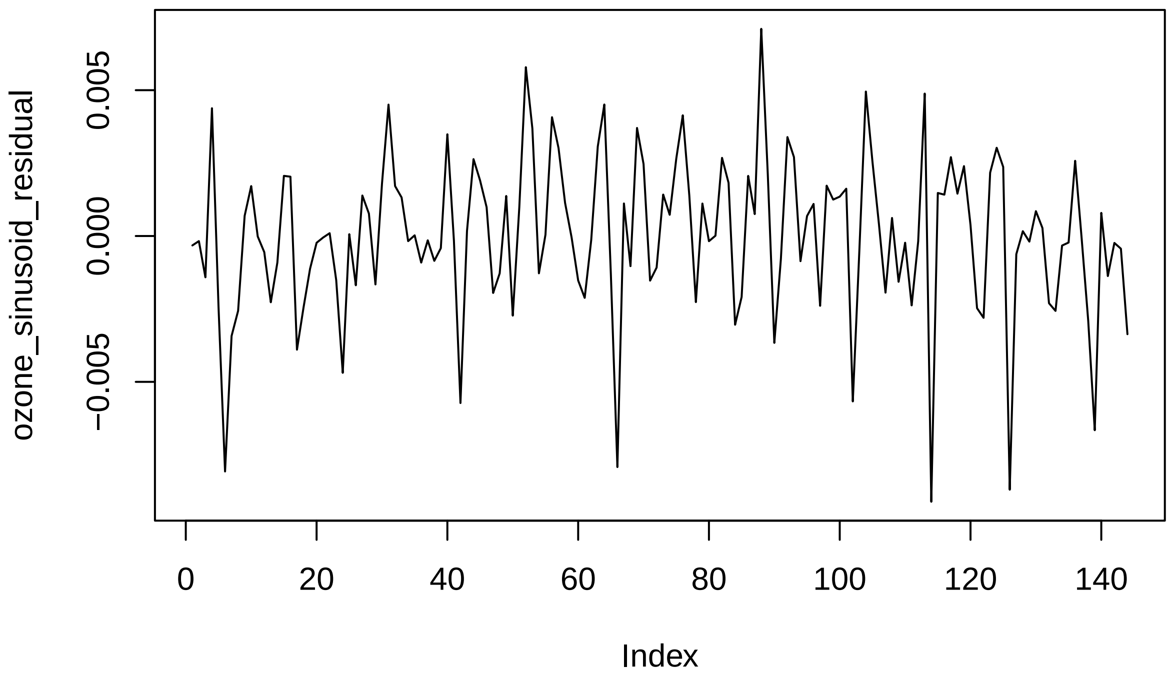
```r
# Get the high magnitudes in descending order
order_spec = sort(periodo$spec,decreasing = TRUE)

# Get the frequency that gives max magnitude
first_max = order_spec[1]
first_maximizing_freq = periodo$freq[periodo$spec==first_max]
first_sin_max = sin(2*pi*first_maximizing_freq*t)
first_cos_max = cos(2*pi*first_maximizing_freq*t)

# Max Sinusoidal fitting
ozone_sinusoid_model = lm(ozone ~ first_sin_max*(1+t)+first_cos_max*(1+t))
print(ozone_sinusoid_model$coefficients)
```

```
##     (Intercept)   first_sin_max               t   first_cos_max first_sin_max:t
##   -3.009489e-03   -1.160248e-03    4.163369e-05   -1.233604e-02    1.963185e-06
## t:first_cos_max
##   -1.058031e-05
```

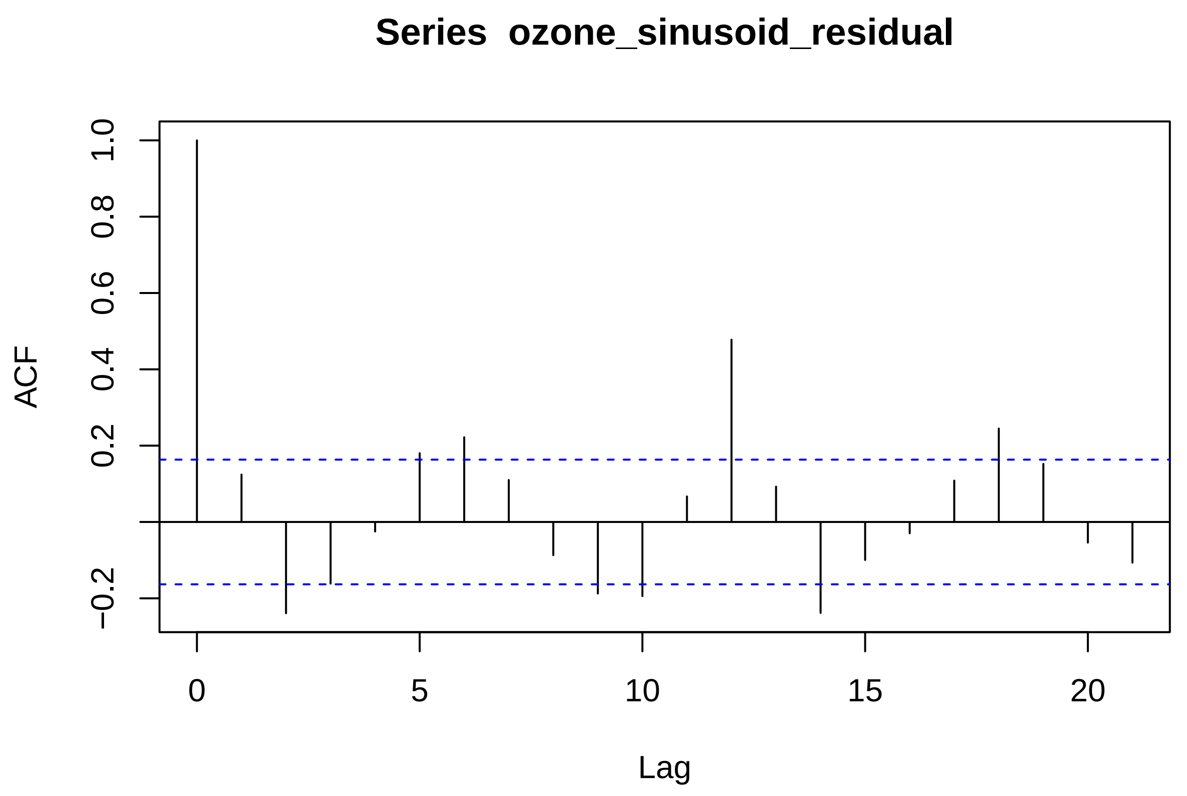
```r
# Overlay the sinusoidal fitting over the original plot
plot(ozone,type = "l")
lines(t,ozone_sinusoid_model$fitted.values,col = "red")
```



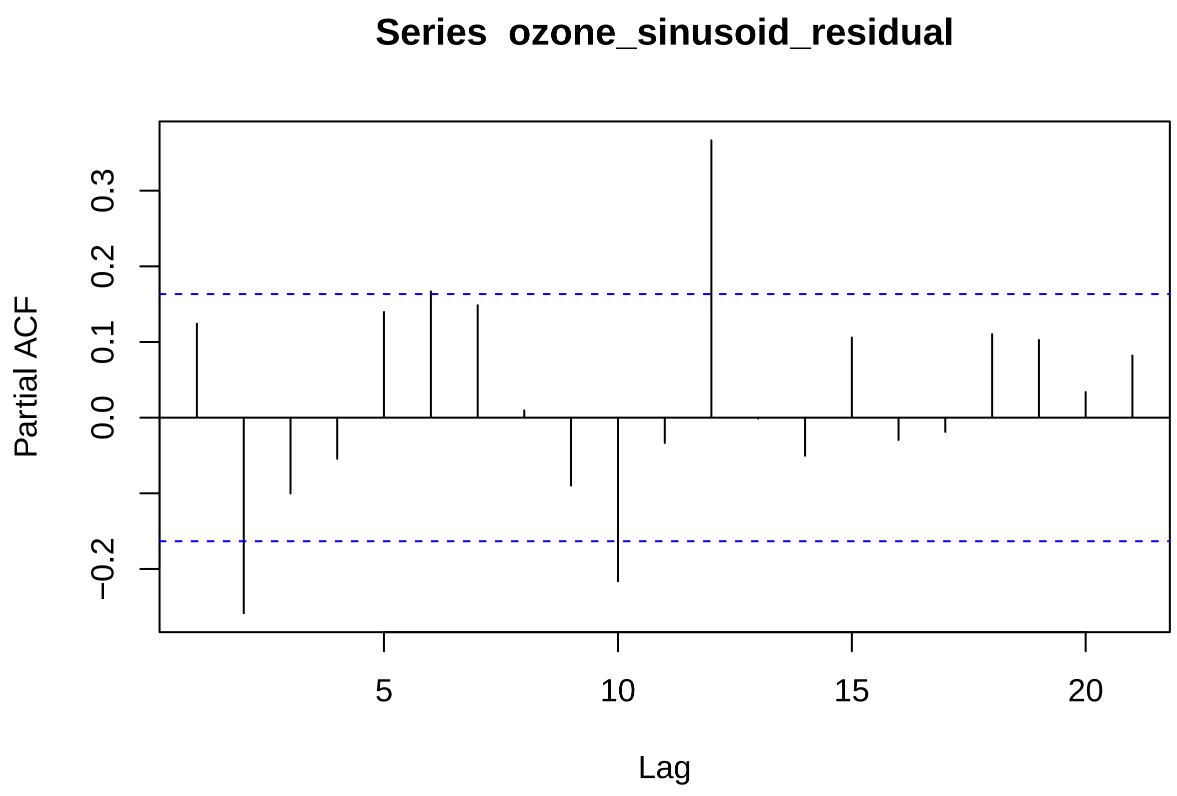
```r
# Get the residual, hoping for removing seasonality
ozone_sinusoid_residual = ozone_sinusoid_model$residuals
plot(ozone_sinusoid_residual,type = "l") # residual seems to be stationary
```



```
stats::acf(ozone_sinusoid_residual)
```



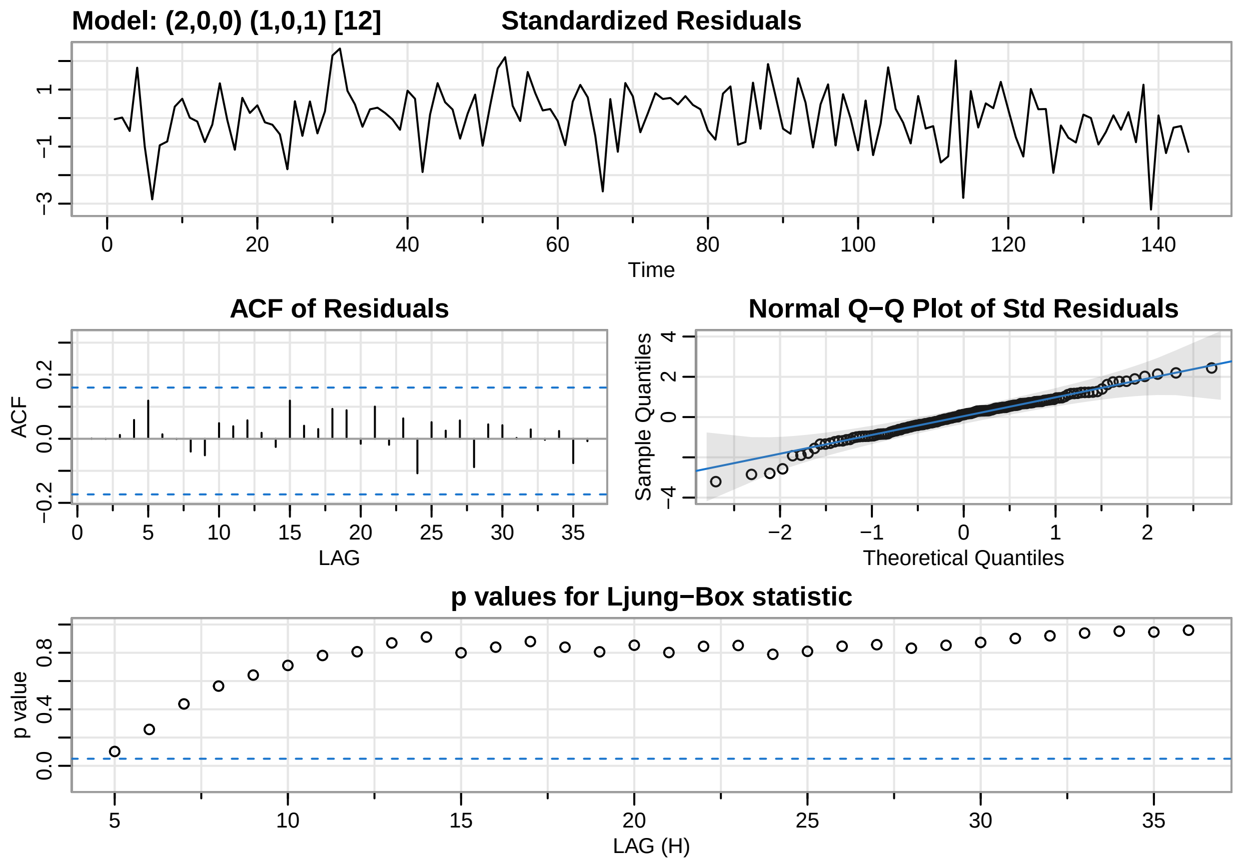
```
stats::pacf(ozone_sinusoid_residual)
```



Model1 : SARIMA(2,0,0)(1,0,1)12

```
model1 <- sarima(ozone_sinusoid_residual, p=2, d=0, q=0, P=1, D=0, Q=1, S=12) # fit the model
```

```
## initial  value -5.921137
## iter   2 value -6.020457
## iter   3 value -6.079641
## iter   4 value -6.085871
## iter   5 value -6.091190
## iter   6 value -6.095582
## iter   7 value -6.095958
## iter   8 value -6.095969
## iter   9 value -6.095970
## iter   9 value -6.095970
## iter   9 value -6.095970
## final  value -6.095970
## converged
## initial  value -6.075290
## iter   2 value -6.078754
## iter   3 value -6.081474
## iter   4 value -6.085882
## iter   5 value -6.088012
## iter   6 value -6.092257
## iter   7 value -6.095296
## iter   8 value -6.097651
## iter   9 value -6.099009
## iter  10 value -6.100678
## iter  11 value -6.101042
## iter  12 value -6.101155
## iter  13 value -6.101161
## iter  14 value -6.101163
## iter  15 value -6.101163
## iter  16 value -6.101166
## iter  17 value -6.101167
## iter  18 value -6.101167
## iter  19 value -6.101168
## iter  20 value -6.101168
## iter  21 value -6.101169
## iter  22 value -6.101169
## iter  22 value -6.101169
## iter  22 value -6.101169
## final  value -6.101169
## converged
```



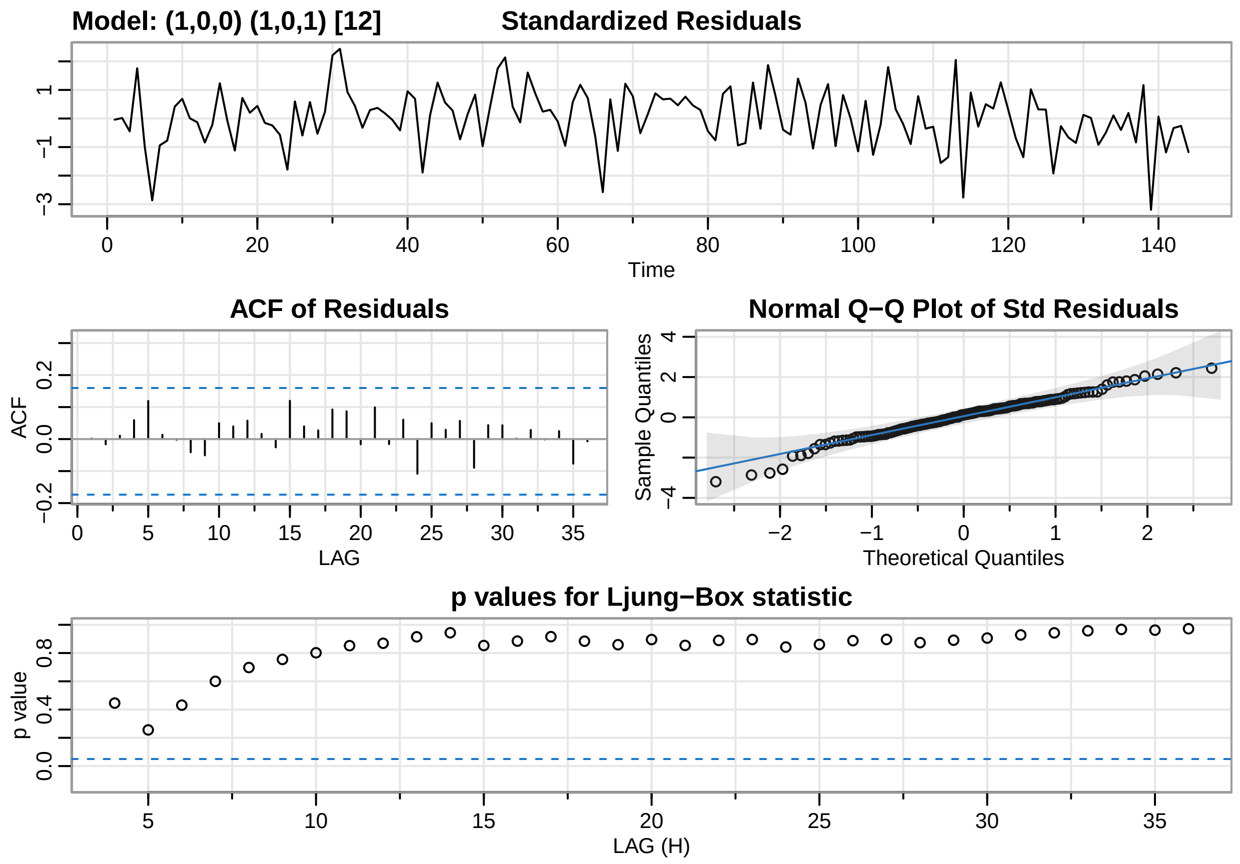
```
coeff_table <- as.data.frame(model1$ttable)
coeff_table <-coeff_table %>% mutate(ci_lower = Estimate-1.96*SE,ci_upper =  Estimate+1.96*SE)
coeff_table # show estimated coefficient and its ci
```

```
##       Estimate     SE t.value p.value  ci_lower  ci_upper
## ar1     0.0903 0.0837  1.0792  0.2824 -0.073752  0.254352
## ar2    -0.0160 0.0881 -0.1810  0.8566 -0.188676  0.156676
## sar1    0.9462 0.0545 17.3486  0.0000  0.839380  1.053020
## sma1   -0.7176 0.1439 -4.9858  0.0000 -0.999644 -0.435556
## xmean  -0.0002 0.0006 -0.3728  0.7099 -0.001376  0.000976
```

Model2: SARIMA(1,0,0)(1,0,1)12

```
model2 <- sarima(ozone_sinusoid_residual, p=1, d=0, q=0, P=1, D=0, Q=1, S=12) # fit the model
```

```
## initial  value -5.924525
## iter   2 value -6.014280
## iter   3 value -6.072474
## iter   4 value -6.080884
## iter   5 value -6.087853
## iter   6 value -6.094164
## iter   7 value -6.094618
## iter   8 value -6.094677
## iter   9 value -6.094683
## iter  10 value -6.094683
## iter  11 value -6.094684
## iter  12 value -6.094684
## iter  12 value -6.094684
## iter  12 value -6.094684
## final  value -6.094684
## converged
## initial  value -6.075303
## iter   2 value -6.078312
## iter   3 value -6.083138
## iter   4 value -6.089196
## iter   5 value -6.092910
## iter   6 value -6.095028
## iter   7 value -6.096554
## iter   8 value -6.098751
## iter   9 value -6.100853
## iter  10 value -6.101030
## iter  11 value -6.101038
## iter  12 value -6.101038
## iter  13 value -6.101041
## iter  14 value -6.101048
## iter  15 value -6.101050
## iter  16 value -6.101052
## iter  17 value -6.101053
## iter  18 value -6.101054
## iter  19 value -6.101055
## iter  20 value -6.101055
## iter  20 value -6.101055
## iter  20 value -6.101055
## final  value -6.101055
## converged
```



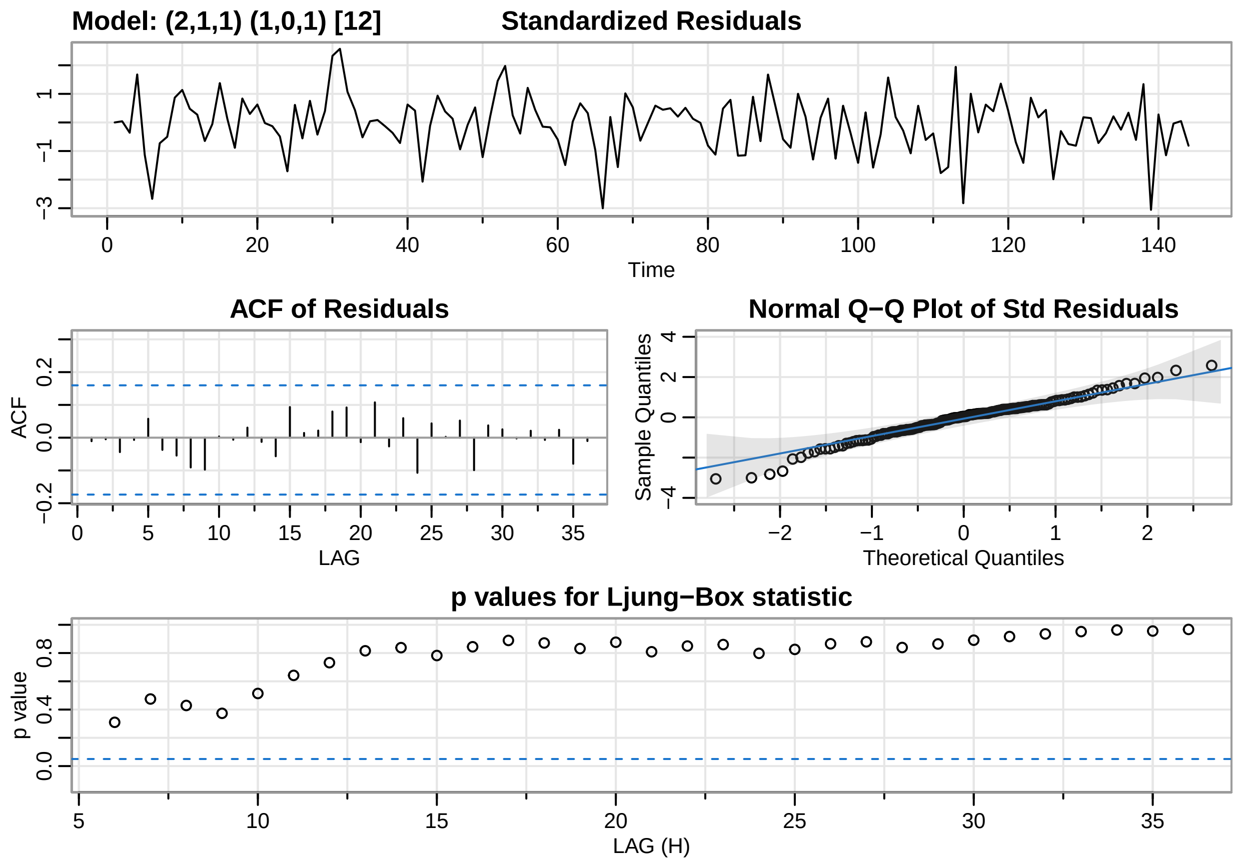
```
coeff_table <- as.data.frame(model2$ttable)
coeff_table <-coeff_table %>% mutate(ci_lower = Estimate-1.96*SE,ci_upper =  Estimate+1.96*SE)
coeff_table # show estimated coefficient and its ci
```

```
##        Estimate     SE t.value p.value  ci_lower  ci_upper
## ar1      0.0888 0.0832  1.0667  0.2879 -0.074272  0.251872
## sar1     0.9489 0.0498 19.0430  0.0000  0.851292  1.046508
## sma1    -0.7226 0.1379 -5.2393  0.0000 -0.992884 -0.452316
## xmean   -0.0002 0.0006 -0.3640  0.7164 -0.001376  0.000976
```

# Model3: SARIMA(2,1,1)(1,0,1)12

```
model3 <- sarima(ozone_sinusoid_residual, p=2, d=1, q=1, P=1, D=0, Q=1, S=12) # fit the model
```

```
## initial  value -5.626415
## iter   2 value -5.817904
## iter   3 value -5.967005
## iter   4 value -6.000235
## iter   5 value -6.022716
## iter   6 value -6.038025
## iter   7 value -6.061036
## iter   8 value -6.062258
## iter   9 value -6.063975
## iter  10 value -6.065519
## iter  11 value -6.066125
## iter  12 value -6.066443
## iter  13 value -6.067328
## iter  14 value -6.067686
## iter  15 value -6.067780
## iter  16 value -6.067831
## iter  17 value -6.067832
## iter  18 value -6.067883
## iter  19 value -6.067920
## iter  20 value -6.067946
## iter  21 value -6.067967
## iter  22 value -6.068029
## iter  23 value -6.068059
## iter  24 value -6.068062
## iter  25 value -6.068063
## iter  25 value -6.068063
## iter  25 value -6.068063
## final  value -6.068063
## converged
## initial  value -6.049027
## iter   2 value -6.063938
## iter   3 value -6.069719
## iter   4 value -6.077660
## iter   5 value -6.081289
## iter   6 value -6.083983
## iter   7 value -6.086264
## iter   8 value -6.089727
## iter   9 value -6.093924
## iter  10 value -6.097960
## iter  11 value -6.098795
## iter  12 value -6.099032
## iter  13 value -6.099192
## iter  14 value -6.099209
## iter  15 value -6.099215
## iter  16 value -6.099216
## iter  17 value -6.099217
## iter  18 value -6.099217
## iter  19 value -6.099217
## iter  20 value -6.099218
## iter  21 value -6.099219
## iter  22 value -6.099220
## iter  23 value -6.099220
## iter  23 value -6.099220
## iter  23 value -6.099220
## final  value -6.099220
## converged
```



```
coeff_table <- as.data.frame(model3$ttable)
coeff_table <-coeff_table %>% mutate(ci_lower = Estimate-1.96*SE,ci_upper =  Estimate+1.96*SE)
coeff_table # show estimated coefficient and its ci
```

```
##           Estimate     SE  t.value p.value  ci_lower  ci_upper
## ar1         0.0192 0.0892   0.2150  0.8301 -0.155632  0.194032
## ar2        -0.0791 0.0894  -0.8846  0.3779 -0.254324  0.096124
## ma1        -0.9352 0.0317 -29.5144  0.0000 -0.997332 -0.873068
## sar1        0.9602 0.0366  26.2225  0.0000  0.888464  1.031936
## sma1       -0.7544 0.1106  -6.8202  0.0000 -0.971176 -0.537624
## constant    0.0000 0.0001  -0.0145  0.9884 -0.000196  0.000196
```

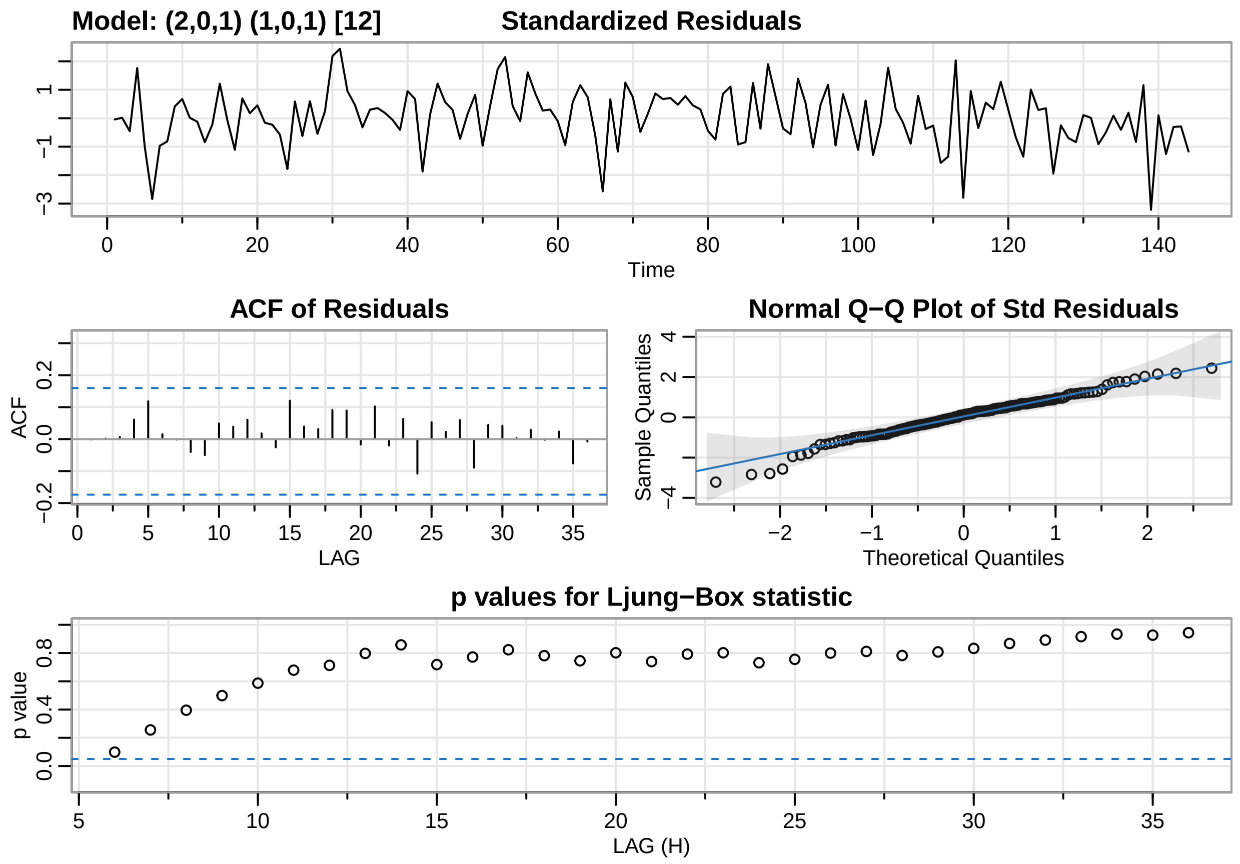
# Model4: SARIMA(2,0,1)(1,0,1)12

```
model4 <- sarima(ozone_sinusoid_residual, p=2, d=0, q=1, P=1, D=0, Q=1, S=12) # fit the model
```

```
## initial  value -5.921137
## iter   2 value -6.021739
## iter   3 value -6.079075
## iter   4 value -6.085438
## iter   5 value -6.091319
## iter   6 value -6.095255
## iter   7 value -6.095886
## iter   8 value -6.095982
## iter   9 value -6.095984
## iter  10 value -6.095987
## iter  11 value -6.095993
## iter  12 value -6.096006
## iter  13 value -6.096024
## iter  14 value -6.096036
## iter  15 value -6.096039
## iter  16 value -6.096040
## iter  17 value -6.096040
## iter  18 value -6.096040
## iter  19 value -6.096040
## iter  19 value -6.096040
## iter  19 value -6.096040
## final  value -6.096040
## converged
## initial  value -6.075372
## iter   2 value -6.077112
## iter   3 value -6.080624
## iter   4 value -6.083981
## iter   5 value -6.086061
## iter   6 value -6.091109
## iter   7 value -6.094569
## iter   8 value -6.096817
## iter   9 value -6.097887
## iter  10 value -6.099778
## iter  11 value -6.100684
## iter  12 value -6.101115
## iter  13 value -6.101118
## iter  14 value -6.101118
## iter  15 value -6.101118
## iter  16 value -6.101124
## iter  17 value -6.101136
## iter  18 value -6.101149
## iter  19 value -6.101171
## iter  20 value -6.101187
## iter  21 value -6.101190
## iter  22 value -6.101192
## iter  23 value -6.101197
## iter  24 value -6.101201
## iter  25 value -6.101209
## iter  26 value -6.101219
## iter  27 value -6.101230
## iter  28 value -6.101233
## iter  29 value -6.101237
## iter  29 value -6.101237
## iter  29 value -6.101237
## final  value -6.101237
## converged
```

```
## Warning in sqrt(diag(fitit$var.coef)): NaNs produced
```

```
## Warning in sqrt(diag(fitit$var.coef)): NaNs produced
```



```
coeff_table <- as.data.frame(model4$ttable)
coeff_table <-coeff_table %>% mutate(ci_lower = Estimate-1.96*SE,ci_upper =  Estimate+1.96*SE)
coeff_table # show estimated coefficient and its ci
```

```
##        Estimate     SE t.value p.value  ci_lower  ci_upper
## ar1     -0.3458    NaN     NaN     NaN       NaN       NaN
## ar2      0.0214    NaN     NaN     NaN       NaN       NaN
## ma1      0.4374    NaN     NaN     NaN       NaN       NaN
## sar1     0.9498 0.0488 19.4654  0.0000  0.854152  1.045448
## sma1    -0.7278 0.1344 -5.4138  0.0000 -0.991224 -0.464376
## xmean   -0.0002 0.0006 -0.3604  0.7191 -0.001376  0.000976
```

Evaluation Matrix

1. The first two models, SARIMA(2,0,0)(1,0,1)12 , SARIMA(1,0,0)(1,0,1)12 gives the lowest AIC,AICc,BIC, hence theses models are the potential best two parametric models

```
# AIC, AICc, BIC
eval<- function(model){
  return (c(model$AIC, model$AICc,model$BIC))
}

m1_evaludation = eval(model1)
```

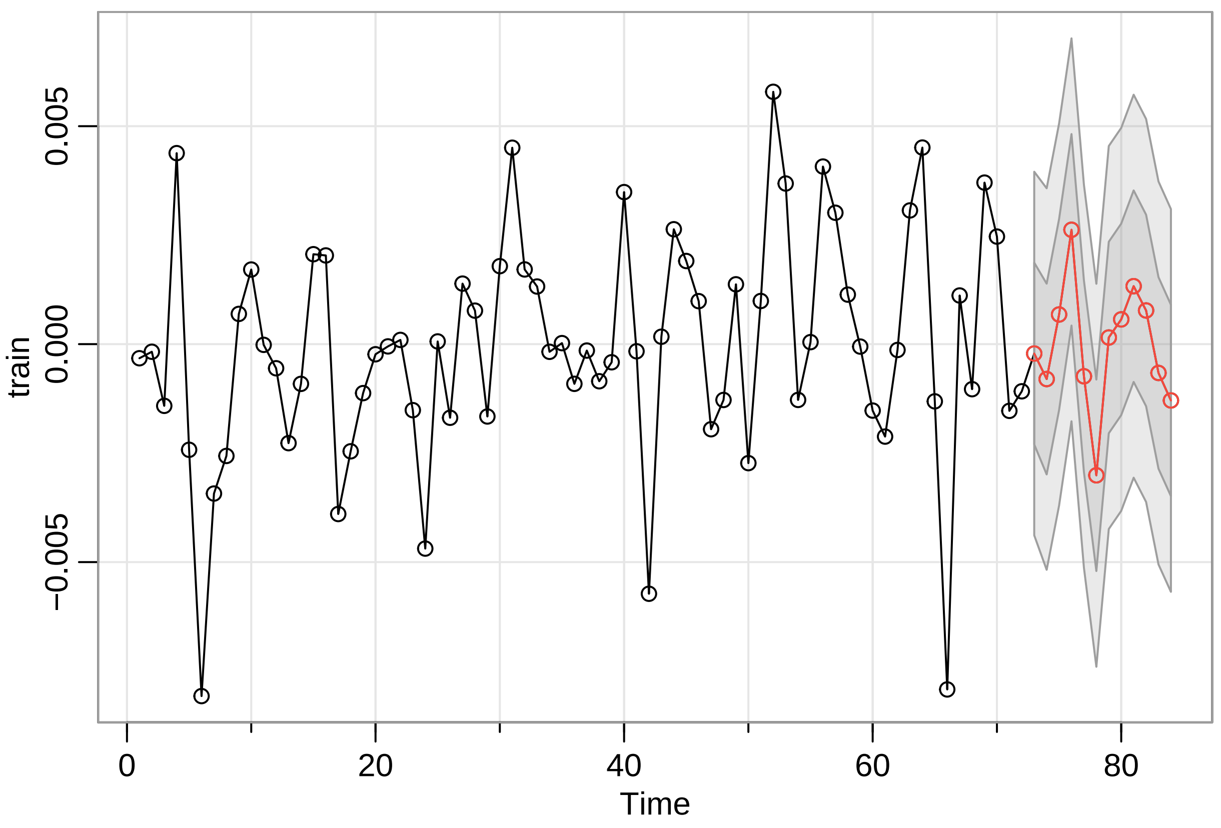
```r
m2_evaludation = eval(model2)
m3_evaludation = eval(model3)
m4_evaludation = eval(model4)

eval_matrix = rbind(m1_evaludation,m2_evaludation,m3_evaludation,m4_evaludation)
rownames(eval_matrix) = c("SARIMA(2,0,0)(1,0,1)12","SARIMA(1,0,0)(1,0,1)12",
                          "SARIMA(2,1,1)(1,0,1)12 ","SARIMA(2,0,1)(1,0,1)12 ")
colnames(eval_matrix) = c("AIC","AICc","BIC")
eval_matrix
```

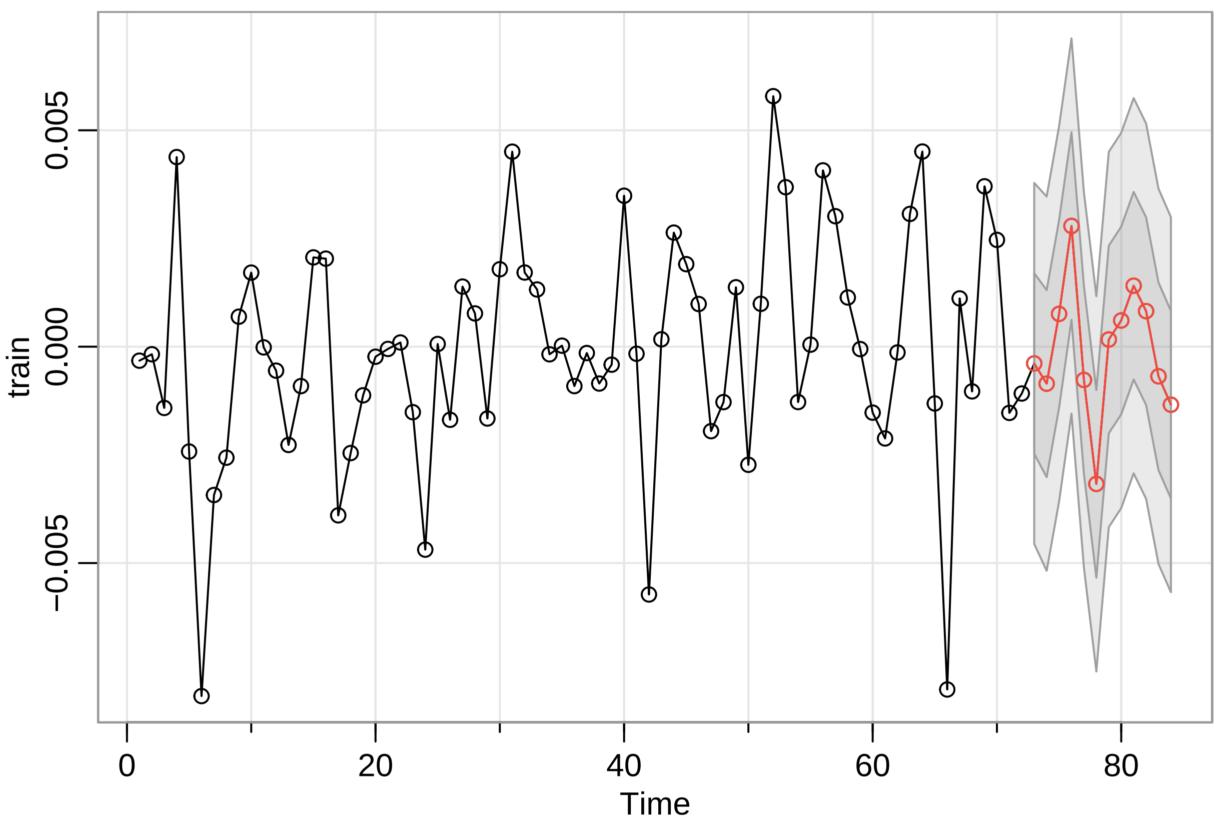
```
##                                 AIC      AICc       BIC
## SARIMA(2,0,0)(1,0,1)12  -9.281128 -9.278109 -9.157386
## SARIMA(1,0,0)(1,0,1)12  -9.294789 -9.292790 -9.191670
## SARIMA(2,1,1)(1,0,1)12  -9.262661 -9.258342 -9.117626
## SARIMA(2,0,1)(1,0,1)12  -9.267376 -9.263118 -9.123010
```

Cross Validation

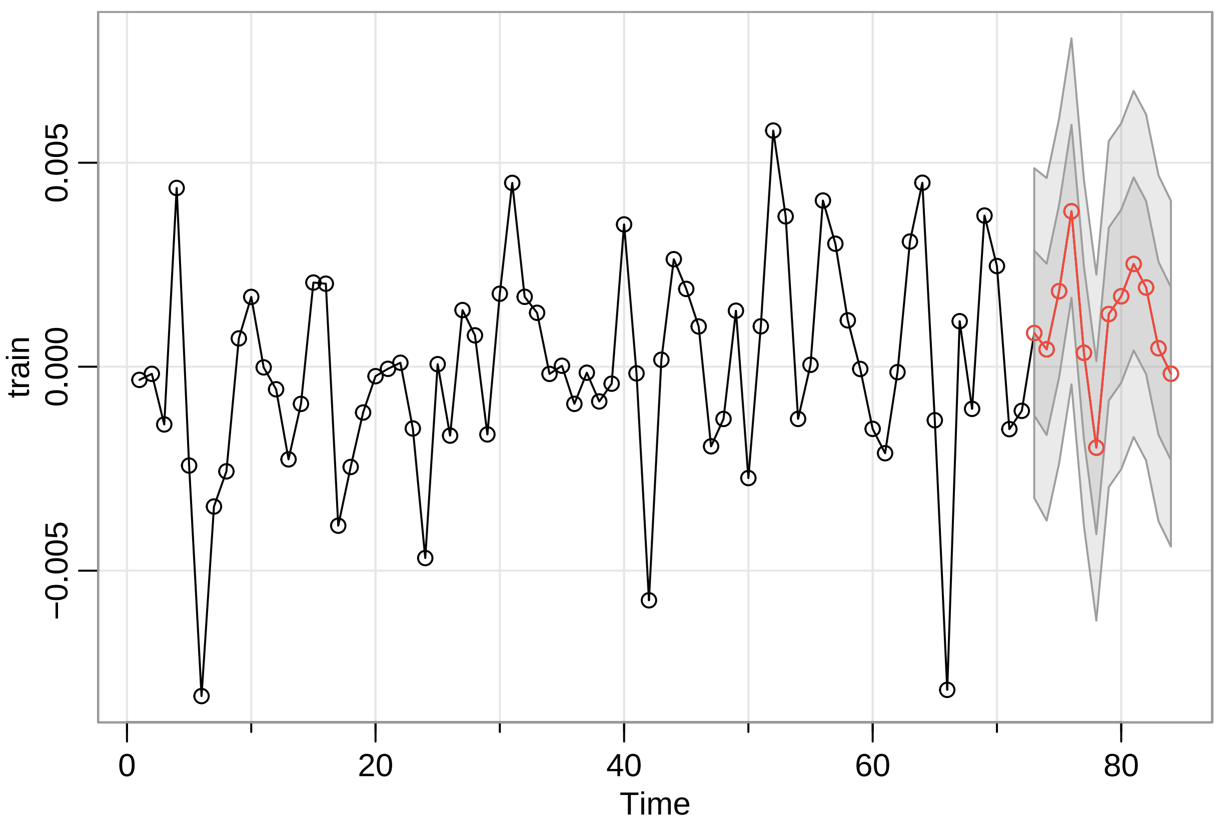
1. To determine best two models, use crovalidations and find two models that gives the lowest SSEs. Train : 2004 ~ 2011 Test : 2012 - 2015
2. The first two models SARIMA(2,0,0)(1,0,1)12 , SARIMA(1,0,0)(1,0,1)12 gives the lowest SSEs.
3. Overall, the first two models gives the lowest values on both (AIC,AICc,BIC) and the SSE. Hence, the best two parametric models are SARIMA(2,0,0)(1,0,1)12, SARIMA(1,0,0)(1,0,1)12
4. The SSE for both model are low, but SARIMA(2,0,0)(1,0,1)12 is slighly better

```r
sse1 = c()
sse2 = c()
sse3 = c()
sse4 = c()
test_years = seq(10,15,1)
for (year in test_years) {

train_index = 1:(12*(year-4))
test_index = (12*(year-4)+1):(12*(year-4+1))

train <- ozone_sinusoid_residual[train_index]
test <- ozone_sinusoid_residual[test_index]
m1_forecast <- sarima.for(train, p=2, d=0, q=0, P=1, D=0, Q=1, S=12, n.ahead=12)$pred
m2_forecast <- sarima.for(train, p=1, d=0, q=0, P=1, D=0, Q=1, S=12, n.ahead=12)$pred
m3_forecast <- sarima.for(train, p=2, d=1, q=1, P=1, D=0, Q=1, S=12, n.ahead=12)$pred
m4_forecast <- sarima.for(train, p=2, d=0, q=1, P=1, D=0, Q=1, S=12, n.ahead=12)$pred

sse1 = c(sse1,sum((m1_forecast - test)^2))
sse2 = c(sse2,sum((m2_forecast - test)^2))
sse3 = c(sse3,sum((m3_forecast - test)^2))
sse4 = c(sse4,sum((m4_forecast - test)^2))
}
```

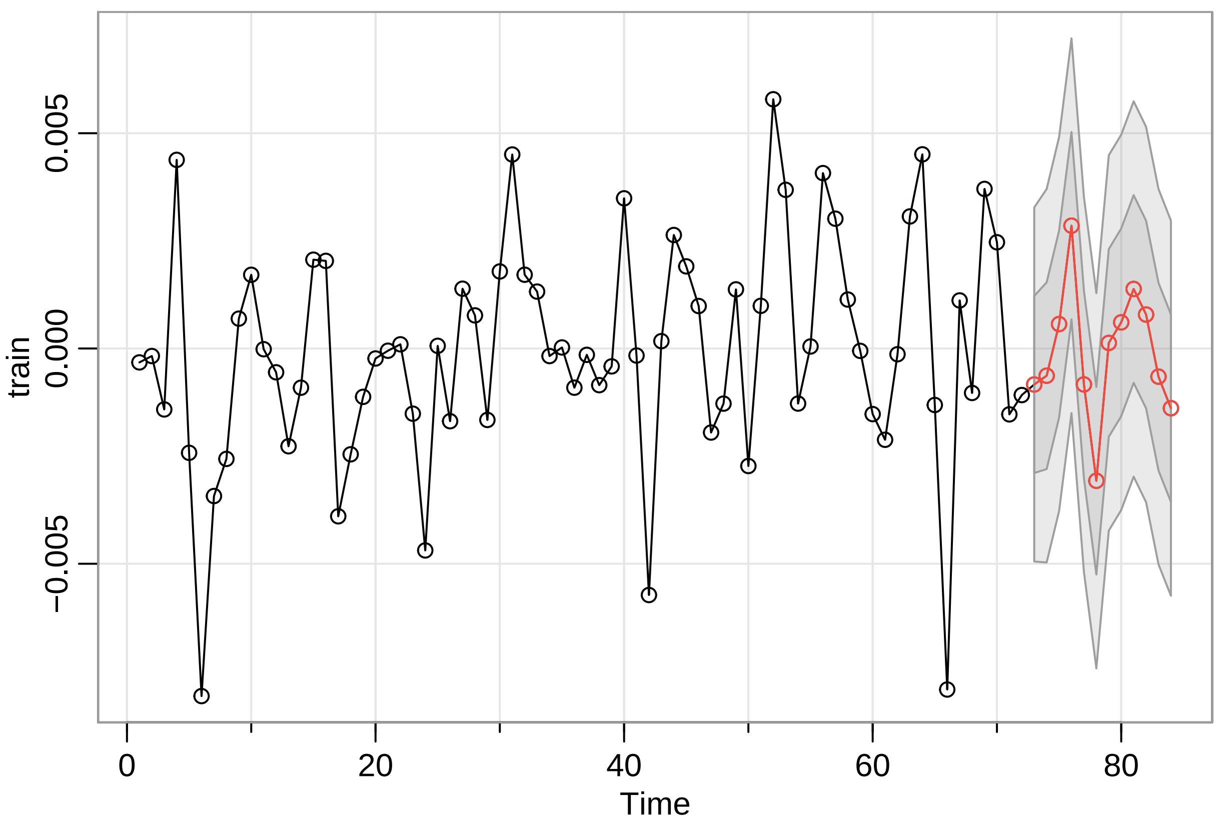

0.005
0.000
−0.005
train
0
20
40
60
80
Time

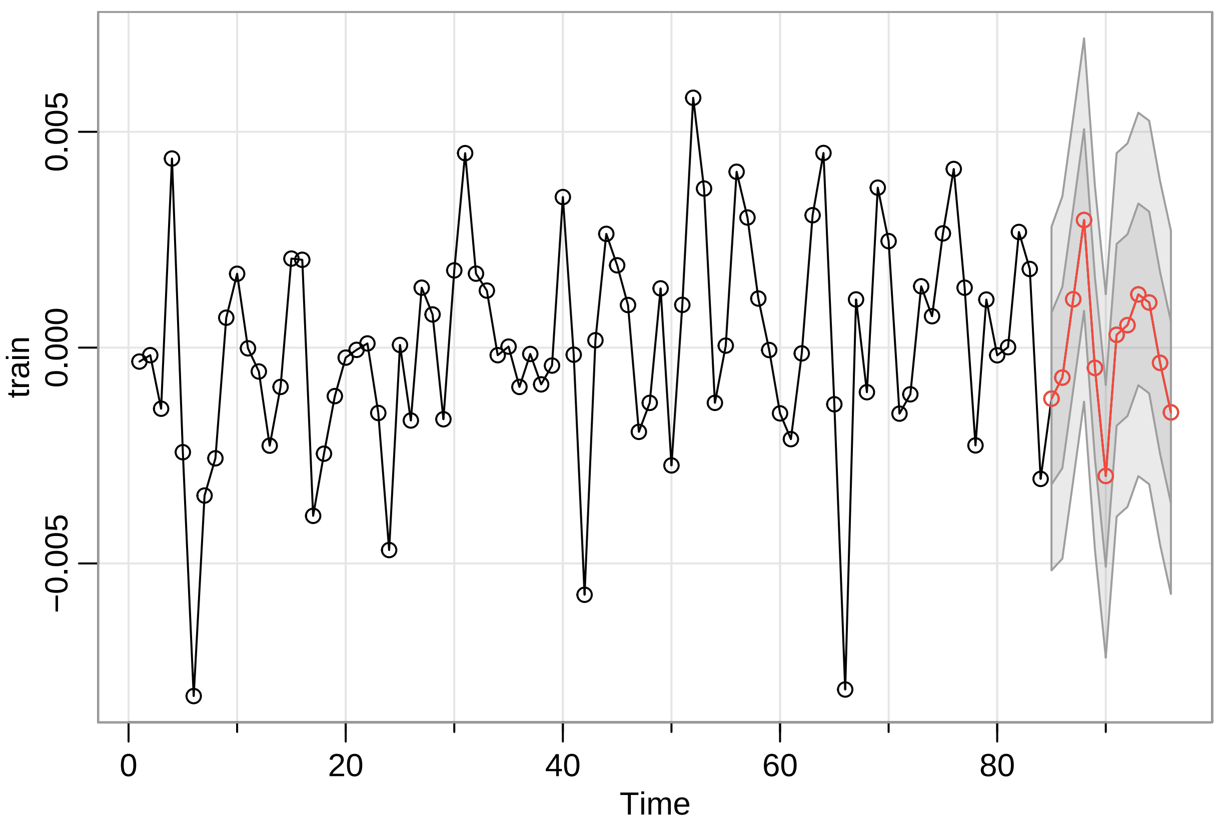


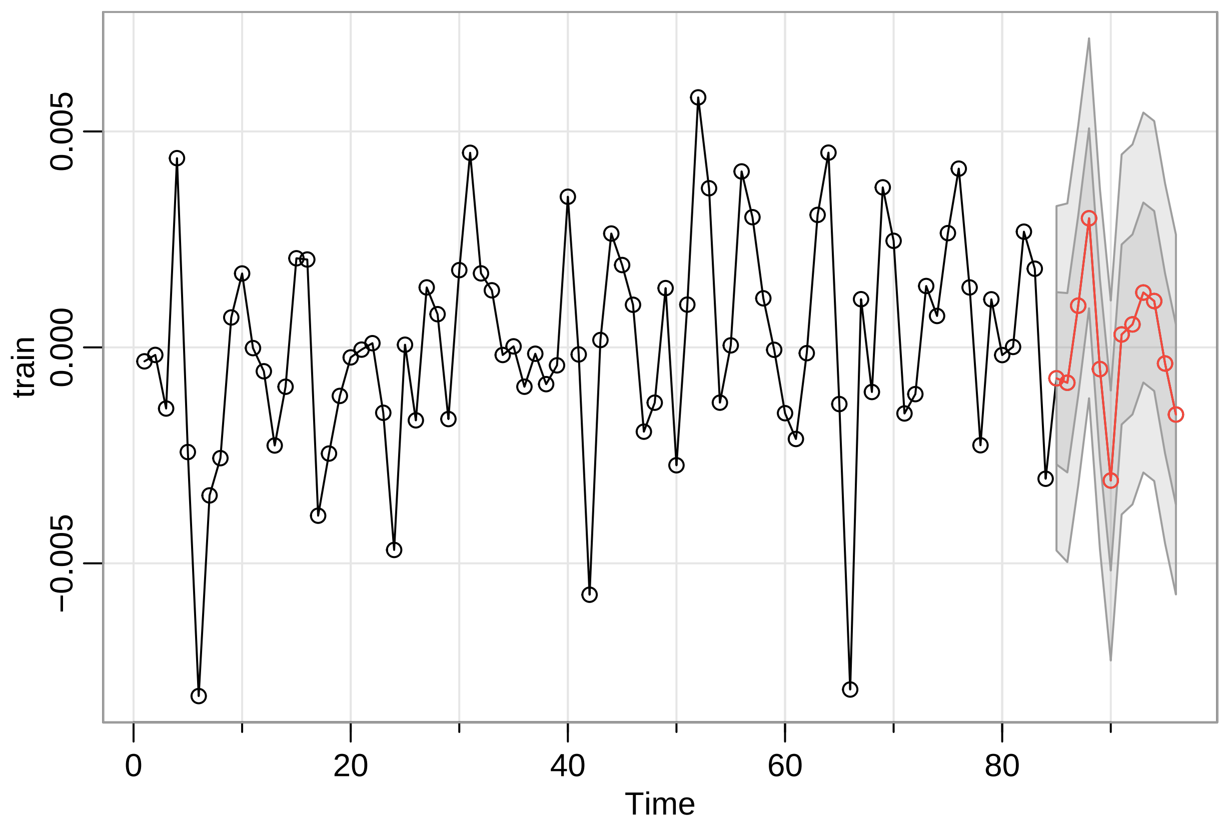
train
0.005
0.000
−0.005
0
20
40
60
80
Time

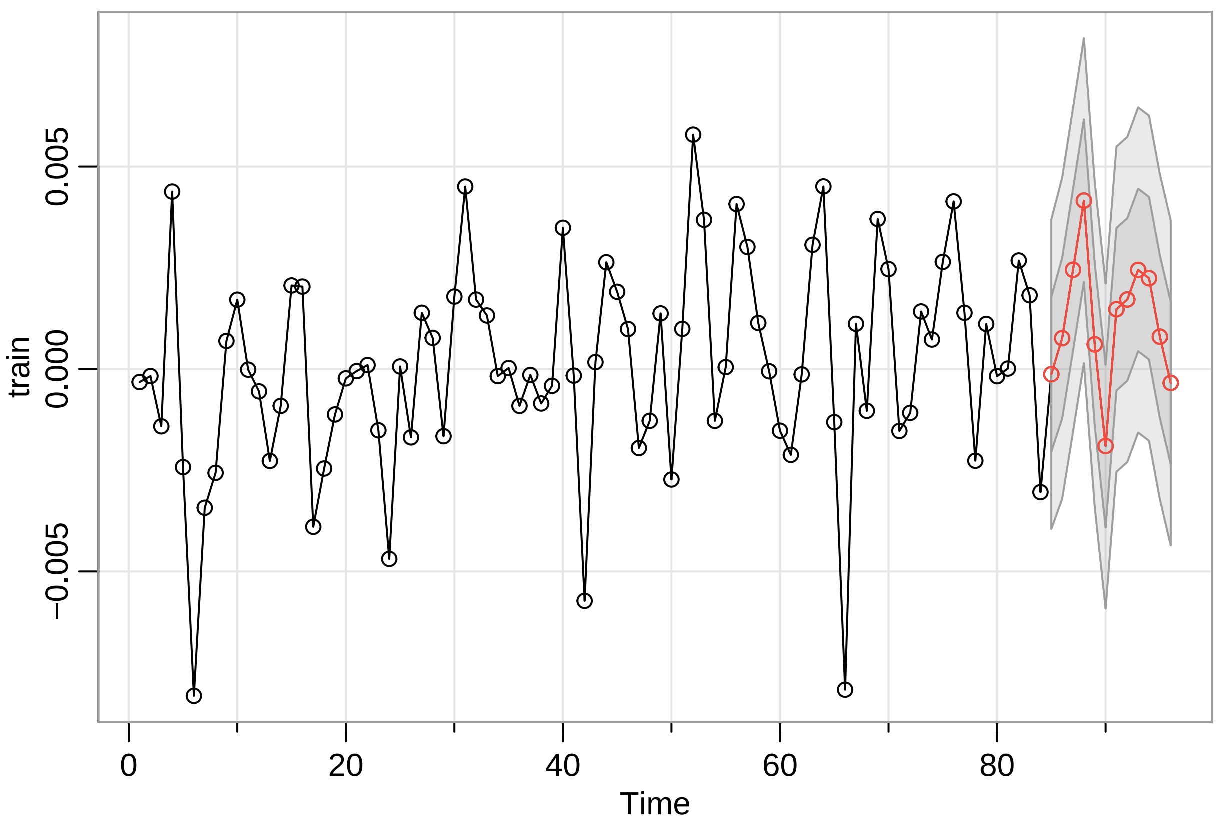


0.005
0.000
−0.005
train
0
20
40
60
80
Time

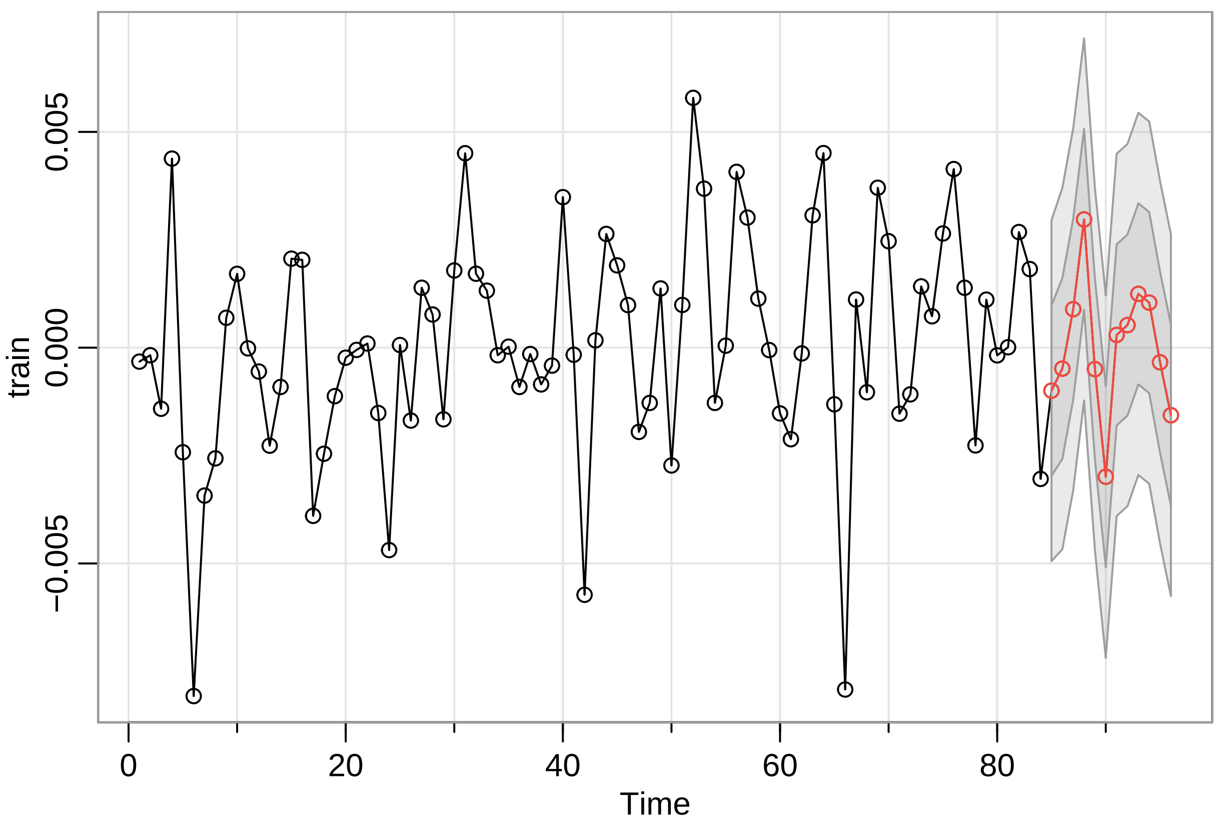

train
0.005
0.000
−0.005
0
20
40
60
80
Time


train
0.005
0.000
−0.005
0
20
40
60
80
Time

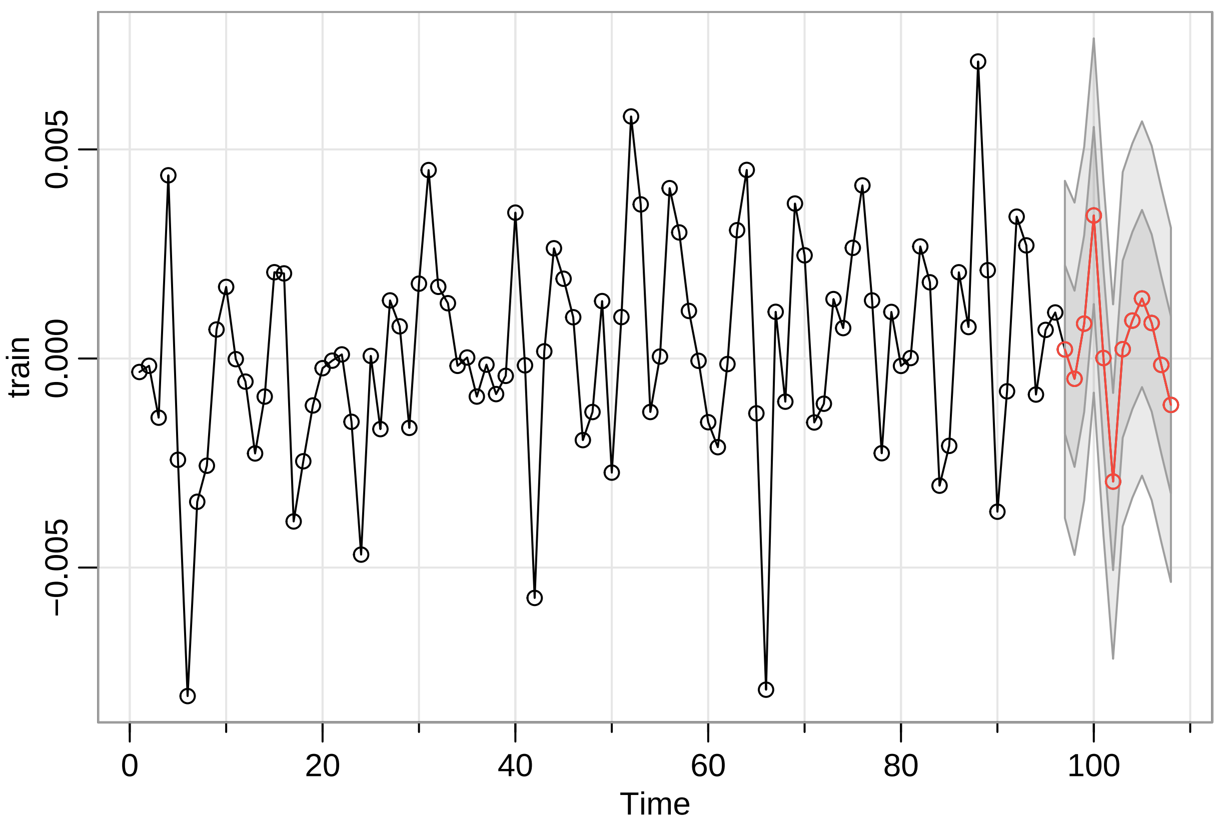

0.005
0.000
−0.005
train
0
20
40
60
80
Time

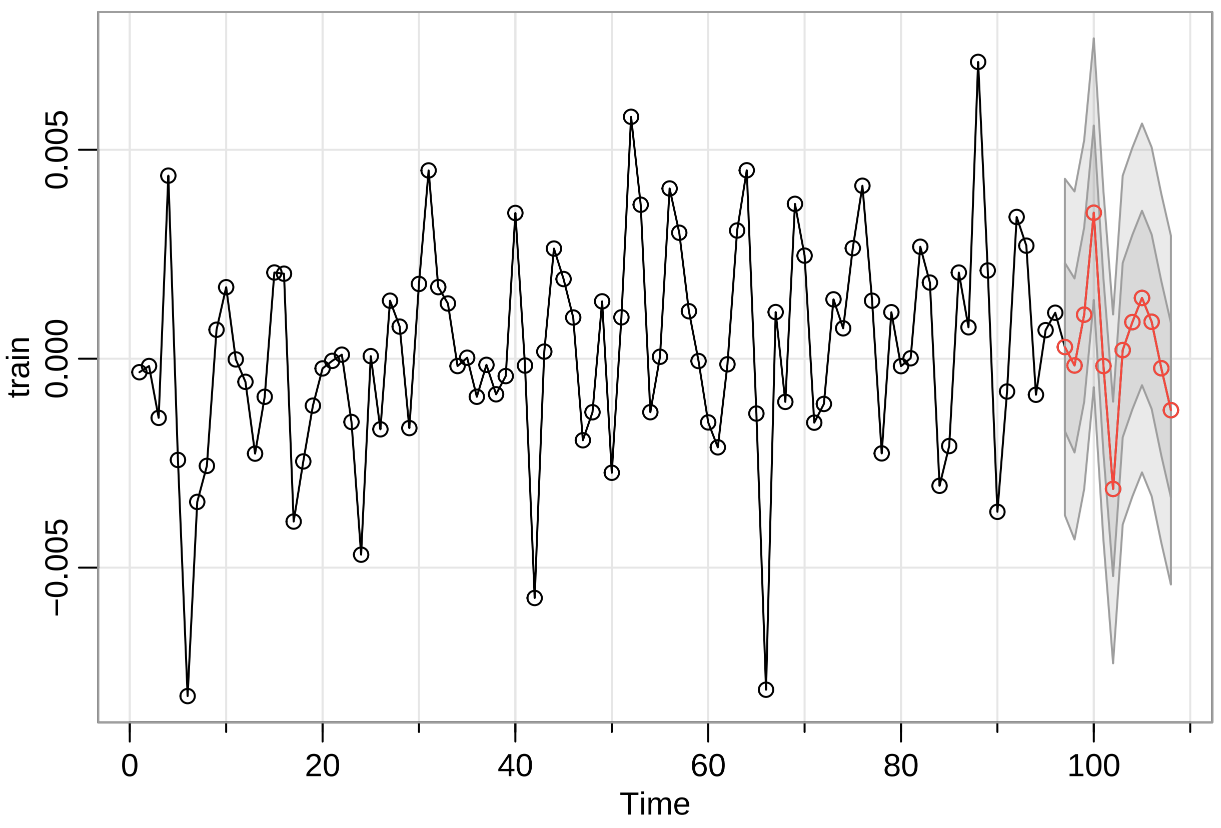


train
0.005
0.000
−0.005
0
20
40
60
80
Time

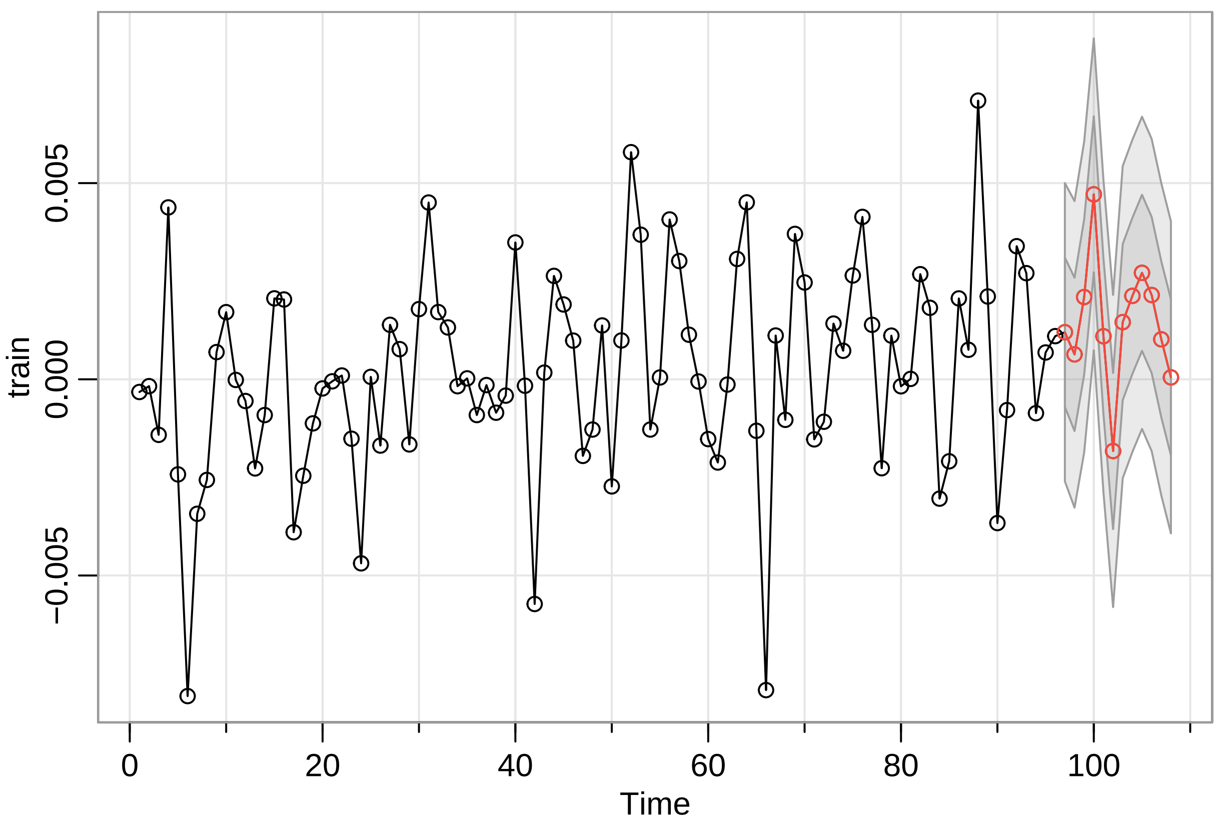


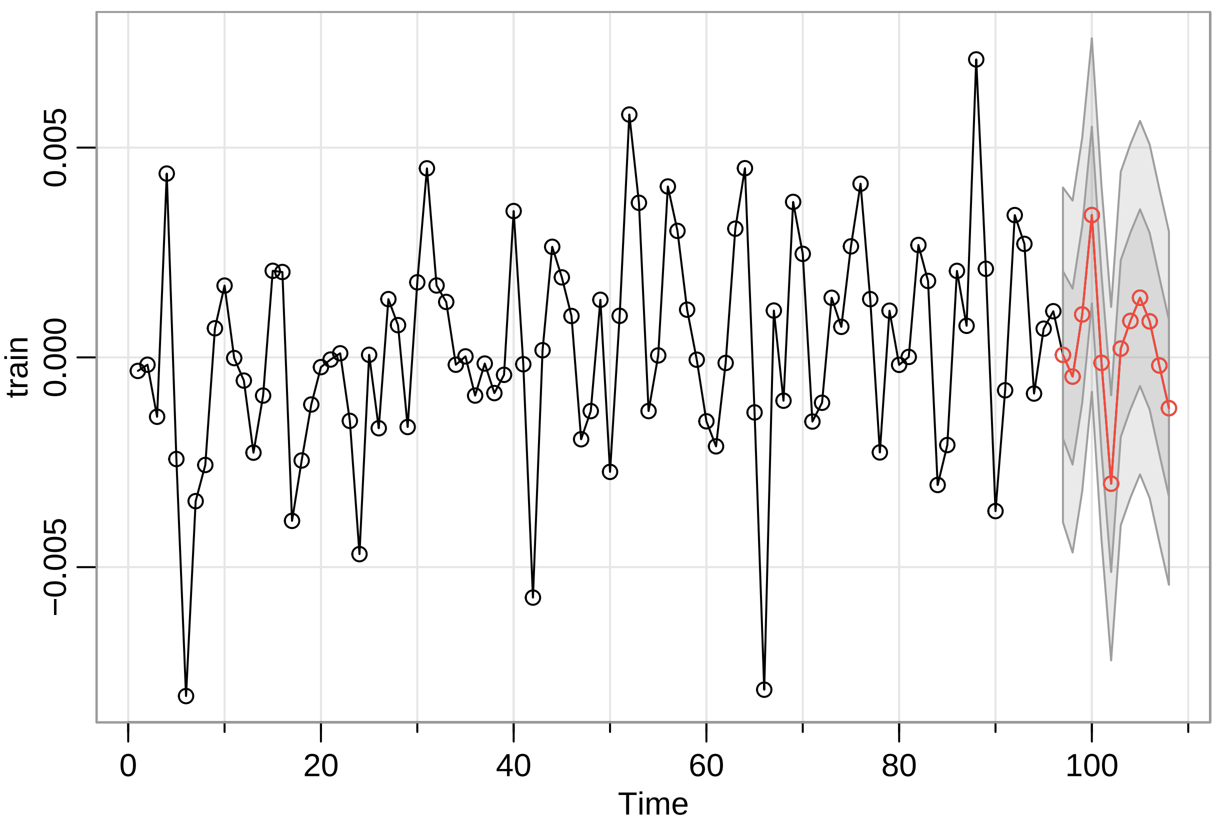
0.005
0.000
−0.005
train
0
20
40
60
80
Time


train
0.005
0.000
−0.005
Time
0
20
40
60
80
100

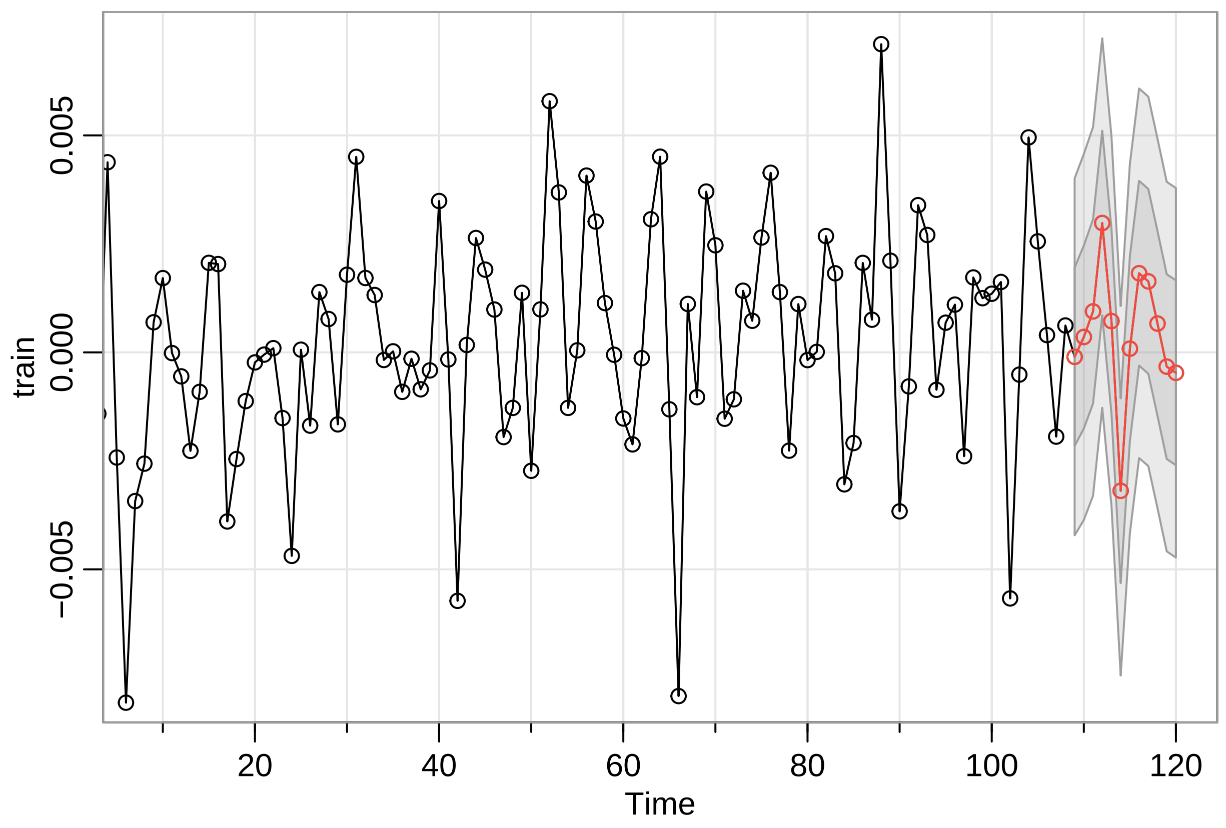


train
0.005
0.000
−0.005
0
20
40
60
80
100
Time

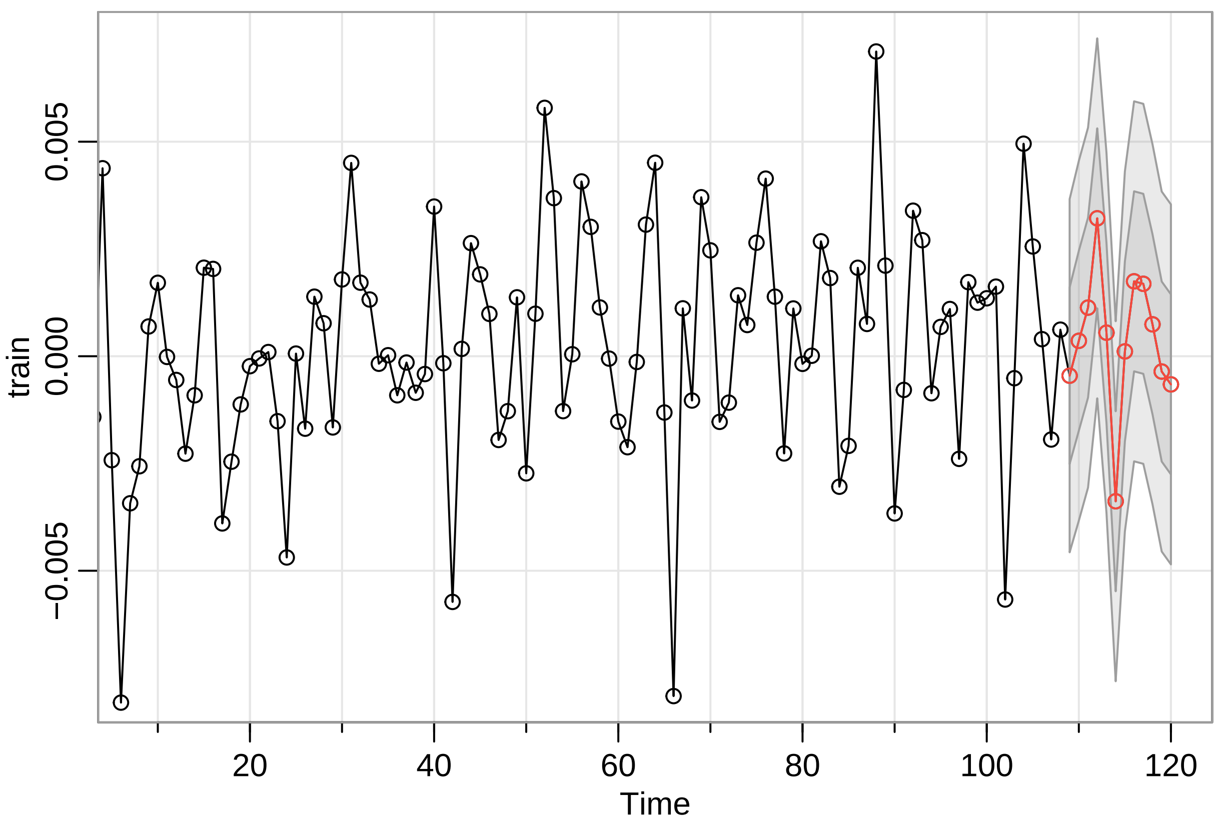

train
0.005
0.000
−0.005
0
20
40
60
80
100
Time



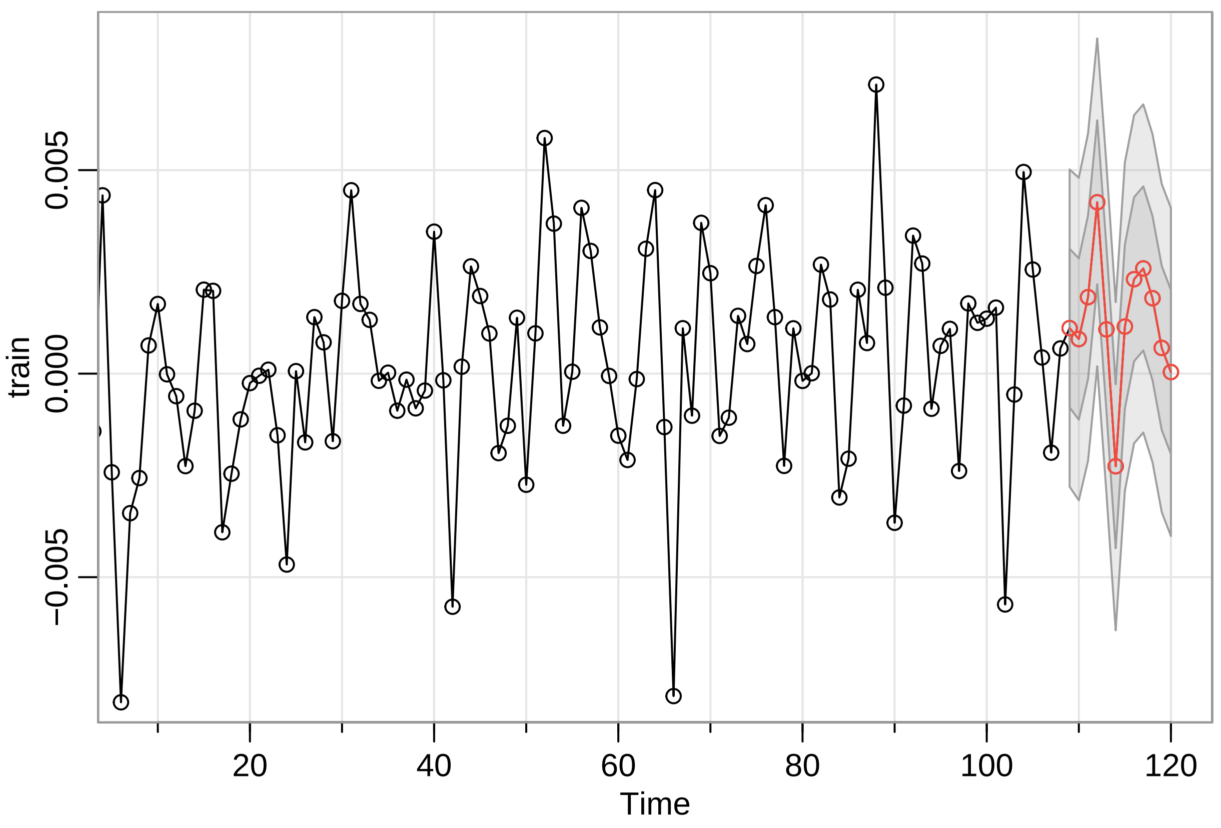
train
0.005
0.000
−0.005
0
20
40
60
80
100
Time



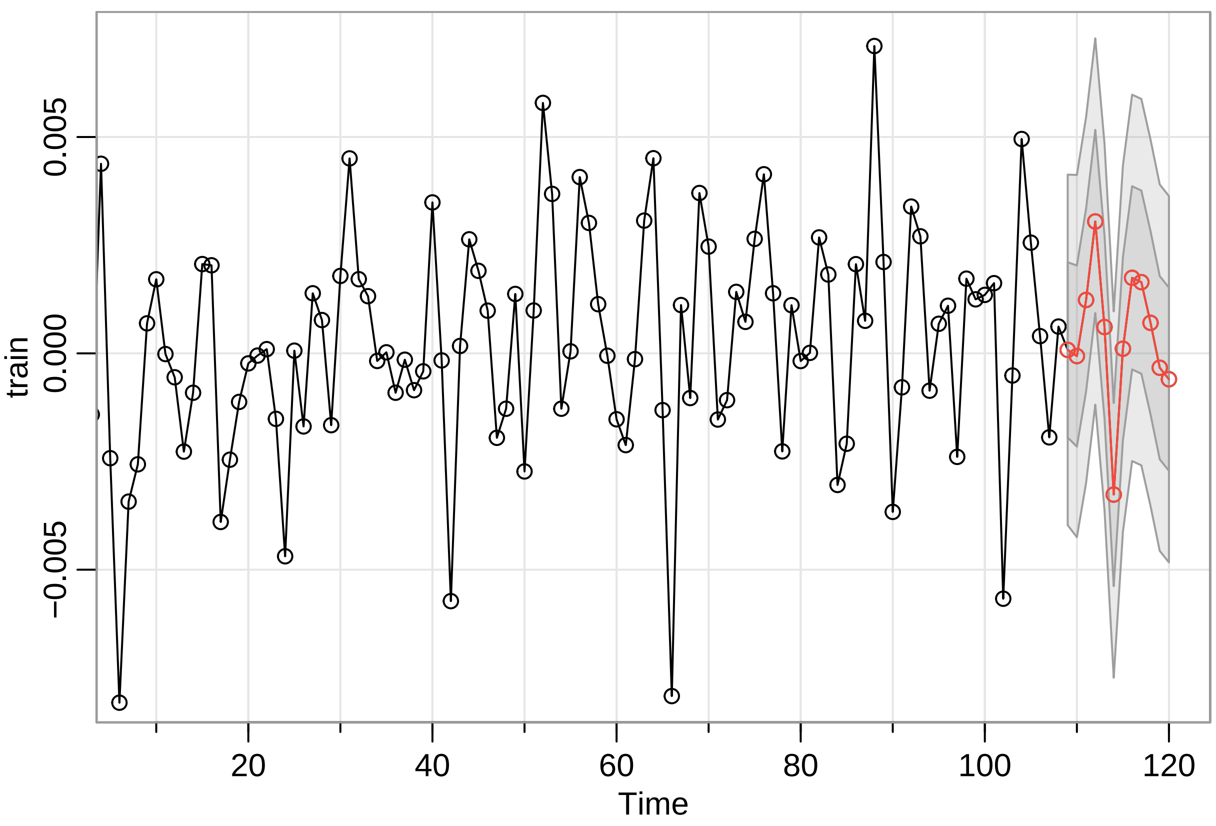
0.005
0.000
−0.005
train
20
40
60
80
100
120
Time

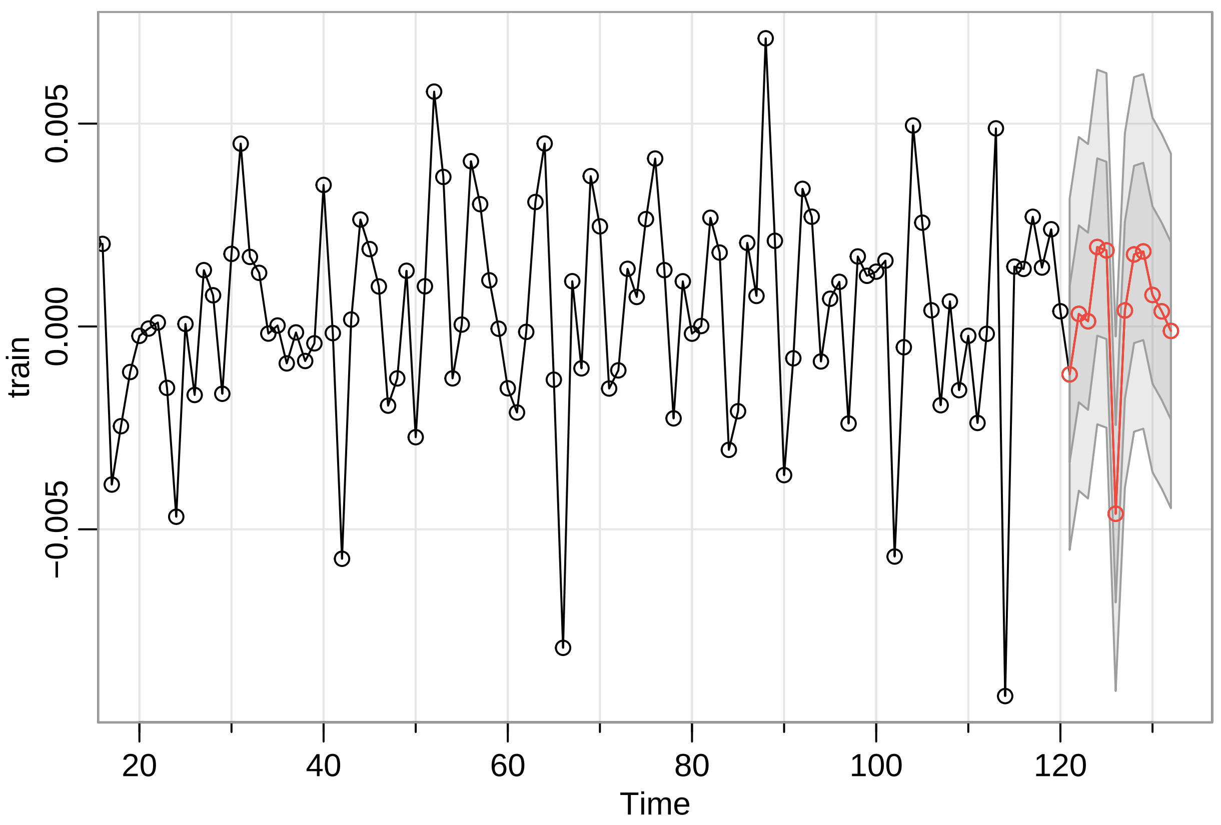


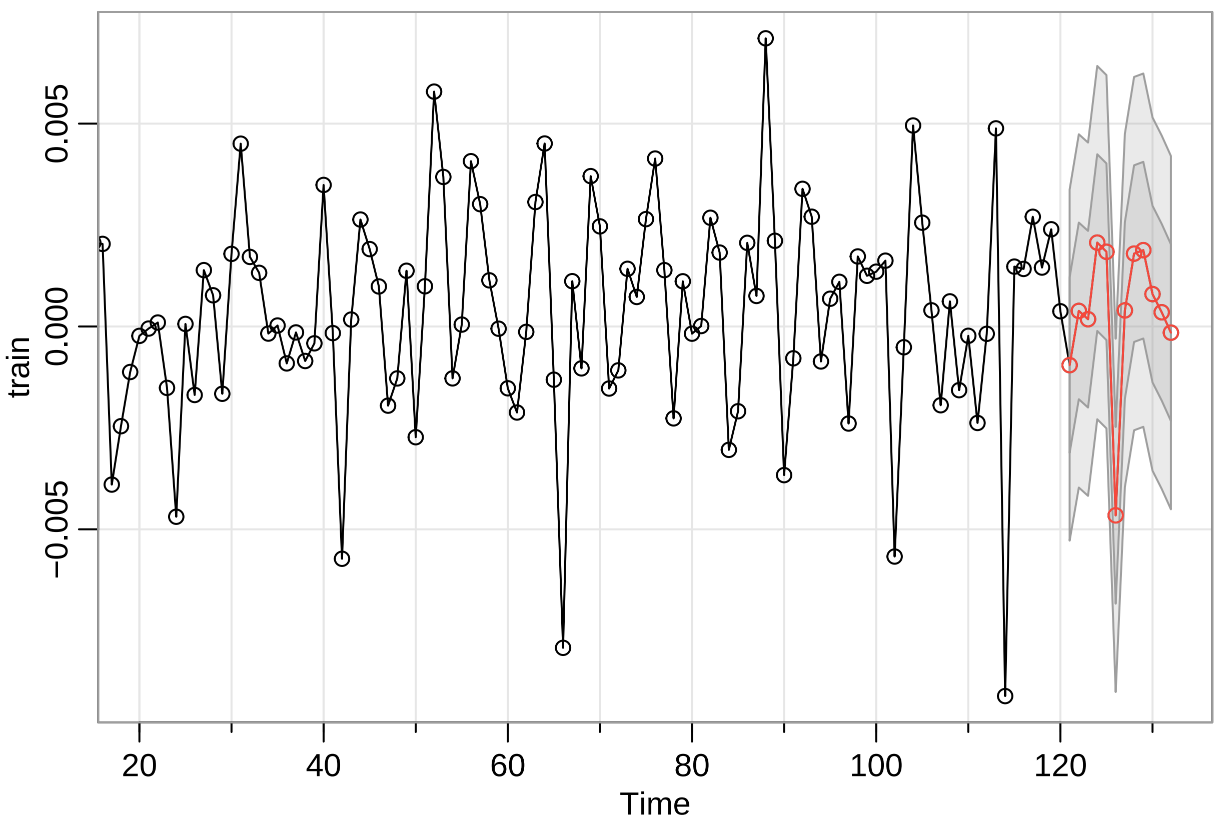
train
0.005
0.000
−0.005
20
40
60
80
100
120
Time



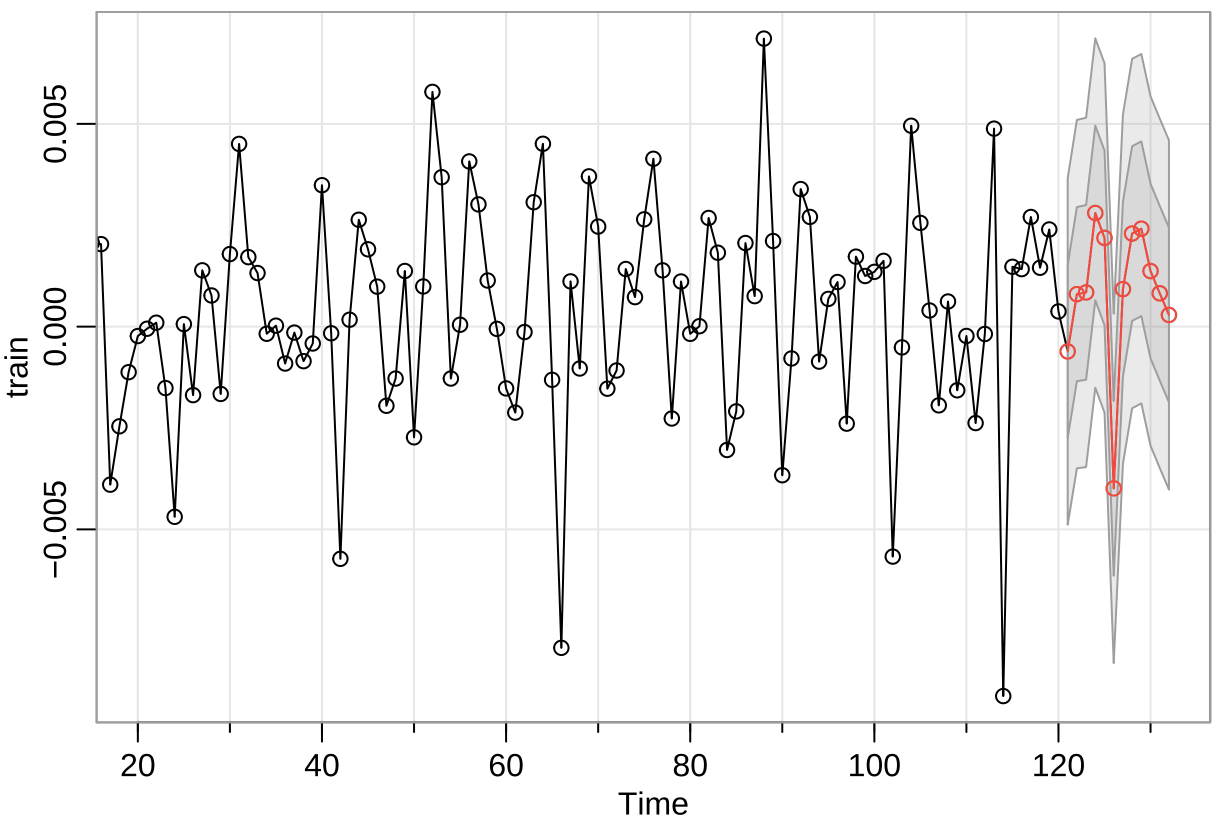
train
0.005
0.000
−0.005
20
40
60
80
100
120
Time

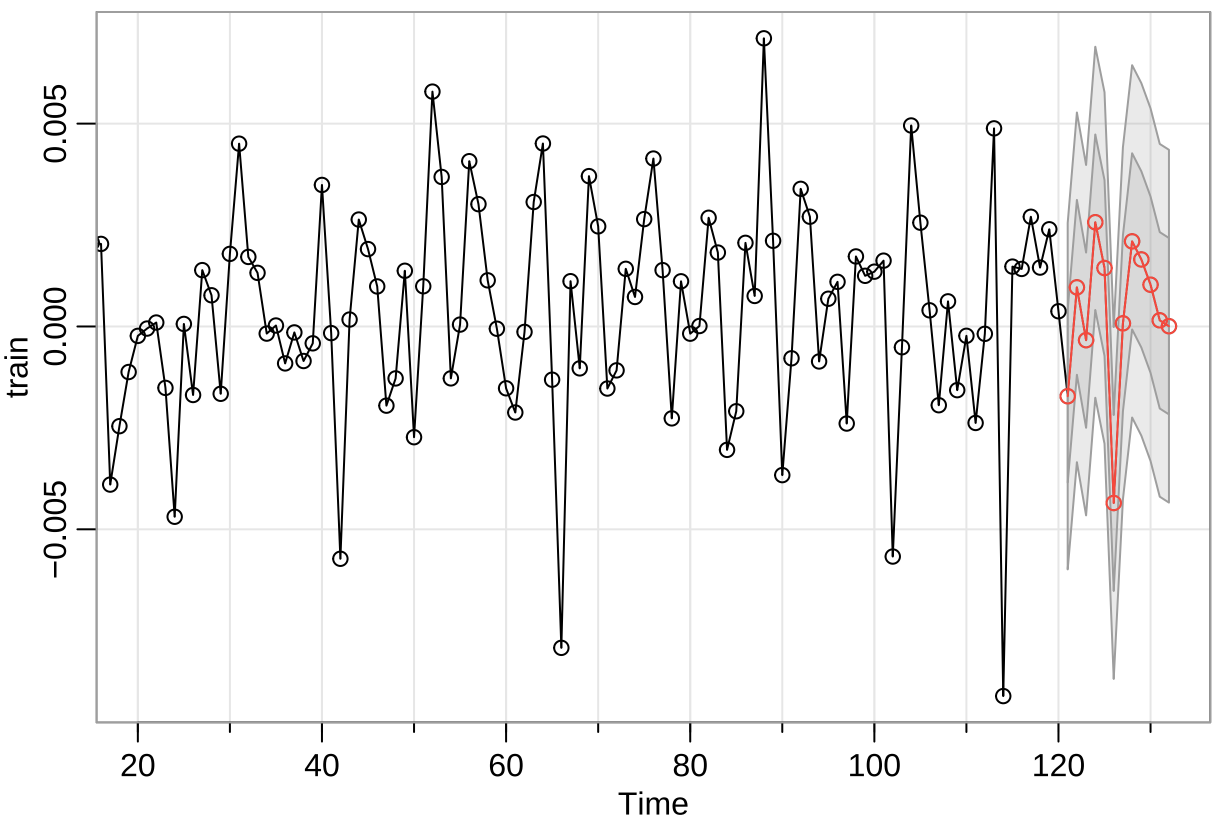

train
0.005
0.000
−0.005
20
40
60
80
100
120
Time


train
0.005
0.000
−0.005
20
40
60
80
100
120
Time

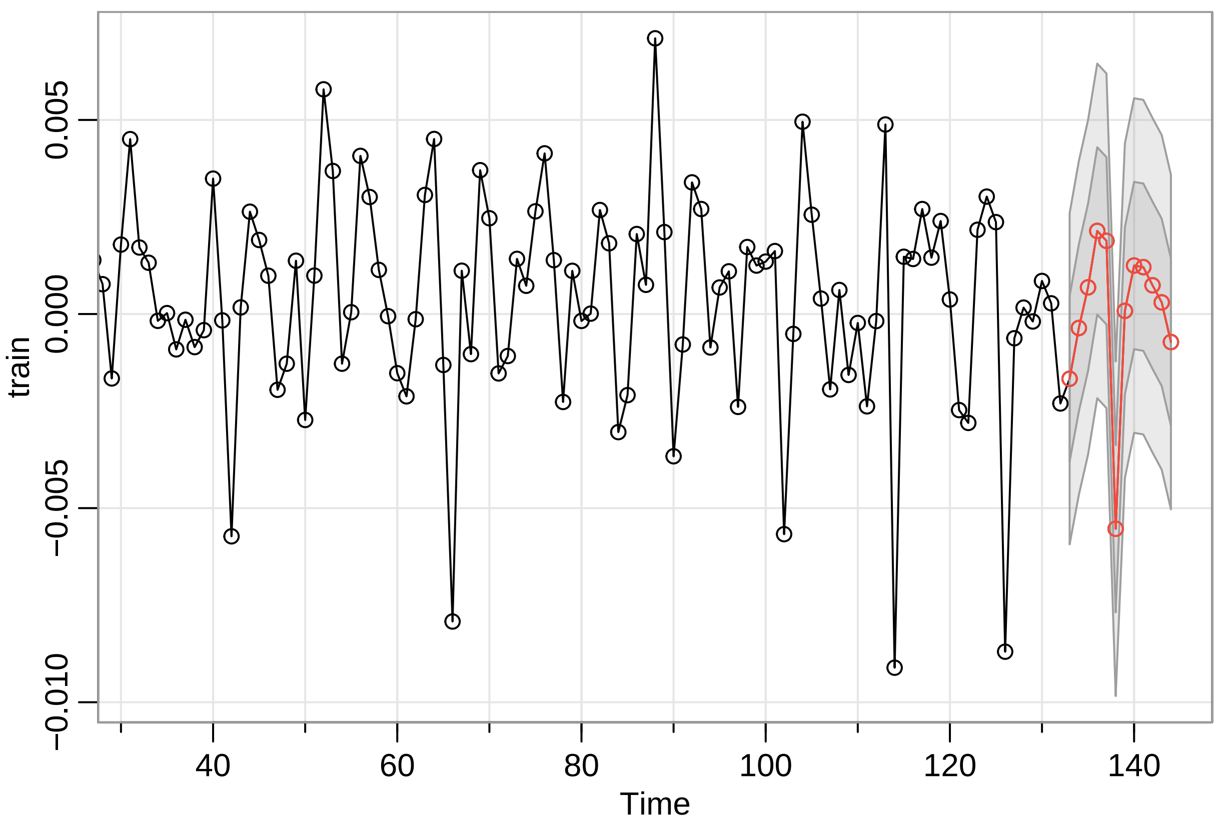


train
0.005
0.000
−0.005
20
40
60
80
100
120
Time

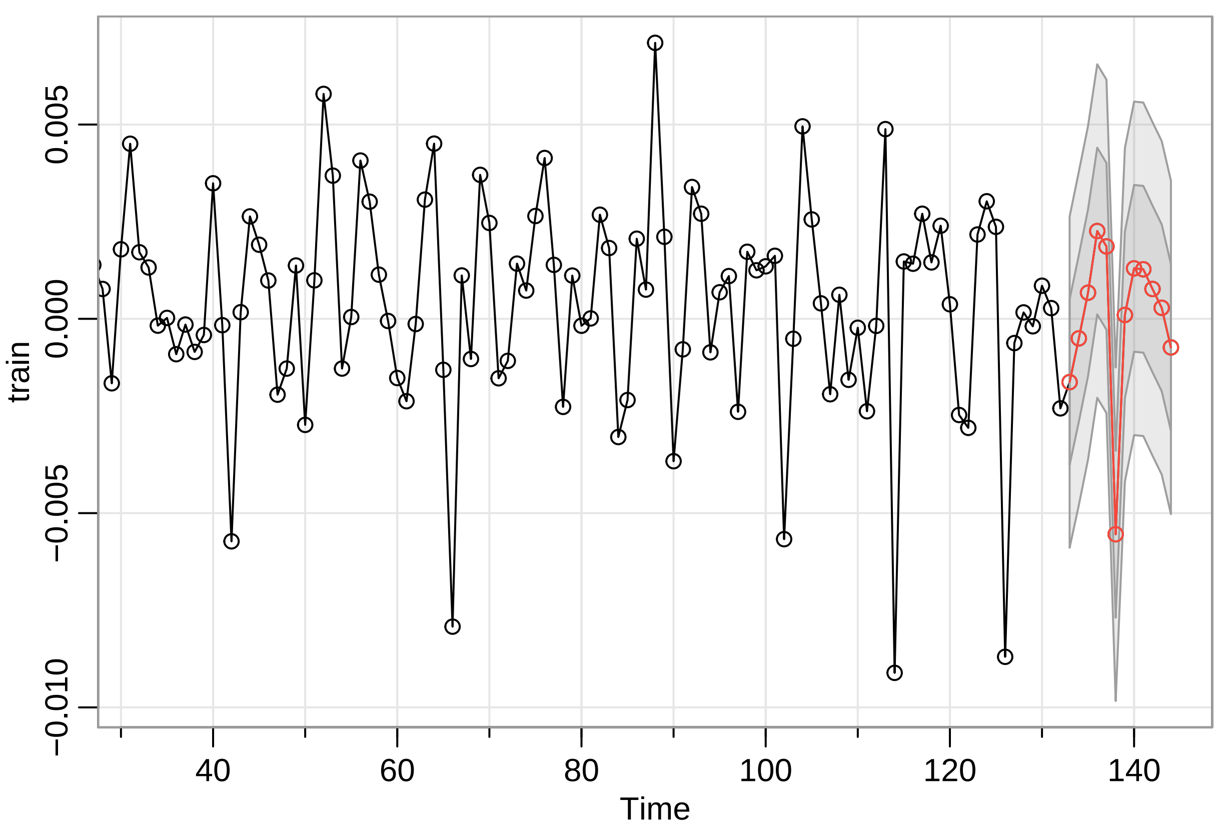

train
0.005
0.000
−0.005
20
40
60
80
100
120
Time



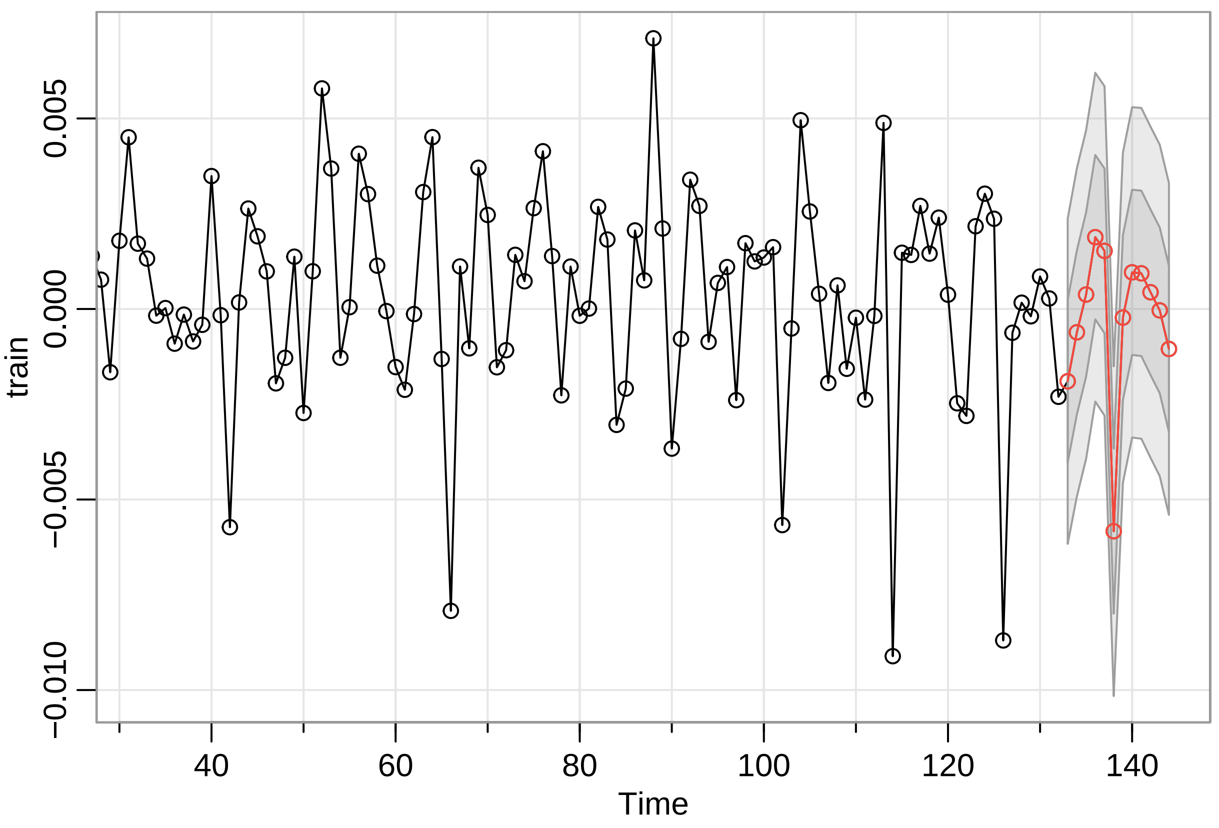
0.005
0.000
−0.005
train
20
40
60
80
100
120
Time

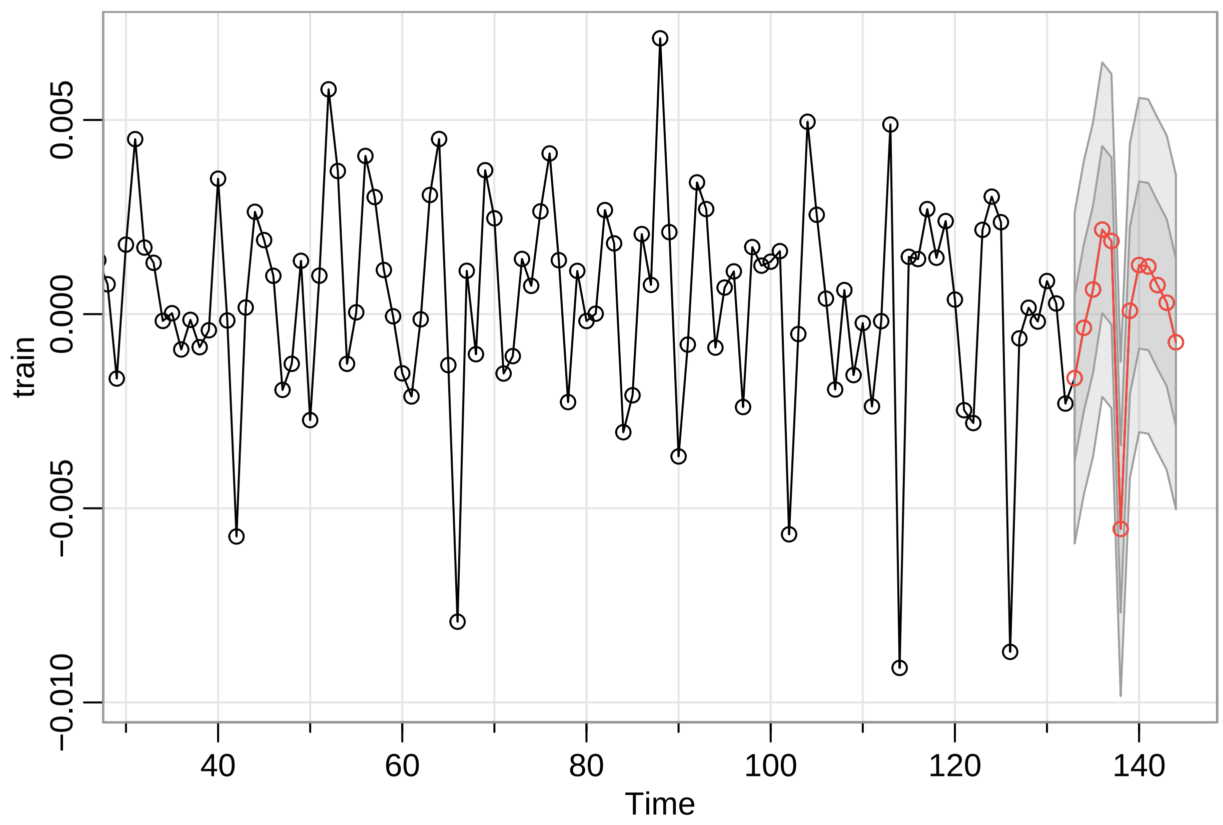


train
0.005
0.000
−0.005
−0.010
40
60
80
100
120
140
Time



train
0.005
0.000
−0.005
−0.010
40
60
80
100
120
140
Time


0.005
0.000
−0.005
−0.010
train
40
60
80
100
120
140
Time

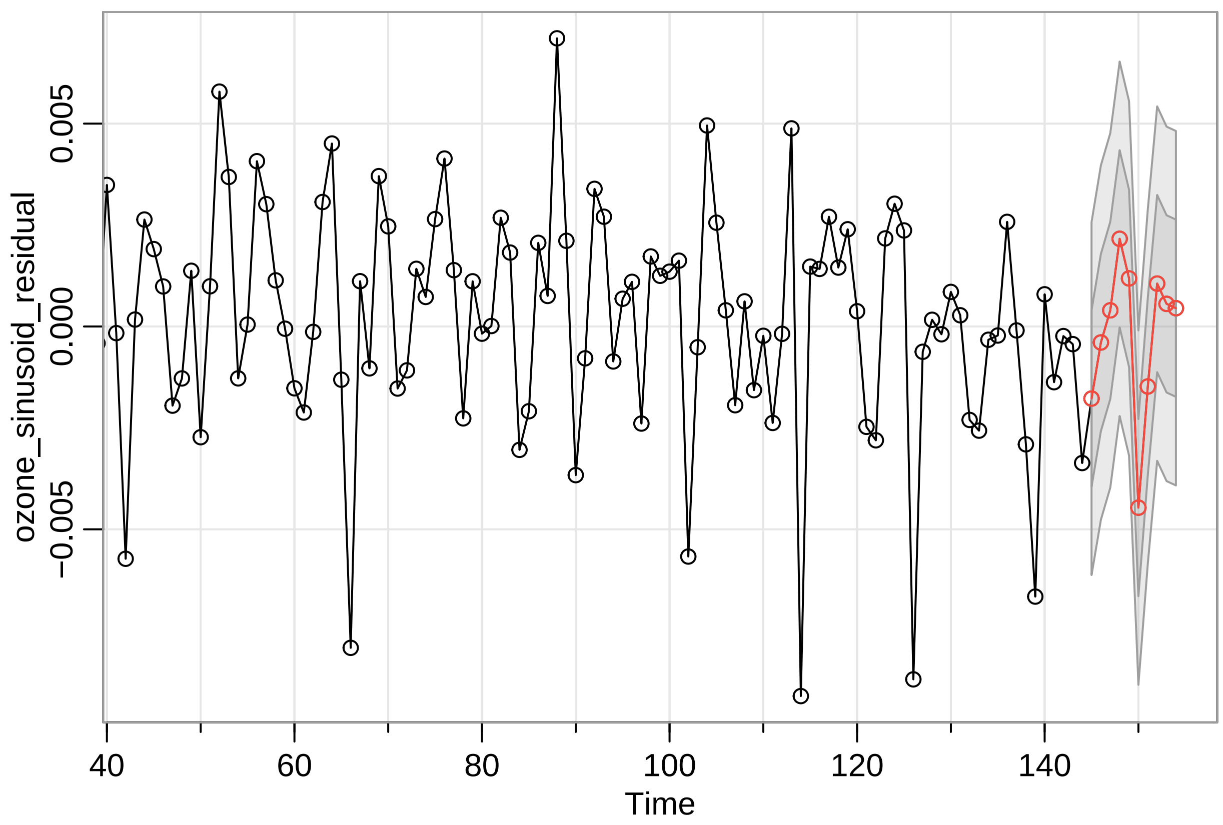


```
sse = rbind(sum(sse1),sum(sse2),sum(sse3),sum(sse4))
rownames(sse) = c("SARIMA(2,0,0)(1,0,1)12","SARIMA(1,0,0)(1,0,1)12","SARIMA(2,1,1)(1,0,1)12","SARIMA(2,(
colnames(sse) = c("SSE")
print(sse)
```

```
##                                  SSE
## SARIMA(2,0,0)(1,0,1)12 0.0003465800
## SARIMA(1,0,0)(1,0,1)12 0.0003502556
## SARIMA(2,1,1)(1,0,1)12 0.0003684125
## SARIMA(2,0,1)(1,0,1)12 0.0003583199
```

Future 10 values prediction

```
m1_forecast <- sarima.for(ozone_sinusoid_residual, p=2, d=0, q=0, P=1, D=0, Q=1, S=12, n.ahead=10)$pred
```

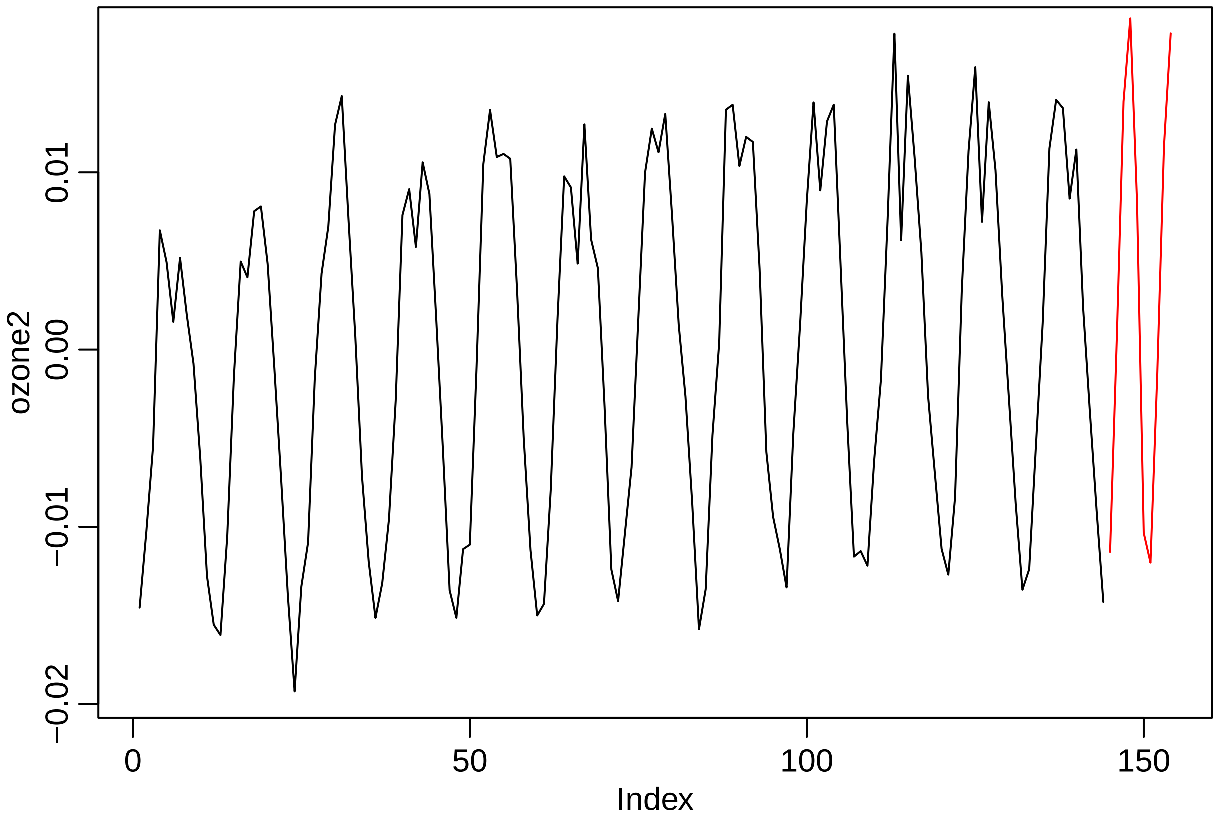


```
ts = as.data.frame(145:154)
predict_t = 145:154
prediction = c()
coef = ozone_sinusoid_model$coefficients

for (t in predict_t){
  pred = coef[1]+coef[2]*sin(t) + coef[3]*t+ coef[4]*cos(t)+ coef[5]*sin(t)*t + coef[6]*cos(t)*t
  prediction = c(prediction,pred)
}
prediction
```

```
##  (Intercept)  (Intercept)  (Intercept)  (Intercept)  (Intercept)  (Intercept)
## -0.009641503  0.001033070  0.013597515  0.016528621  0.007152463 -0.005881246
##  (Intercept)  (Intercept)  (Intercept)  (Intercept)
## -0.010543420 -0.002492147  0.010919518  0.017393830
```

```
m1_forecast = as.vector(m1_forecast)
prediction = as.vector(prediction)
predictions = m1_forecast+prediction

ozone2 = c(ozone,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA)

ts2 = 1:154
preds = rep(NA, 144)
preds = c(preds,predictions)
```

```r
plot(ozone2,type ="l")
lines(ts2,preds,col = "red")
```


ozone2
Index