case study 4 v3

Lance Dacy, Reanan McDaniel, Shawn Jung, Jonathan Tan

```r
#analysis packages
library(tswge)
library(changepoint)
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Successfully loaded changepoint package version 2.2.2
##  NOTE: Predefined penalty values changed in version 2.2.  Previous penalty values with a postfix 1 i
```

```r
library(tseries)
```

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

```r
library(DataExplorer)
library(nnfor)
```

```
## Loading required package: forecast
```

```r
#data manipulation
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

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

```r
library(lubridate)
```

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

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

```r
library(tibbletime)
```

```
## Warning: package 'tibbletime' was built under R version 4.0.2

##
## Attaching package: 'tibbletime'

## The following object is masked from 'package:stats':
##
##     filter
```

```r
#visualization
library(ggplot2)
library(cowplot)
```

```
##
## ********************************************************

## Note: As of version 1.0.0, cowplot does not change the

##   default ggplot2 theme anymore. To recover the previous

##   behavior, execute:
##   theme_set(theme_cowplot())

## ********************************************************

##
## Attaching package: 'cowplot'

## The following object is masked from 'package:lubridate':
##
##     stamp
```

```r
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```r
#data source
library(cdcfluview)
```

import data from… ??? https://apps.who.int/flumart/Default?ReportNo=12 query below is SE Asia from 1995 to 2019

```r
raw_data  <- read.csv('D:/SMU/DS 7333 Quantify The World/FluNetInteractiveReport_SE_Asia_1995-2019.csv')
```

Column guide for WHO data csv columns

SPEC_RECIEVED_NB - Specimens recieved by the WHO

SPEC_PROCESSED_NB - specimens processed by the WHO

AH1 - all A type influenza viruses categorized by subtype

A(H1N1pdm09

AH3

AH5

ANOTSUBTYPED

INF(A) - total of all influenza A viruses detected

BYAMAGATA - B(Yamagata lineage) - various types of influenze B type viruses

BVICTORIA - B(Victoria Lineage)

BNOTDETERMINED

INF_B - total influenze B viruses

ALL_INF - number of influenze positive viruses

ALL_INF2 - number of influenze negative viruses

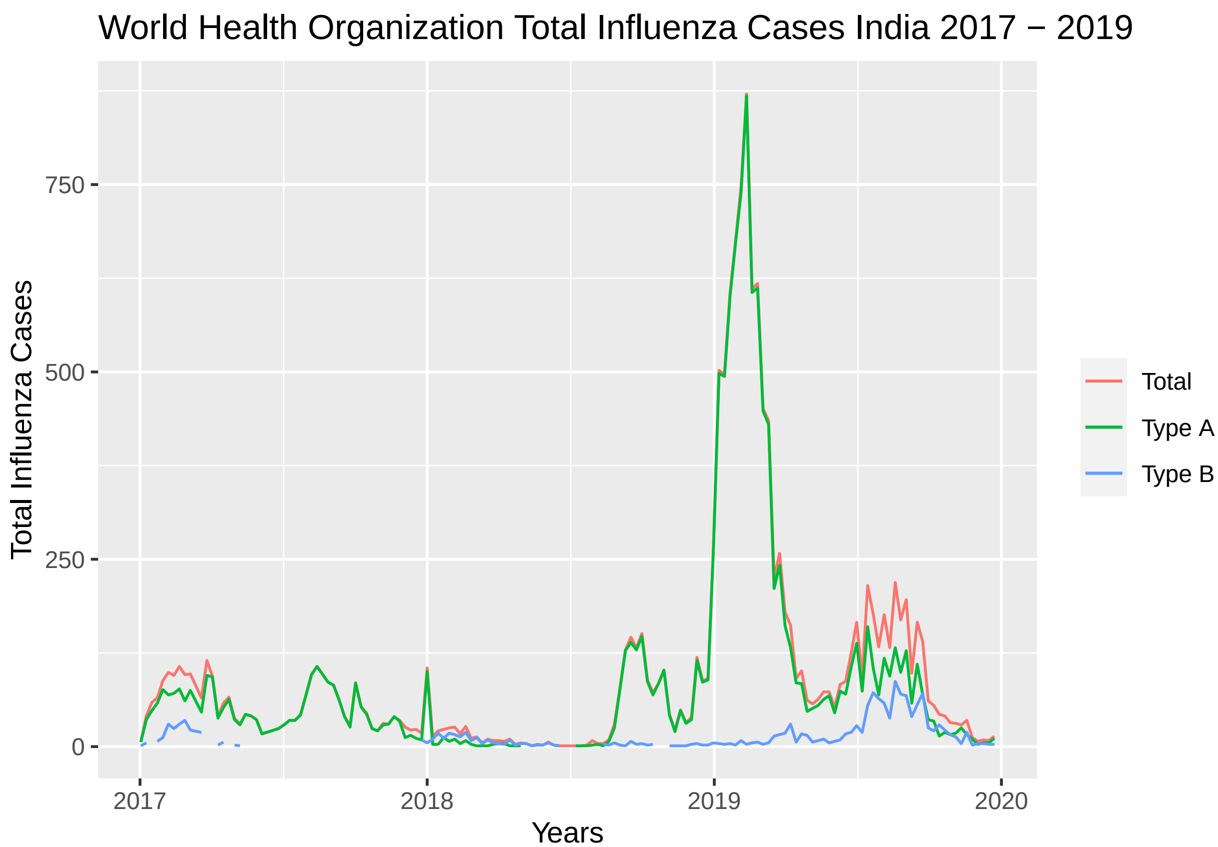
TITLE - ili activity? most have no report as the column

will have to look more into the details, but it looks like this dataset is recording the number of virus strains, not individual cases of that virus. specimens recieved/processed usually matches up with INF + or -, indicating that those specimens were all tested and sorted into either yes, it was an influenza virus, or no, it was not.

We'll pick S asia, India for total influenza cases

```r
S_Asia <- raw_data[which(raw_data$FLUREGION == 'Southern Asia'), ] #select rows from SE asia
tlist <- S_Asia[complete.cases(S_Asia$ALL_INF), ] #select only rows that aren't missing data in the all
t_ind <- tlist[which(tlist['ï..Country'] == 'India'), ]
t_ind <- t_ind[which(t_ind$Year == 2017| t_ind$Year == 2018 | t_ind$Year == 2019), ] #select 3 years
ttime = t_ind$SDATE %>% as.Date("%m/%d/%Y") #store the timeline as date object vector
#it looks like all the weeks fall on the same SDATE's, so it should be safe to use the same date vector

ggplot(t_ind) +
  geom_line(aes(x = ttime, y = ALL_INF, col = 'Total'))+
  #geom_line(aes(x = ttime, y = AH1, col = 'AH1'))+
  #geom_line(aes(x = ttime, y = AH1N12009, col = '2009 Swine Flu'))+
  #geom_line(aes(x = ttime, y = AH3, col = 'AH3'))+
  #geom_line(aes(x = ttime, y = AH5, col = 'AH5'))+
  #geom_line(aes(x = ttime, y = ANOTSUBTYPED, col = 'Untyped Inf A'))+
```

```r
  geom_line(aes(x = ttime, y = INF_A, col = 'Type A'))+
  #geom_line(aes(x = ttime, y = BYAMAGATA, col = 'Yamagata STRAIN B'))+
  #geom_line(aes(x = ttime, y = BVICTORIA, col = 'Victoria Strain B'))+
  #geom_line(aes(x = ttime, y = BNOTDETERMINED, col = 'Untyped Inf B'))+
  geom_line(aes(x = ttime, y = INF_B, col = 'Type B')) +
  ggtitle("World Health Organization Total Influenza Cases India 2017 - 2019") +
  xlab('Years') + ylab('Total Influenza Cases') + theme(legend.title = element_blank())
```

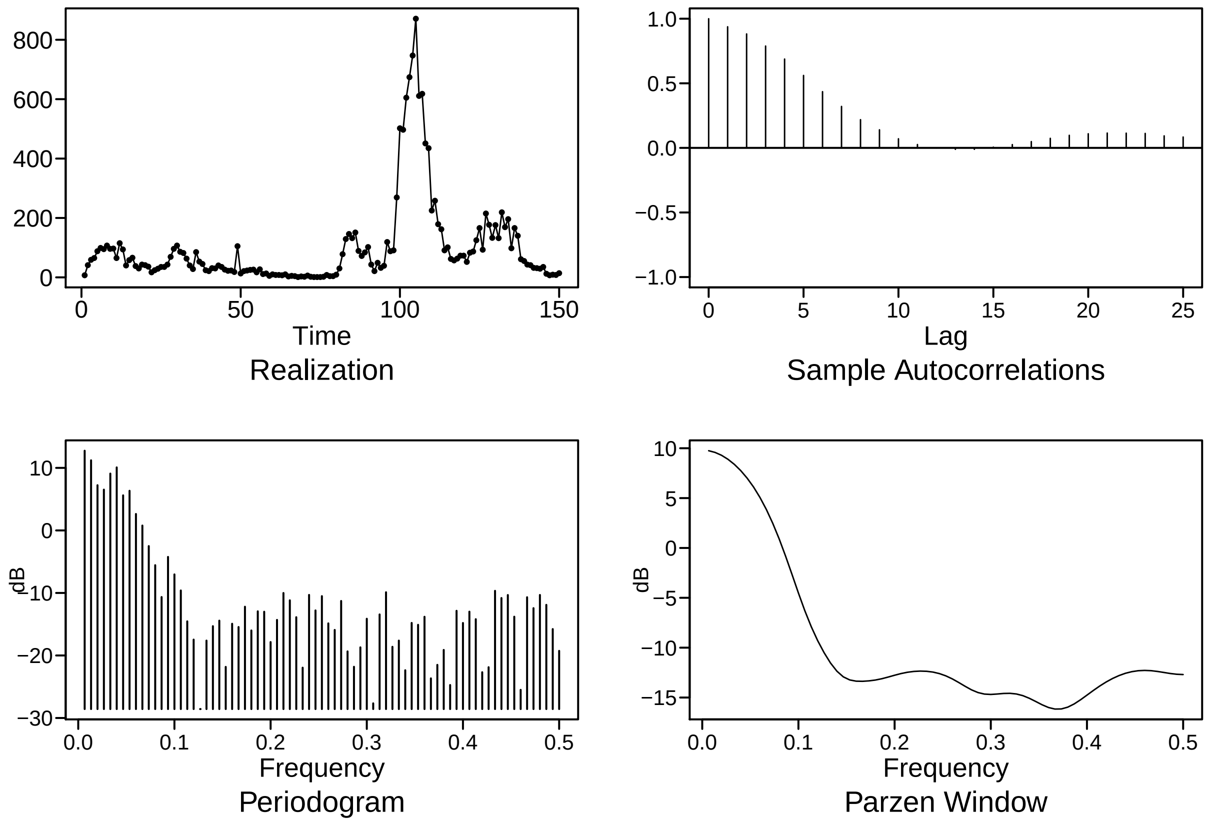



```r
  scale_y_continuous()
```

```
## <ScaleContinuousPosition>
##  Range:
##  Limits:    0 --    1
```

diagnostics, stationarity

```r
tdata = t_ind$ALL_INF
#wrap objects in invisible() to hide the long console output and leave only the graph
invisible(plotts.sample.wge(tdata))
```

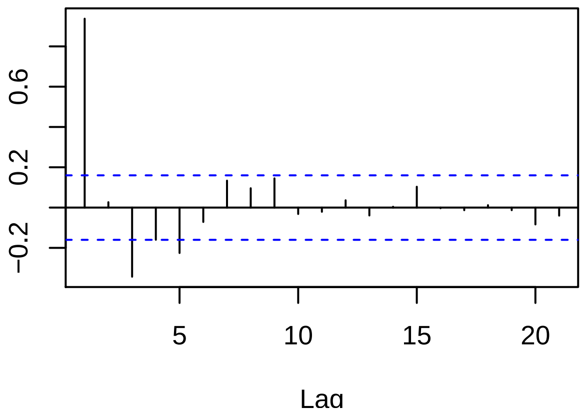



```r
#stationairity
adf.test(tdata)
```

```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  tdata
## Dickey-Fuller = -4.3055, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```r
pacf(tdata)
#changepoint
par(mfrow = c(2,1))
```



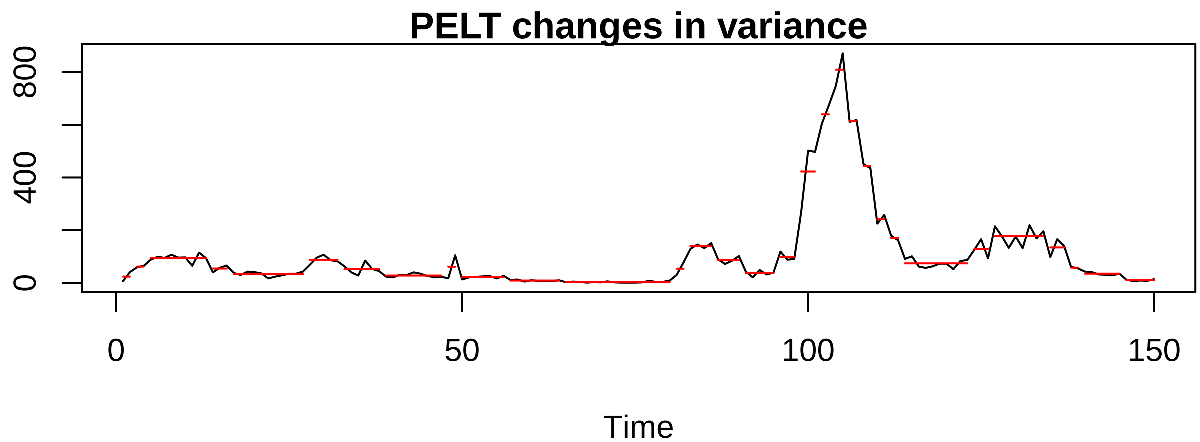

```
v2 <- tdata
v2.pelt <- cpt.meanvar(v2, test.stat = 'Poisson', method = "PELT")
plot(v2.pelt, main = 'PELT changes in variance')
cpts.ts(v2.pelt)
```

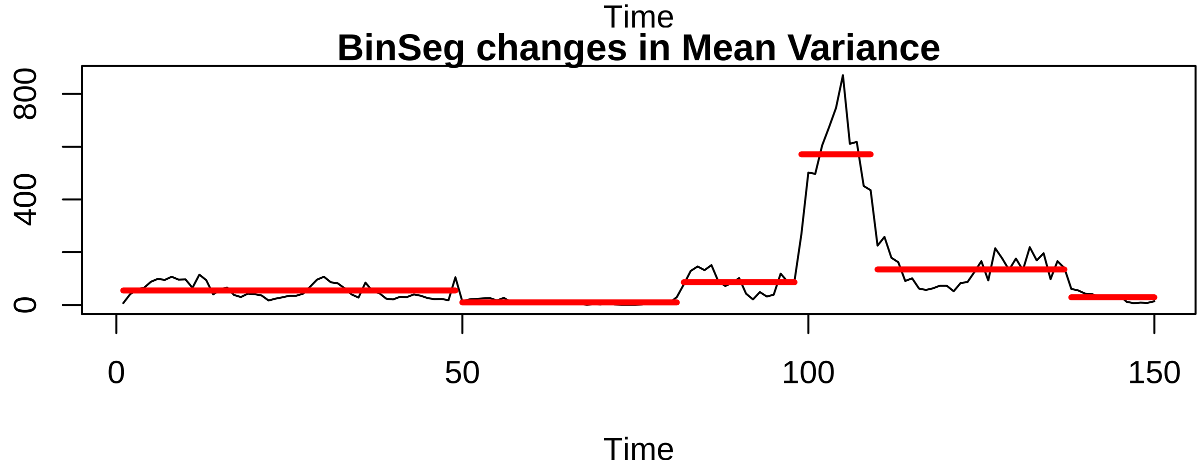
```
##  [1]   2   4  13  16  27  32  38  47  49  56  64  80  82  86  90  95  98 101 103
## [20] 105 107 109 111 113 123 126 134 137 139 145
```

```
v2.bs <- cpt.meanvar(v2, test.stat = 'Poisson', method = 'BinSeg')
```

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

```
plot(v2.bs, cpt.width = 3, main = 'BinSeg changes in Mean Variance')
```






```
cpts.ts(v2.bs)
```

```
## [1]  49  81  98 109 137
```

augmented dicky fuller test yields -4.97, less than p-value of 0.01; reject null; insufficient evidency to claim time series is NOT stationary basically it is stationary

feed into modeling block

modelling block parts -

AIC5.wge returns the 5 lowest AIC values, used to determine the number of p and q coefficients for the AutoRegressive and Moving Average parts of ARIMA, respectively

est.arma.wge takes the time series object and the number of p and q terms as arguments to estimate the coefficients for those p and q terms

fore.aruma.wge takes the coeficients and forecasts a specified number of units forwards

insert definitions for ASE(mean squared error), accuracy, mape, ect

```
#model fitting with tdata
pspan = 0:10 #range of values to look for possible p and q coefficients for AR and MA
qspan = 0:8
difference = 0
aic_results <- aic5.wge(tdata, p = pspan, q = qspan)
```

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
```

```r
bic_results <- aic5.wge(tdata, p = pspan, q = qspan, type = 'bic')
```

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  bic
```

```r
aic_results
```

```
##      p    q        aic
## 32   3    4   7.681212
## 16   1    6   7.683209
## 34   3    6   7.684040
## 23   2    4   7.686687
## 17   1    7   7.690025
```

```r
bic_results
```

```
##      p    q        bic
## 21   2    2   7.802110
## 14   1    4   7.814297
## 30   3    2   7.826343
## 23   2    4   7.827183
## 15   1    5   7.832998
```

```r
m1 = est.arma.wge(tdata, p = aic_results[1, 1], q = aic_results[1, 2], factor = TRUE) #feed top AIC or
```

```
##
## Coefficients of Original polynomial:
## 1.1622 -0.0434 -0.2316
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.5366B+0.6187B^2    1.2419+-0.2722i      0.7866       0.0343
## 1+0.3744B             -2.6710               0.3744       0.5000
##
##
```

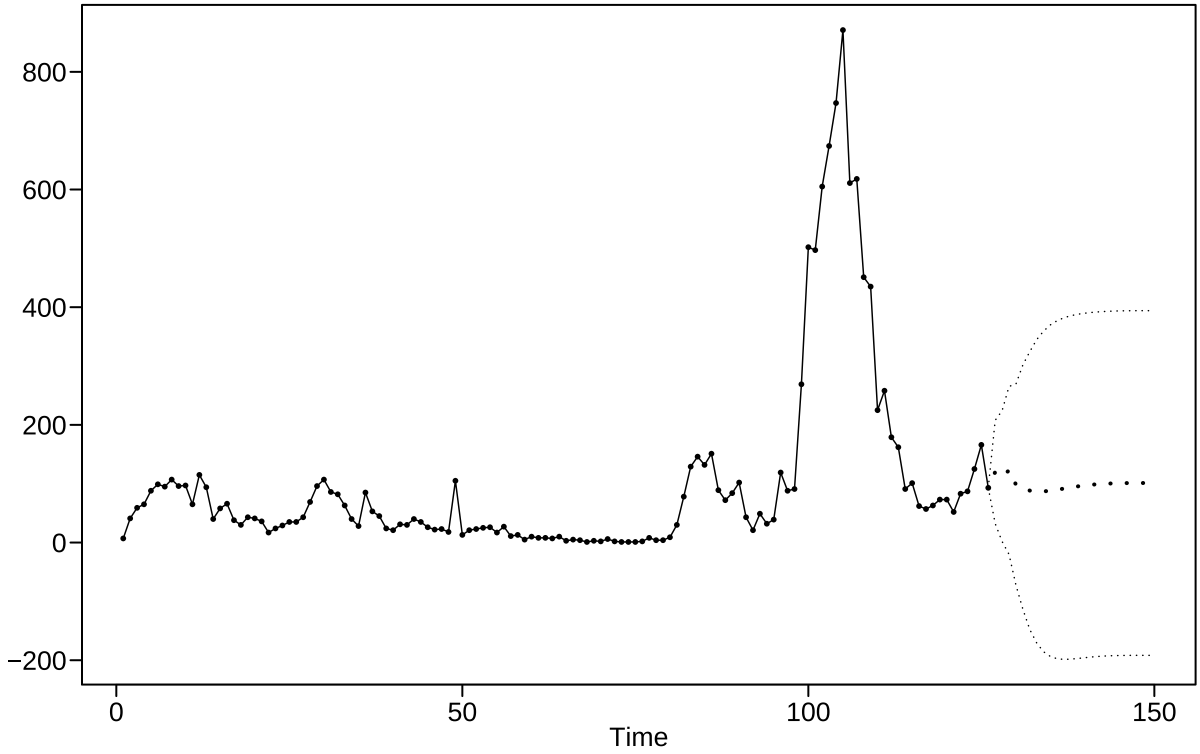
```r
m1$phi
```

```
## [1]  1.16224819 -0.04338446 -0.23162370
```

```r
m1$theta
```

```
## [1]  0.3861406 -0.1622587 -0.1475116 -0.3263750
```

```r
#simple single forecast and test
weeks_compare = 24 #how many weeks to reserve for testing : 6months
f1 <- tdata[1:(length(tdata)-weeks_compare)] %>% fore.aruma.wge(phi = m1$phi, theta = m1$theta, d = 0, 
```

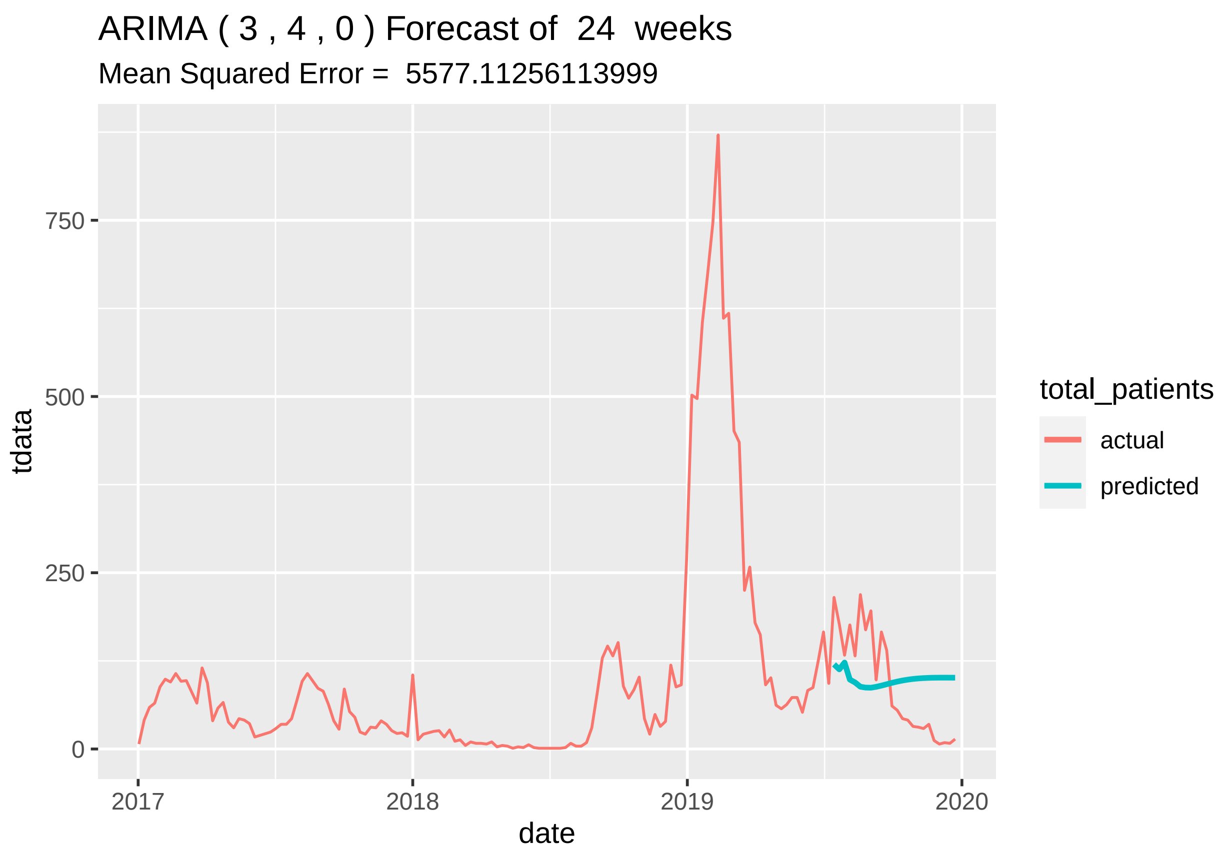



```r
mse <- mean((tdata[((length(tdata)-weeks_compare)+1):(length(tdata))] - f1$f)^2)
paste('Mean Squared Error: ', mse)
```

```
## [1] "Mean Squared Error:  5577.11256113999"
```

```r
#additional metrics
a_metrics <- accuracy(f1$f, tdata[((length(tdata)-weeks_compare)+1):(length(tdata))] )

#change x and y line names to match whatever dataframe you've stored the time series in
timeFrame <- data.frame(date = ttime, inf_cases = tdata)
g3 <- ggplot(timeFrame)+
  geom_line(aes(x = date, y = tdata, color = 'black'), size = 0.5)+
  geom_line(aes(x = date, y = c(rep(NA, length(tdata)-weeks_compare), f1$f), color = 'red'), size= 1) +
  scale_color_discrete(name = "total_patients", labels = c('actual', 'predicted')) +
  ggtitle(paste('ARIMA (',aic_results[1, 1],',',aic_results[1, 2],',',difference,')', 'Forecast of ', we
g3
```

```
## Warning: Removed 126 row(s) containing missing values (geom_path).
```



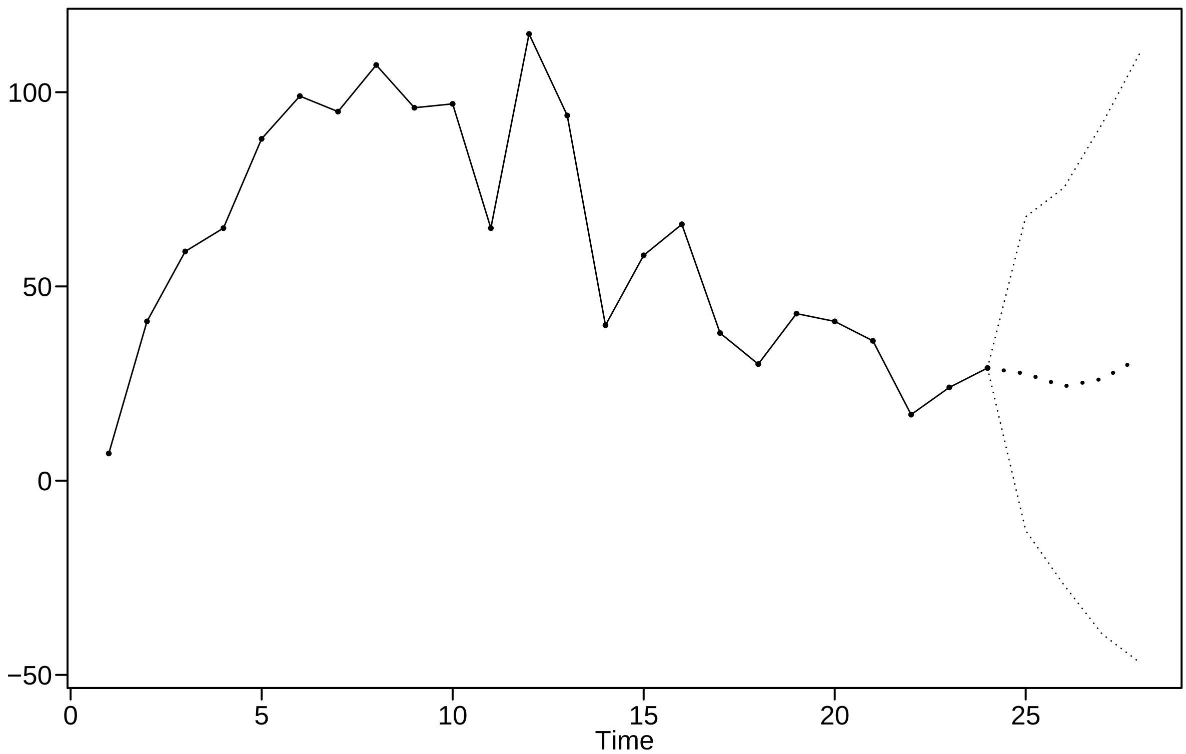
training size = how many weeks to use as training data horizon = how many weeks to use as testing data step_size - how far to move the "window" every iteration

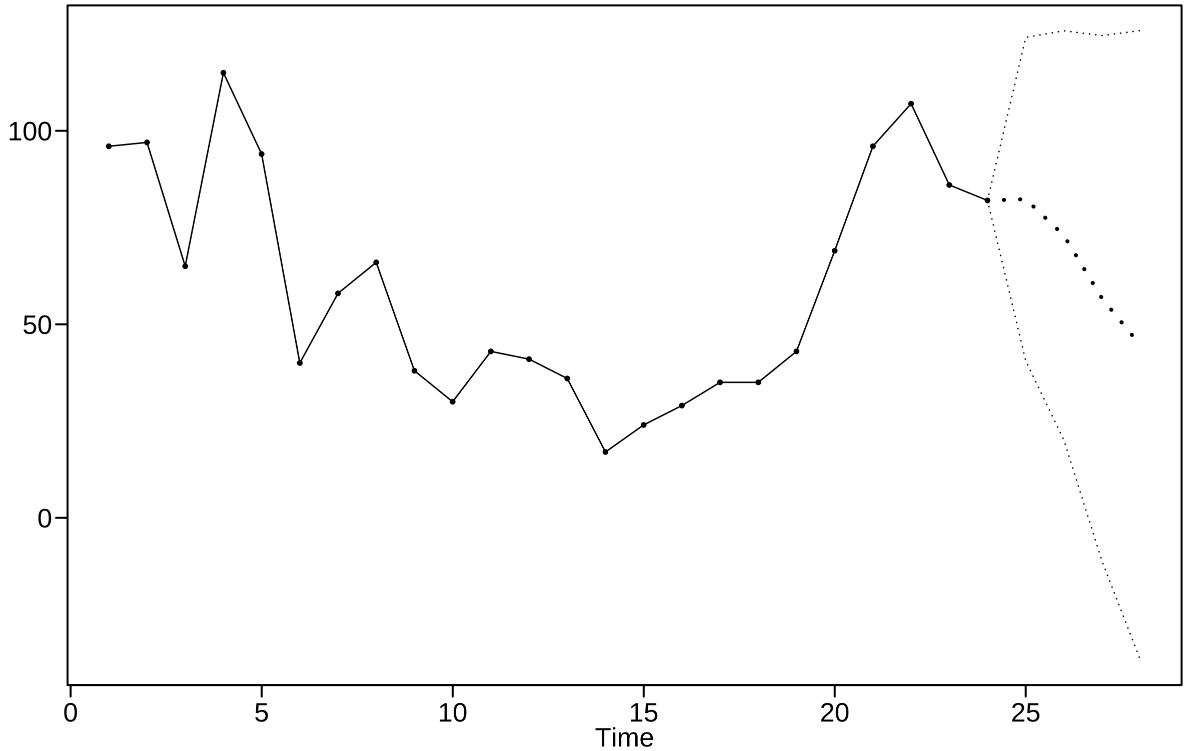
breaks up the time series object into a slices of specified length, evaluate using the mean of all ASE at the end

lets you know how well a model generalizes across the "body" of the time series, or if it only good before a spike, when stationary, ect

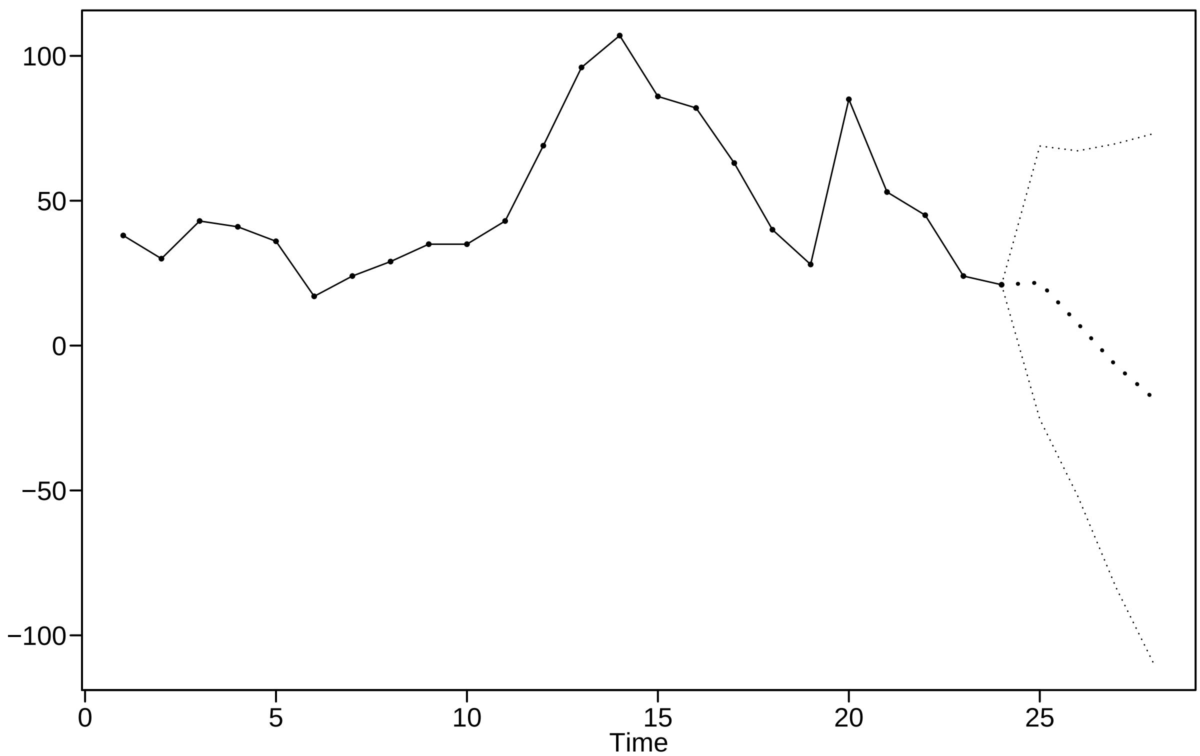
```
#updated with step size,how much the "window" moves per interval
#rolling window ase: apply model predictions to small segments of the given data and aggregate metrics
trainingSize = 24
horizon = 4
step_size = 8
n_windows = round((length(tdata)-(trainingSize + horizon))/step_size) #number of whole windows of train
ASEHolder = numeric()

fcastHolder = matrix(nrow = n_windows, ncol = horizon) #create matrix that has the number of rows and c
fUpperHolder = matrix(nrow = n_windows, ncol = horizon) #same for upper and lower intervals
fLowerHolder = matrix(nrow = n_windows, ncol = horizon)
phis = m1$phi
thetas = m1$theta
s = 0
d = 0

for( i in 1:n_windows) #how many "windows" can fit into the total length of time, rounding down
{
```

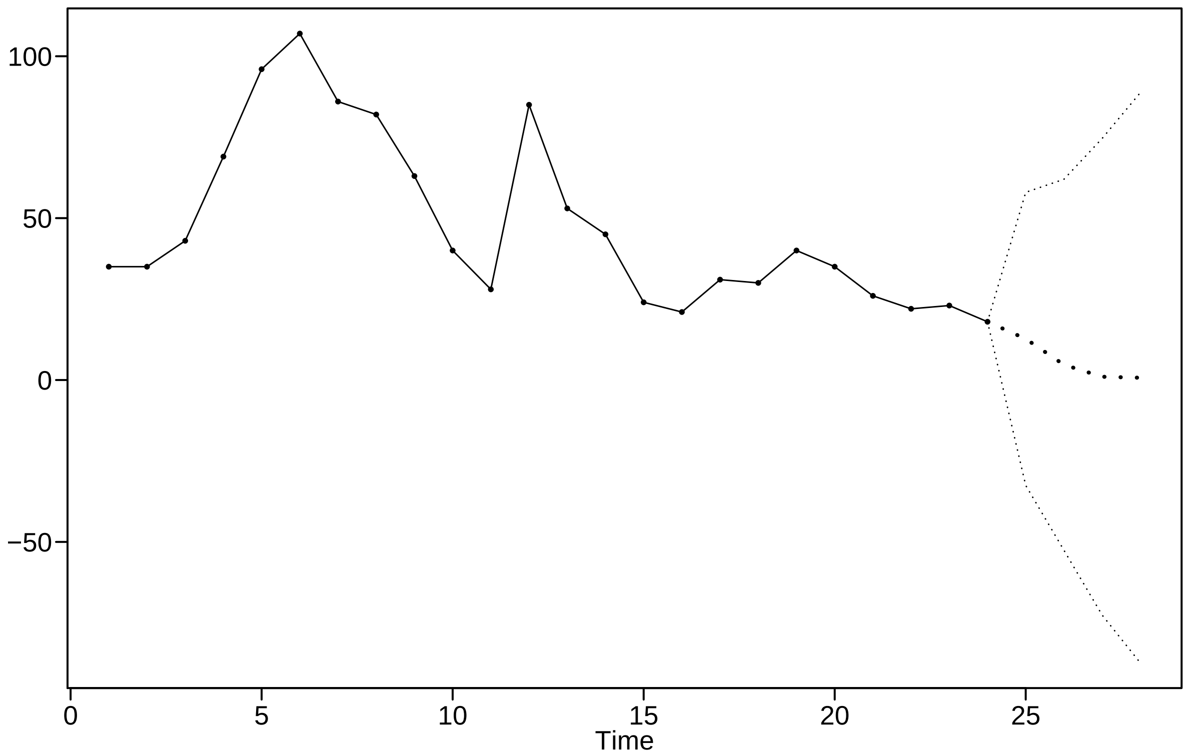
```
  t_start = 1+(step_size*(i-1)) #starting point for each window's training data
  t_end = trainingSize + (step_size*(i-1)) #endpoint
  forecasts = fore.aruma.wge(tdata[t_start:t_end],phi = phis, theta = thetas, s = s, d = d, n.ahead = ho

  ASE = mean((tdata[(t_end + 1):(t_end+horizon)] - forecasts$f)^2)
  ASEHolder[i] = ASE
  fcastHolder[i, ] <- forecasts$f
  fUpperHolder[i, ] <- forecasts$ul
  fLowerHolder[i, ] <- forecasts$ll
  #print(paste(t_start, t_end))
}
```

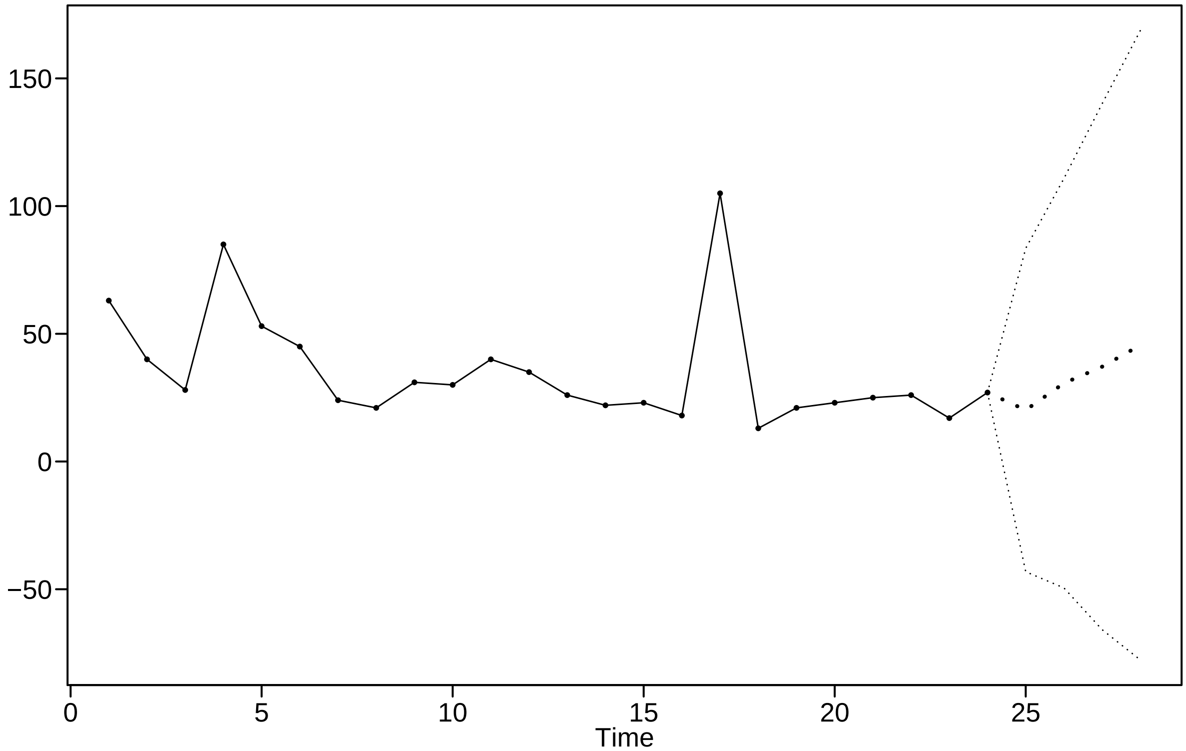




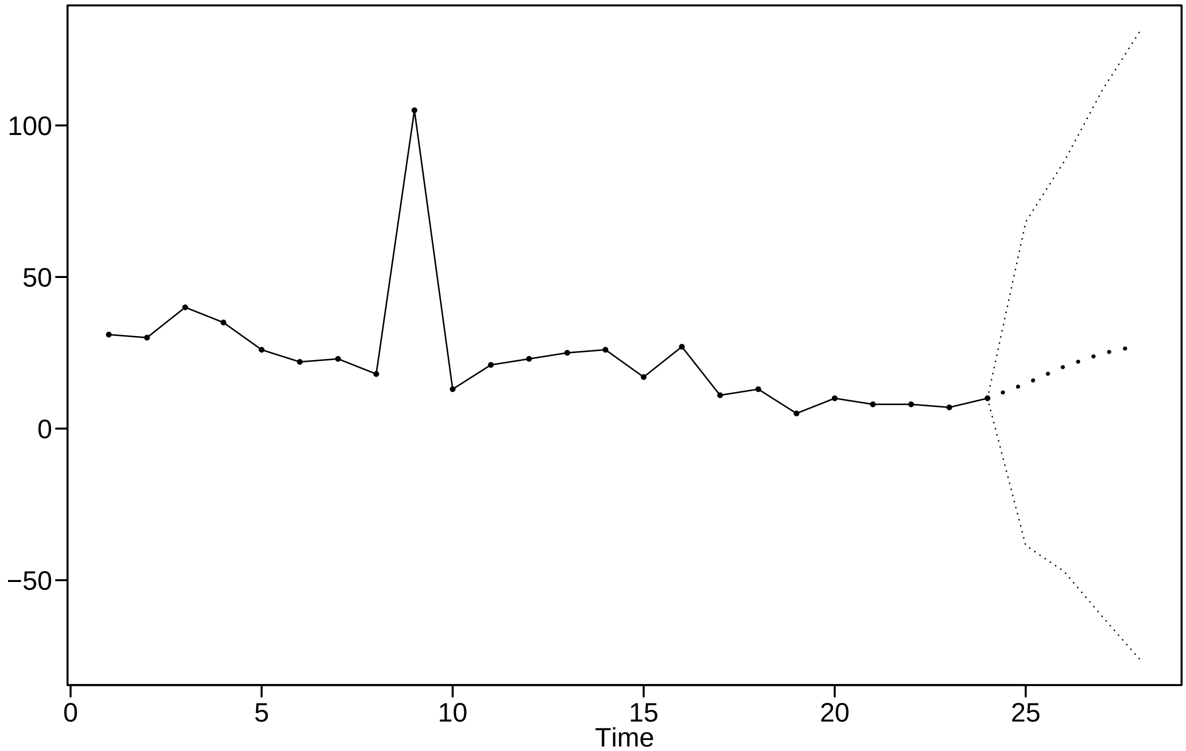
100
50
0
0
5
10
15
20
25
Time

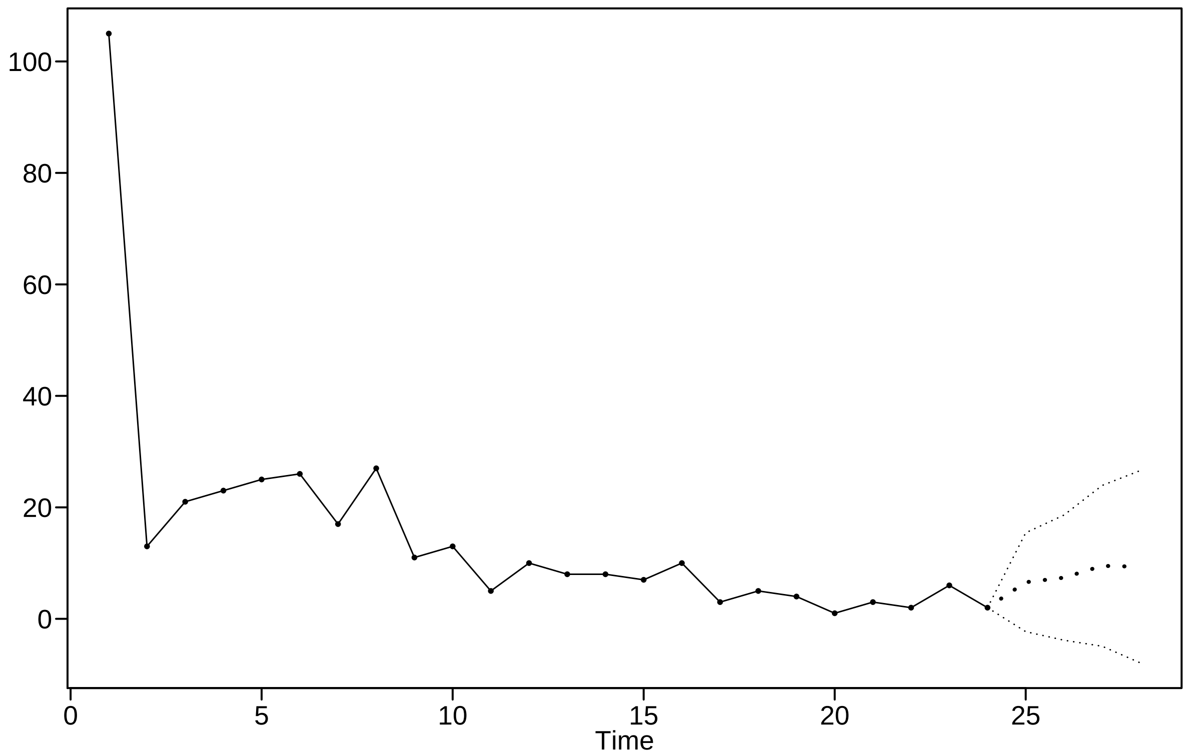

100
50
0
−50
−100
0
5
10
15
20
25
Time


100
50
0
−50
0
5
10
15
20
25
Time

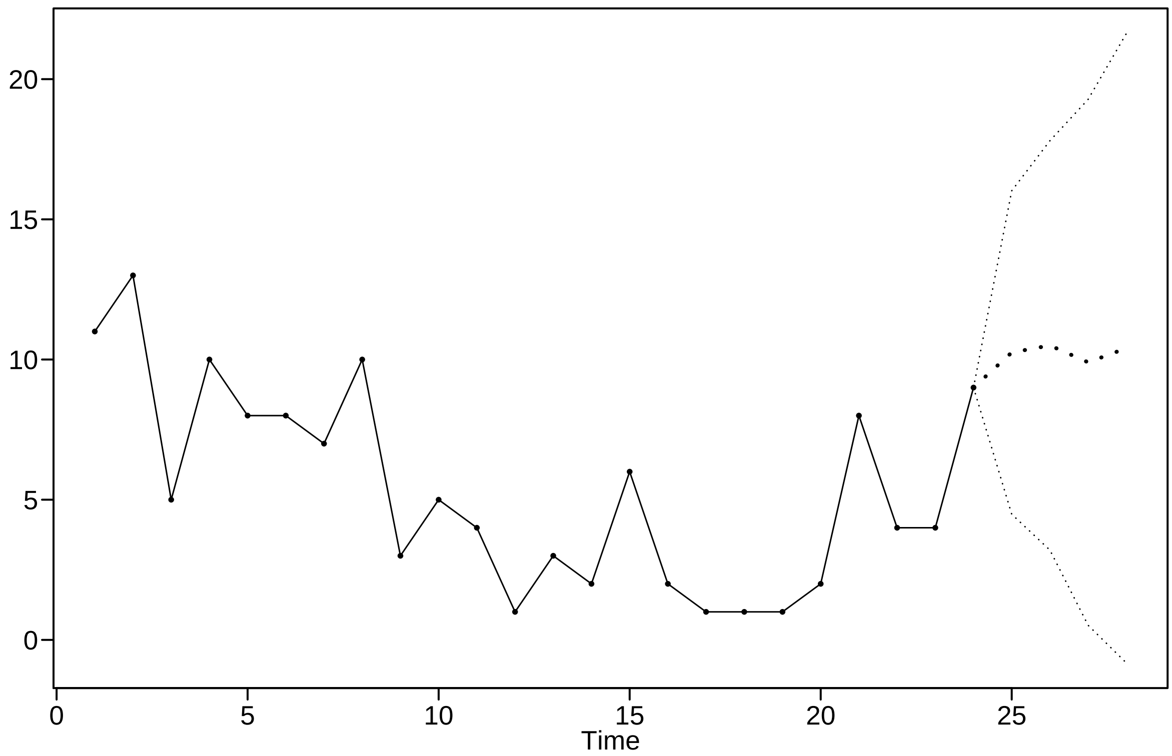


150
100
50
0
−50
0
5
10
15
20
25
Time

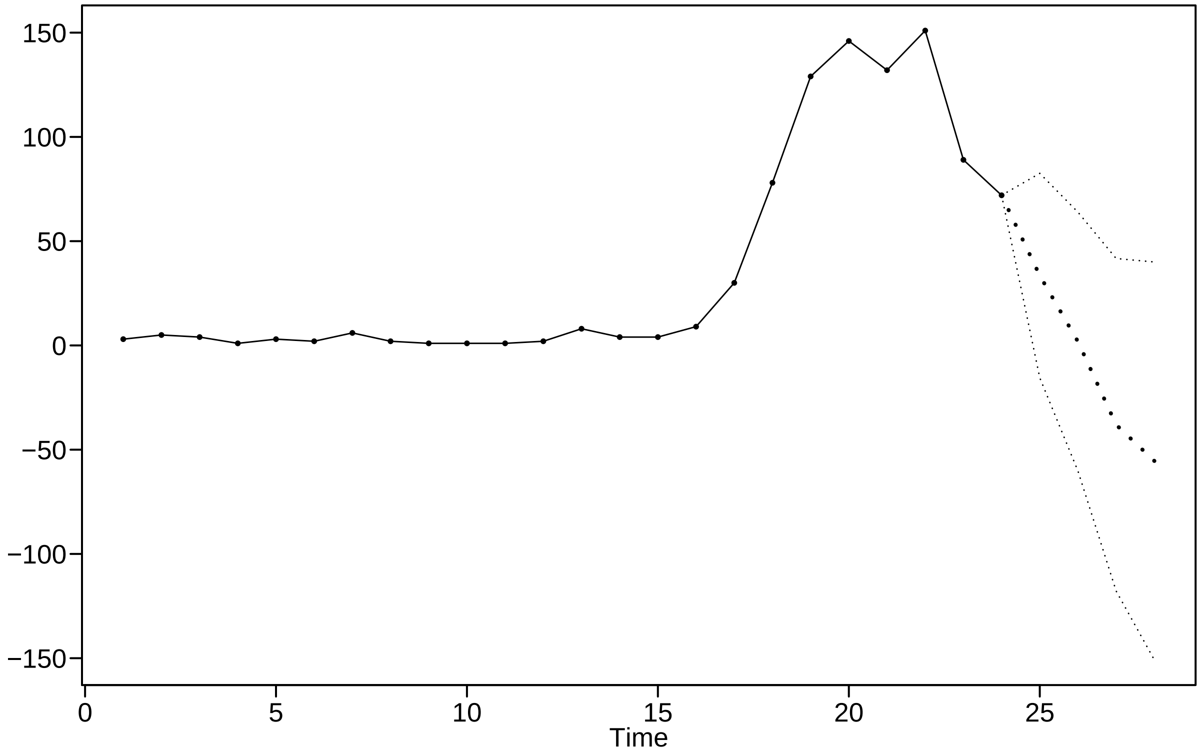

100
50
0
−50
0
5
10
15
20
25
Time

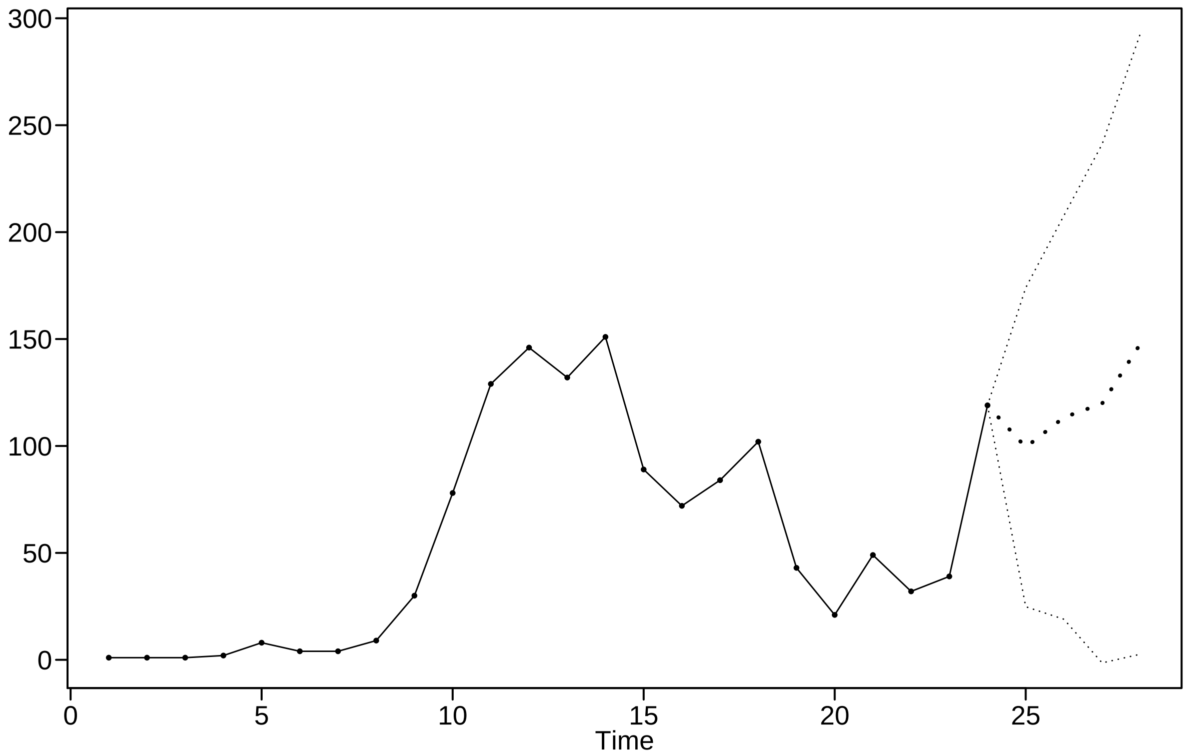


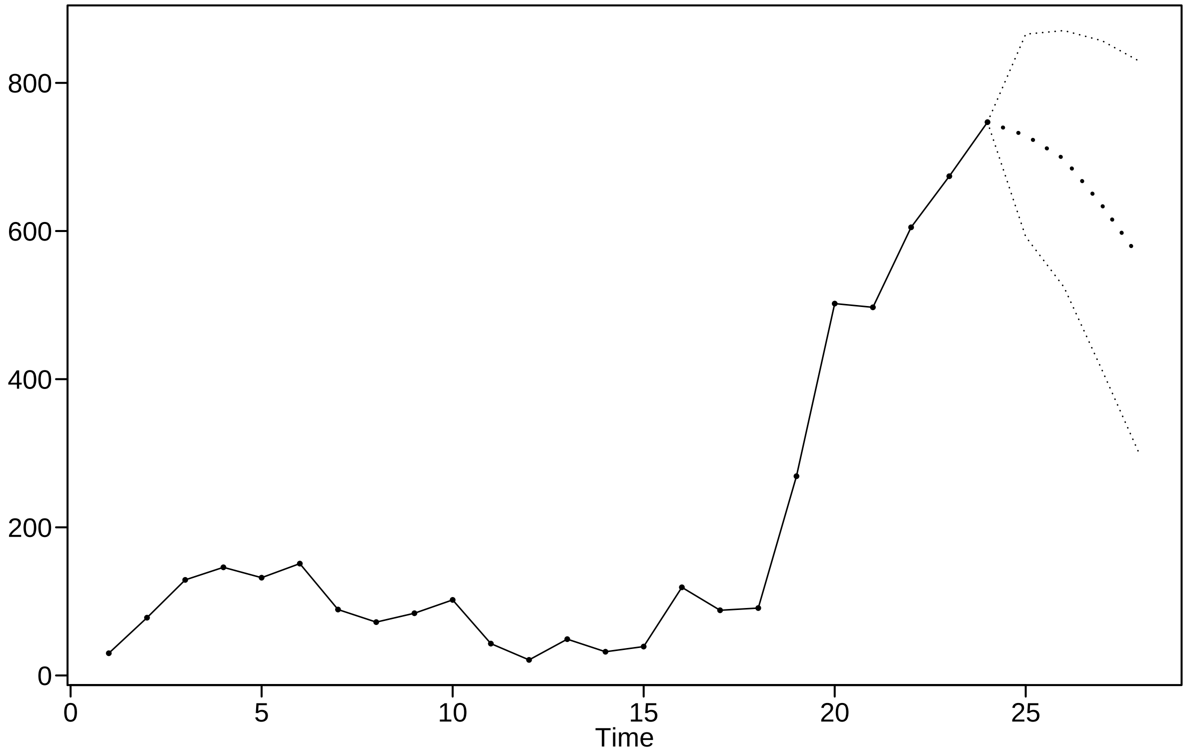
100
80
60
40
20
0
0
5
10
15
20
25
Time

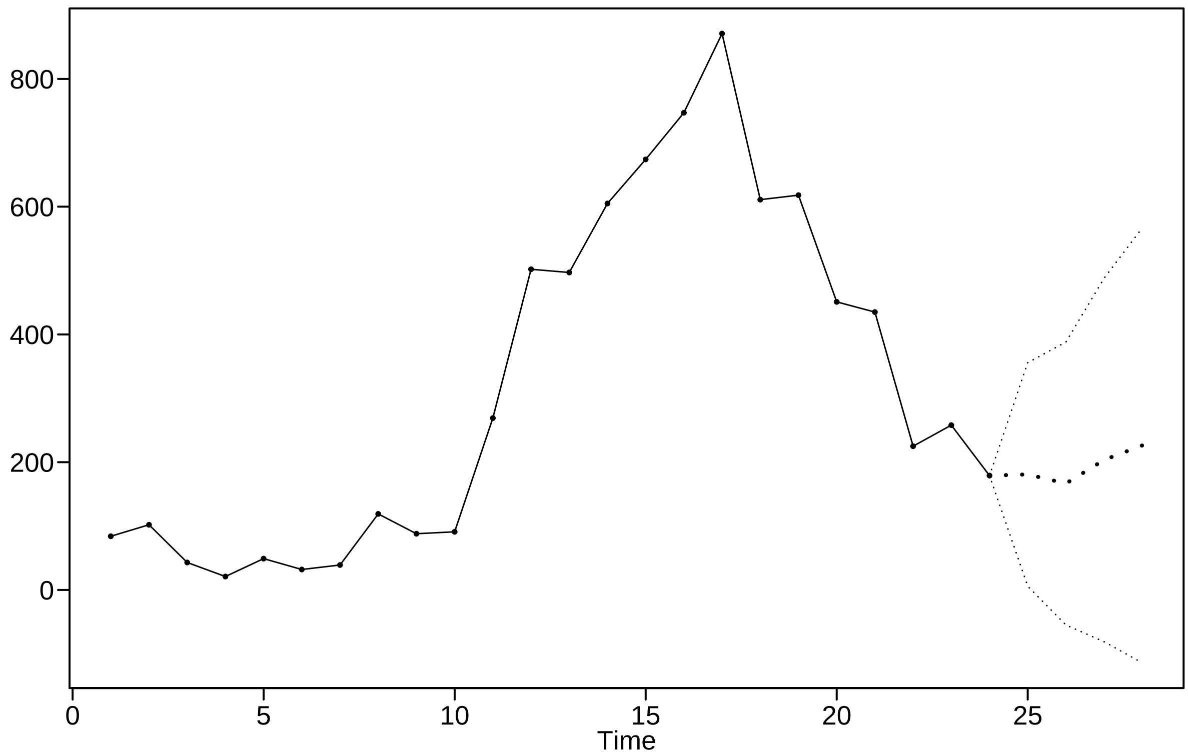


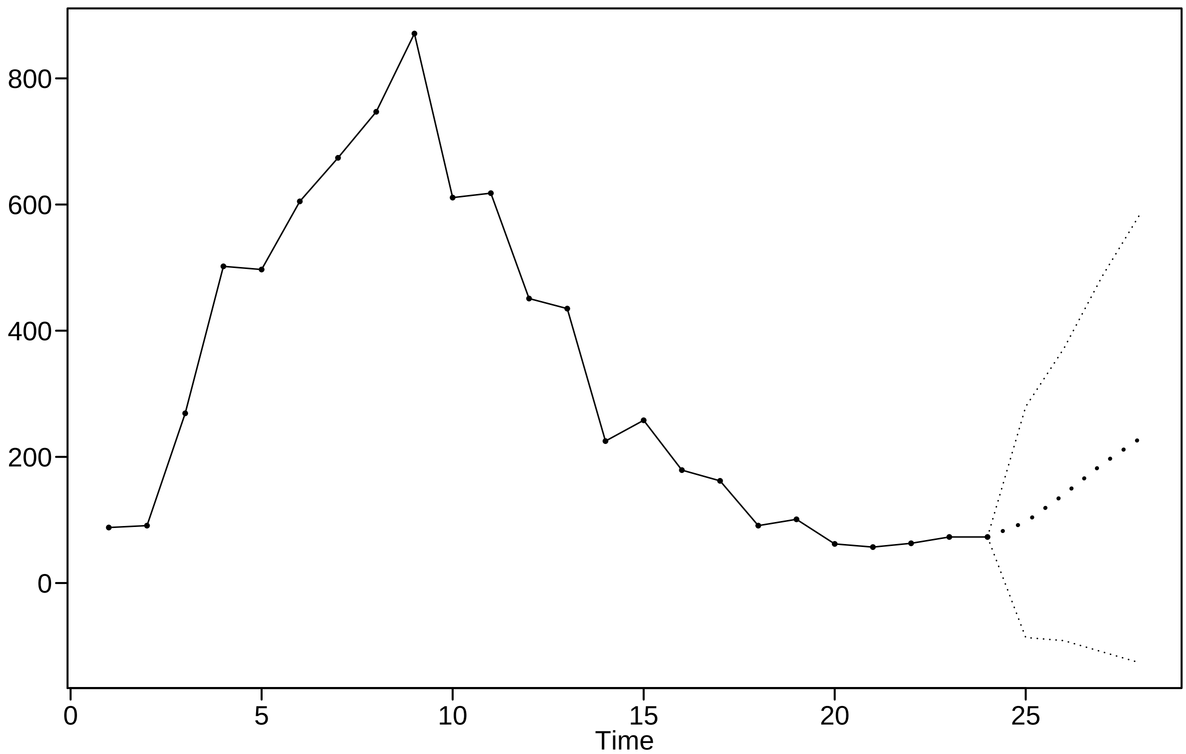
20
15
10
5
0
0
5
10
15
20
25
Time

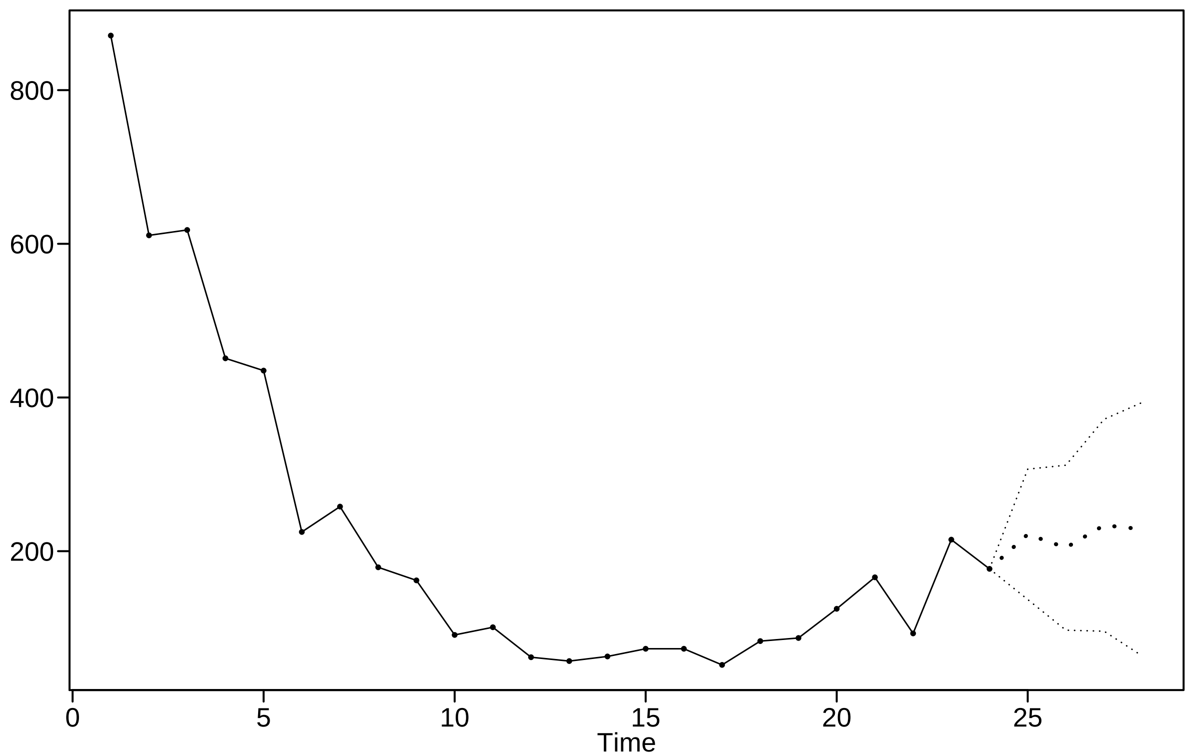

150
100
50
0
−50
−100
−150
0
5
10
15
20
25
Time


300
250
200
150
100
50
0
0
5
10
15
20
25
Time

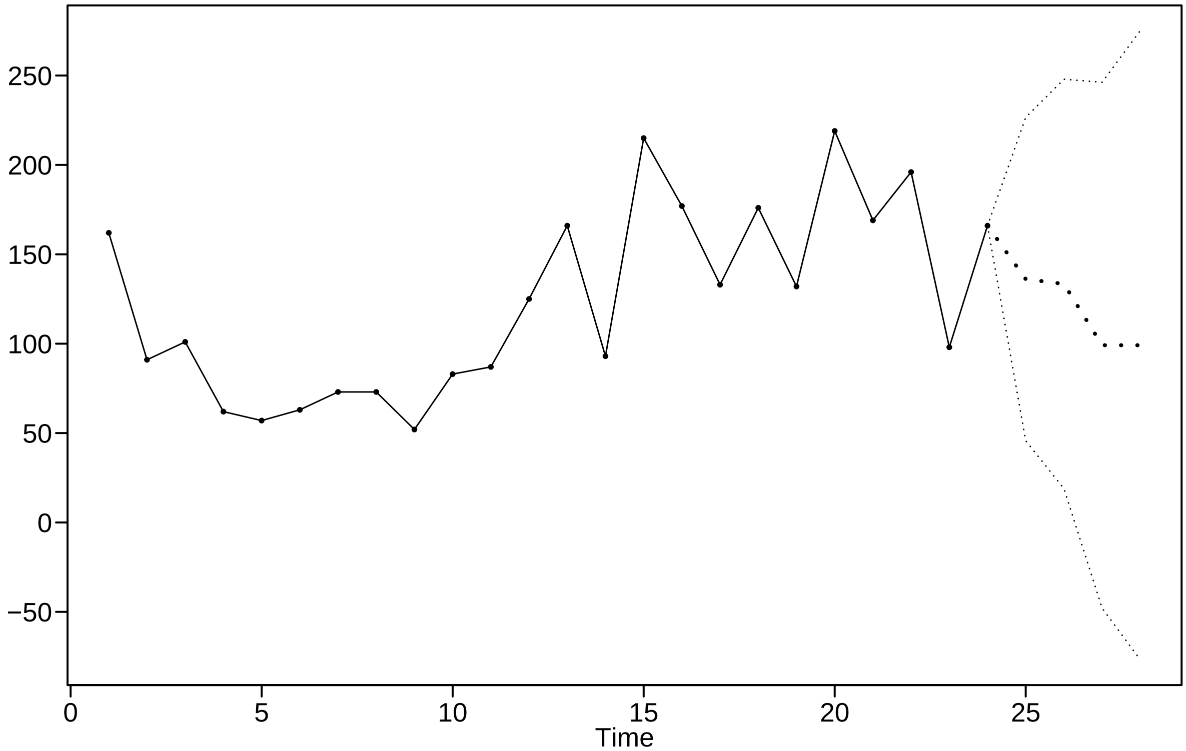

800
600
400
200
0
0
5
10
15
20
25
Time


800
600
400
200
0
0
5
10
15
20
25
Time


800
600
400
200
0
0
5
10
15
20
25
Time



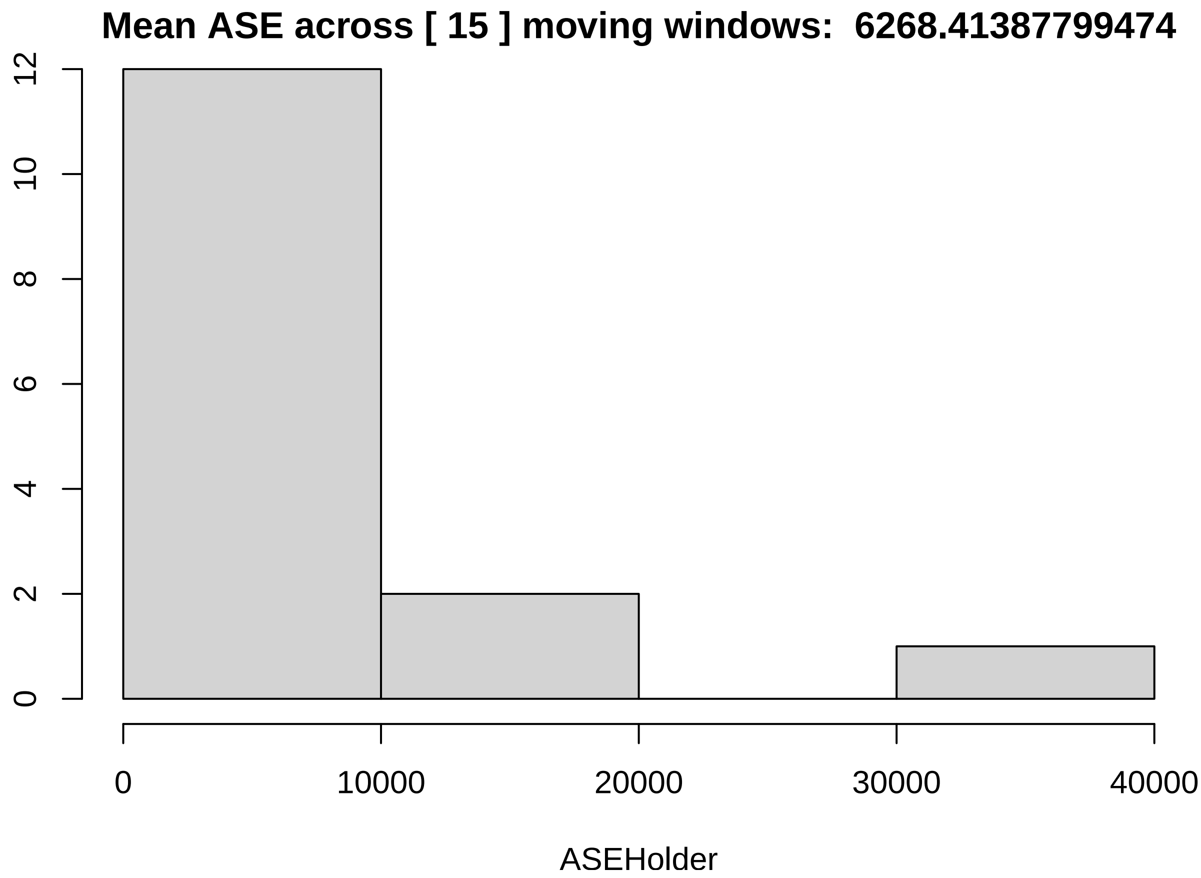
800
600
400
200
0
5
10
15
20
25
Time




```r
ASEHolder
```

```
##  [1]   461.19694   978.48580  1413.84230  2370.38991   672.18770   377.92029
##  [7]    49.51142  9379.33976  6248.70551 37096.95415 10080.37904 10891.45100
## [13]  6623.50866  4791.81958  2590.51611
```

```r
WindowedASE = mean(ASEHolder)
hist(ASEHolder, main = paste("Mean ASE across [", n_windows, ’] moving windows: ’, WindowedASE))
```




```
summary(ASEHolder)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    49.51   825.34  2590.52  6268.41  8001.42 37096.95
```

```
print(paste("Mean ASE across ", n_windows, ' moving windows: ', WindowedASE))
```

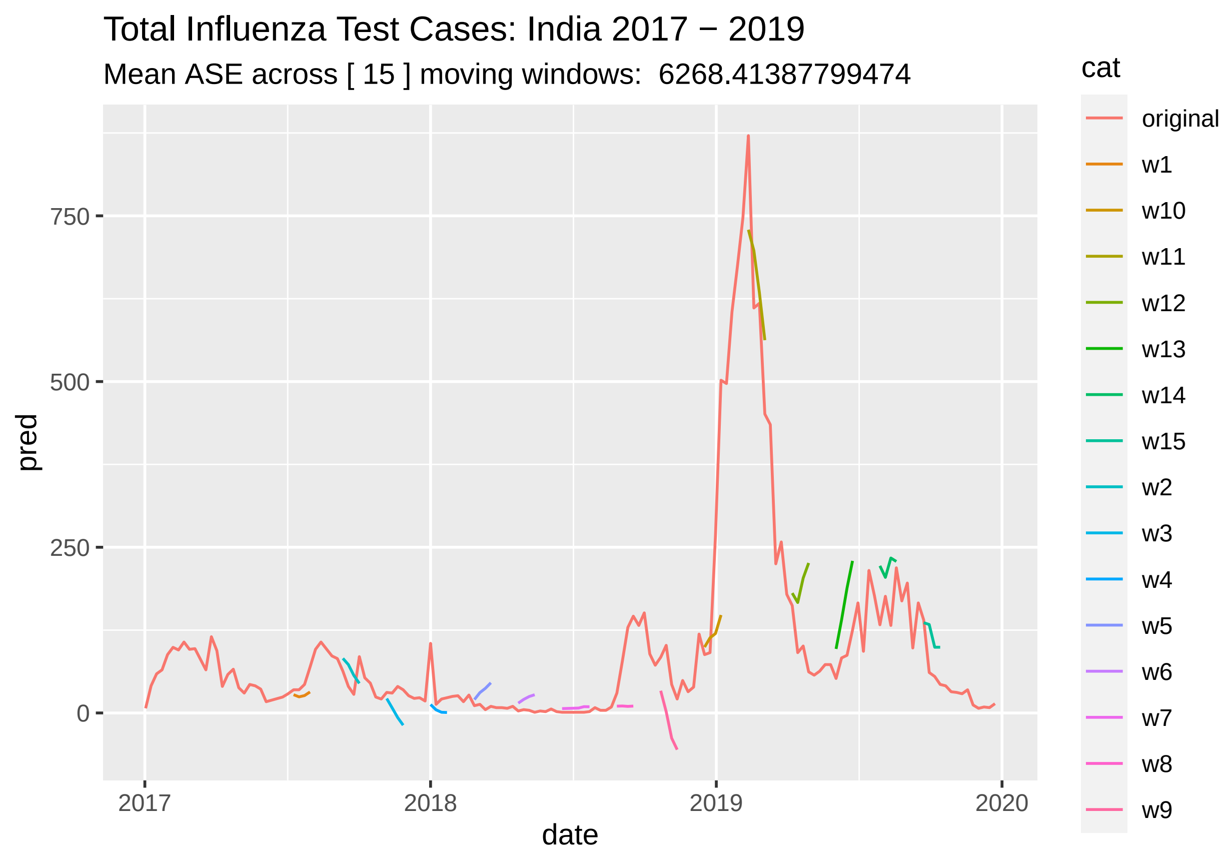
```
## [1] "Mean ASE across  15  moving windows:  6268.41387799474"
```

```
#function to space out forecasts with NAs for plotting
fspacer <- function(fcasts){
  cast_frame = 0
  cast_frame = matrix(nrow = n_windows, ncol = length(tdata)) #empty matrix to hold spaced out forecast
  for(i in 1:n_windows)
    {
    #print(i)
    before_block <- rep(NA, (trainingSize + (step_size*(i-1))))
    after_block <- c(rep(NA, (length(tdata) - length(c(rep(NA, (trainingSize + step_size*(i-1))))) - ho
    #print(paste(length(before_block), length(after_block)))
    #print(length(c(before_block, fcasts[i, ], after_block)))
    cast_frame[i, ] <- c(before_block, fcasts[i, ], after_block)
    #print(cast_frame[i, ])
    }
  return(cast_frame)
```

```
}
f_spaced <- fspacer(fcastHolder)
```

```
#create categorical dataframe to store windowed predictions
f_spaced <- fspacer(fcastHolder)
fframe <- list()
for(i in 1:n_windows){
  #name = paste0('df', i)
  lname = paste0('w', i)
  #assign(name, data.frame(date = ttime, pred = f_spaced[i, ]))
  tmp <- data.frame(date = ttime, pred = f_spaced[i, ])
  fframe[[lname]] <- tmp
  }
f_df <- cbind(cat= rep(names(fframe), sapply(fframe, NROW)), do.call(rbind, fframe)) #arrange data with
o_df <- data.frame(cat = rep('original', length(tdata)), date = ttime, pred = tdata) #add actual data a
f_df <- rbind(f_df, o_df)

ggplot(f_df, aes(date,pred, color= cat)) + geom_line() +
  ggtitle("Total Influenza Test Cases: India 2017 - 2019", subtitle = paste("Mean ASE across [", n_windo
```

```
## Warning: Removed 2190 row(s) containing missing values (geom_path).
```

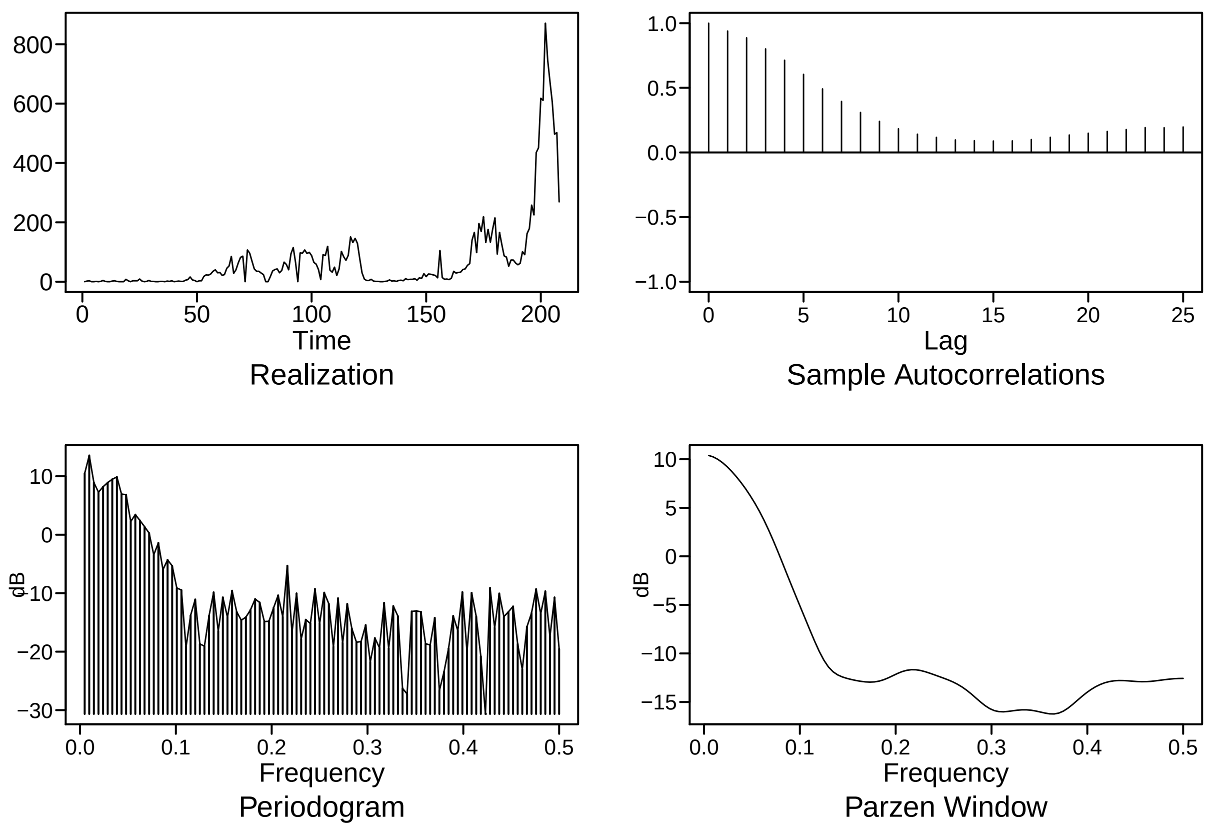



```
year_range = 1995:2019
range_data = vector()
```

```
for(j in year_range[1:5]){
  range_data <- rbind(range_data, t_ind[which(t_ind$Year == j), ])
}
```

#iterable years

```
#dataset generation
S_Asia <- raw_data[which(raw_data$FLUREGION == 'Southern Asia'), ]
t_ind <- S_Asia[which(S_Asia['ï..Country'] == 'India'), ] #leave na, convert to zero
t_ind$ALL_INF[is.na(t_ind$ALL_INF)] <- 0

year_range = 2019:1995 #call with year_range[1], ect
n_years = 4 #this can be a user input type thing? starting from 2019, number of years to include in TS
range_data = vector()

for(j in 1:n_years){
  range_data <- rbind(range_data, t_ind[which(t_ind$Year == year_range[j]), ])
} #there we go, creates a df based on specified index years
tdata <- rev(range_data$ALL_INF) #set primary ts to all influenza cases #IMPORTANT to reverse the data
ttime <- rev(range_data$SDATE) %>% as.Date("%m/%d/%Y") %>% sort()#because it's in descending year from

#### model diagnostics ####
invisible(plotts.sample.wge(tdata)) #wrap objects in invisible() to hide the long console output and le
```

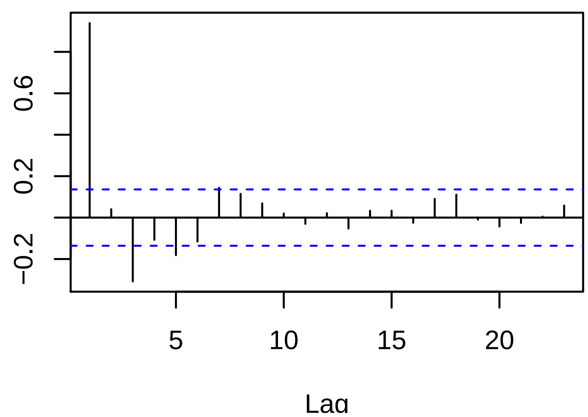


```r
#stationarity testing
adf.test(tdata)
```

```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  tdata
## Dickey-Fuller = -6.0686, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```r
pacf(tdata)
#changepoint
par(mfrow = c(2,1))
```



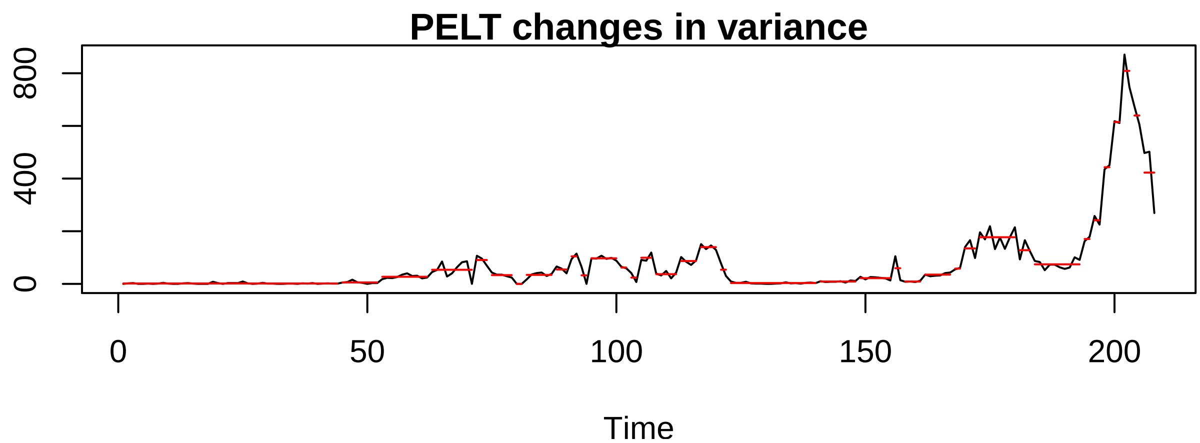

```r
v2 <- tdata #changepoint visualization
v2.pelt <- cpt.meanvar(v2, test.stat = 'Poisson', method = "PELT")
plot(v2.pelt, main = 'PELT changes in variance')
cpts.ts(v2.pelt)
```

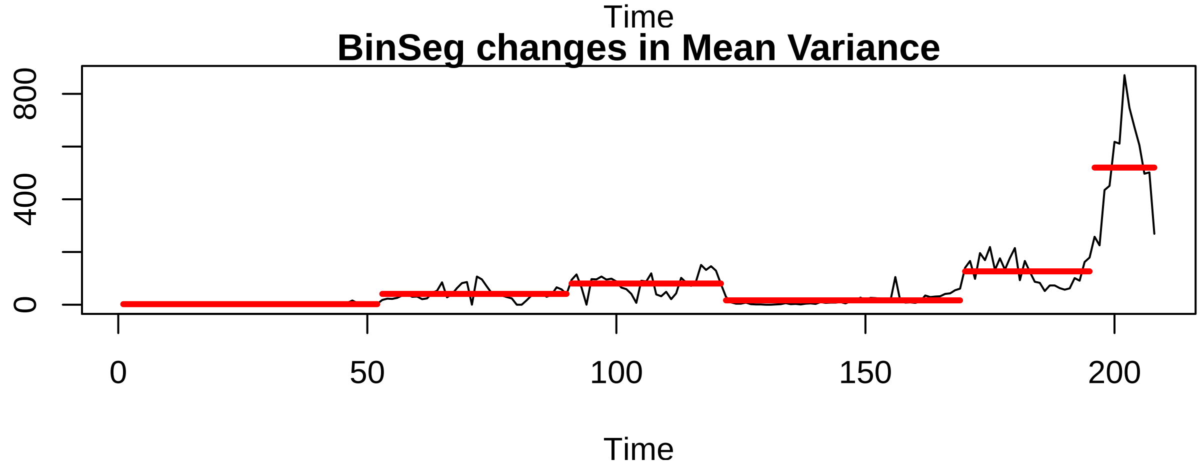
```
##  [1]  44  52  62  71  74  79  81  87  90  92  94 100 102 104 107 112 116 120 122
## [20] 140 148 155 157 161 167 169 172 180 183 193 195 197 199 201 203 205
```

```r
v2.bs <- cpt.meanvar(v2, test.stat = 'Poisson', method = 'BinSeg')
```

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

```r
plot(v2.bs, cpt.width = 3, main = 'BinSeg changes in Mean Variance')
```






```r
cpts.ts(v2.bs)
```

```
## [1]  52  90 121 169 195
```

```r
#### end ####

#### model fitting ####
pspan = 0:5 #range of values to look for possible p and q coefficients for AR and MA
qspan = 0:5
difference = 0
aic_results <- aic5.wge(tdata, p = pspan, q = qspan) #cut bic results
```

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
```

```r
aic_results
```

```
##      p   q       aic
## 29   4   4  7.353360
## 24   3   5  7.364565
## 30   4   5  7.373713
## 11   1   4  7.376669
## 17   2   4  7.382332
```

```r
m1 = est.arma.wge(tdata, p = aic_results[1, 1], q = aic_results[1, 2], factor = TRUE) #feed top AIC int
```

```
##
## Coefficients of Original polynomial:
## 0.4298 0.9433 0.1995 -0.7044
##
## Factor                   Roots                 Abs Recip    System Freq
## 1-1.8492B+0.8903B^2     1.0386+-0.2112i       0.9435       0.0319
## 1+1.4194B+0.7912B^2    -0.8970+-0.6777i       0.8895       0.3970
##
##
```

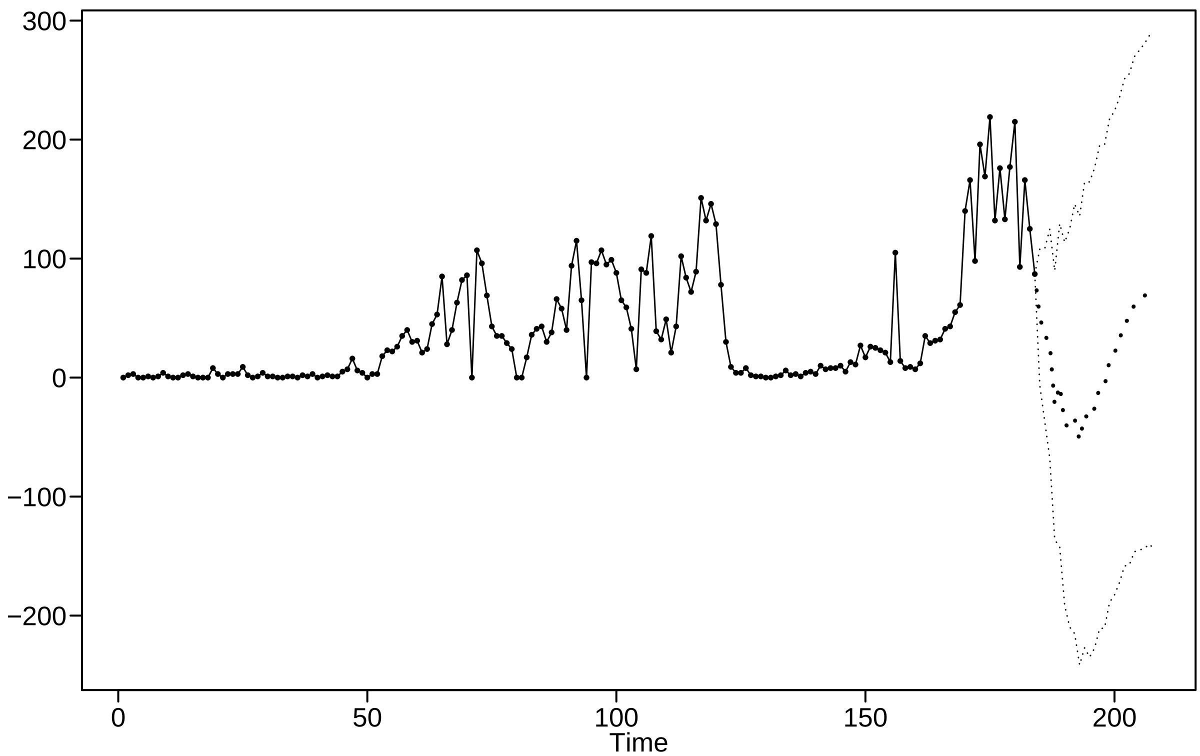
```r
m1$phi
```

```
## [1]  0.4298149  0.9433382  0.1994705 -0.7043920
```

```r
m1$theta
```

```
## [1] -0.3553471  0.1804677  0.3758103 -0.3824985
```

```r
#### end ####


#### endpoint forcast ####
#simple single forecast and test
weeks_compare = 24 #how many weeks to reserve for testing : 6months
f1 <- tdata[1:(length(tdata)-weeks_compare)] %>% fore.aruma.wge(phi = m1$phi, theta = m1$theta, d = 0, r
```

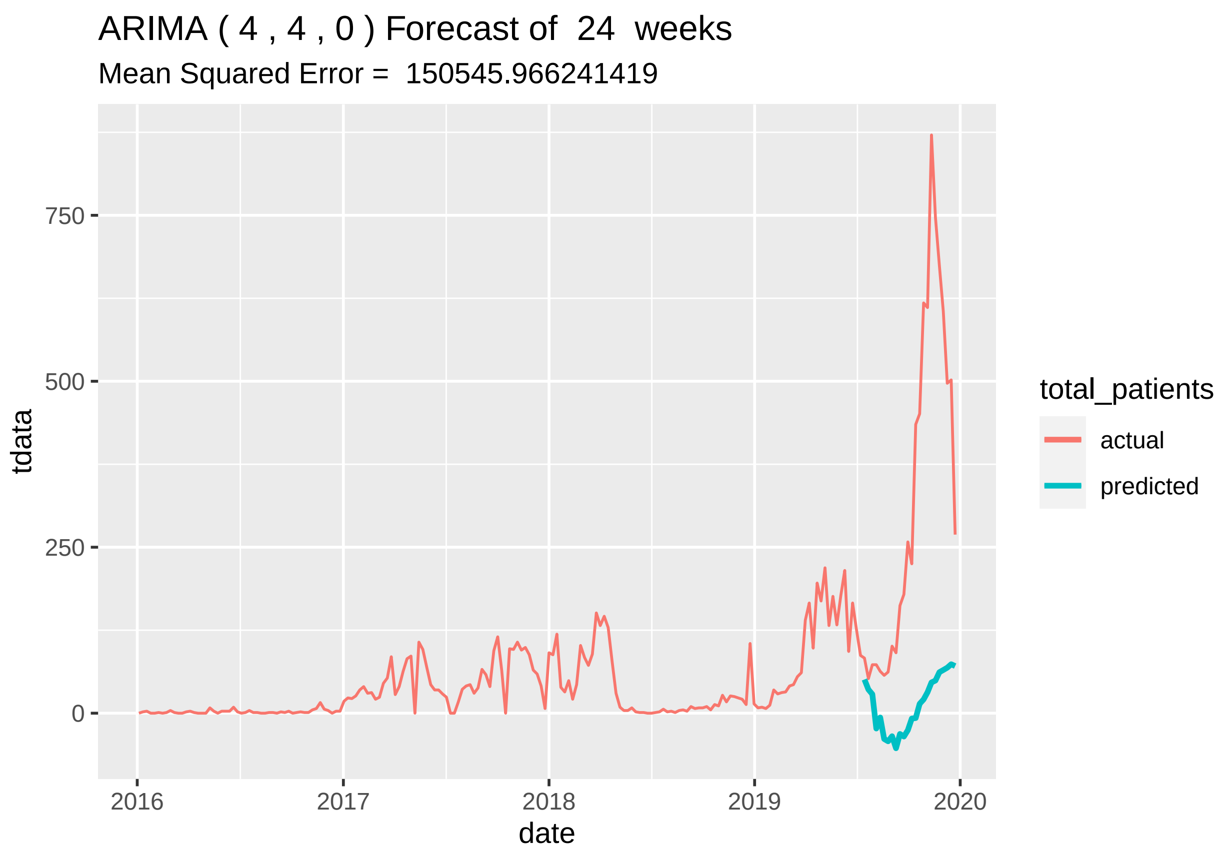



```r
mse <- mean((tdata[((length(tdata)-weeks_compare)+1):(length(tdata))] - f1$f)^2)
paste('Mean Squared Error: ', mse)
```

```
## [1] "Mean Squared Error:  150545.966241419"
```

```r
#additional metrics
a_metrics <- accuracy(f1$f, tdata[((length(tdata)-weeks_compare)+1):(length(tdata))] )

#change x and y line names to match whatever dataframe you've stored the time series in
timeFrame <- data.frame(date = ttime, inf_cases = tdata)
g3 <- ggplot(timeFrame)+
  geom_line(aes(x = date, y = tdata, color = 'black'), size = 0.5)+
  geom_line(aes(x = date, y = c(rep(NA, (length(tdata)-weeks_compare)), f1$f), color = 'red'), size= 1)
  scale_color_discrete(name = "total_patients", labels = c('actual', 'predicted')) +
  ggtitle(paste('ARIMA (',aic_results[1, 1],',',aic_results[1, 2],',',difference,')', 'Forecast of ', we
g3
```

```
## Warning: Removed 184 row(s) containing missing values (geom_path).
```

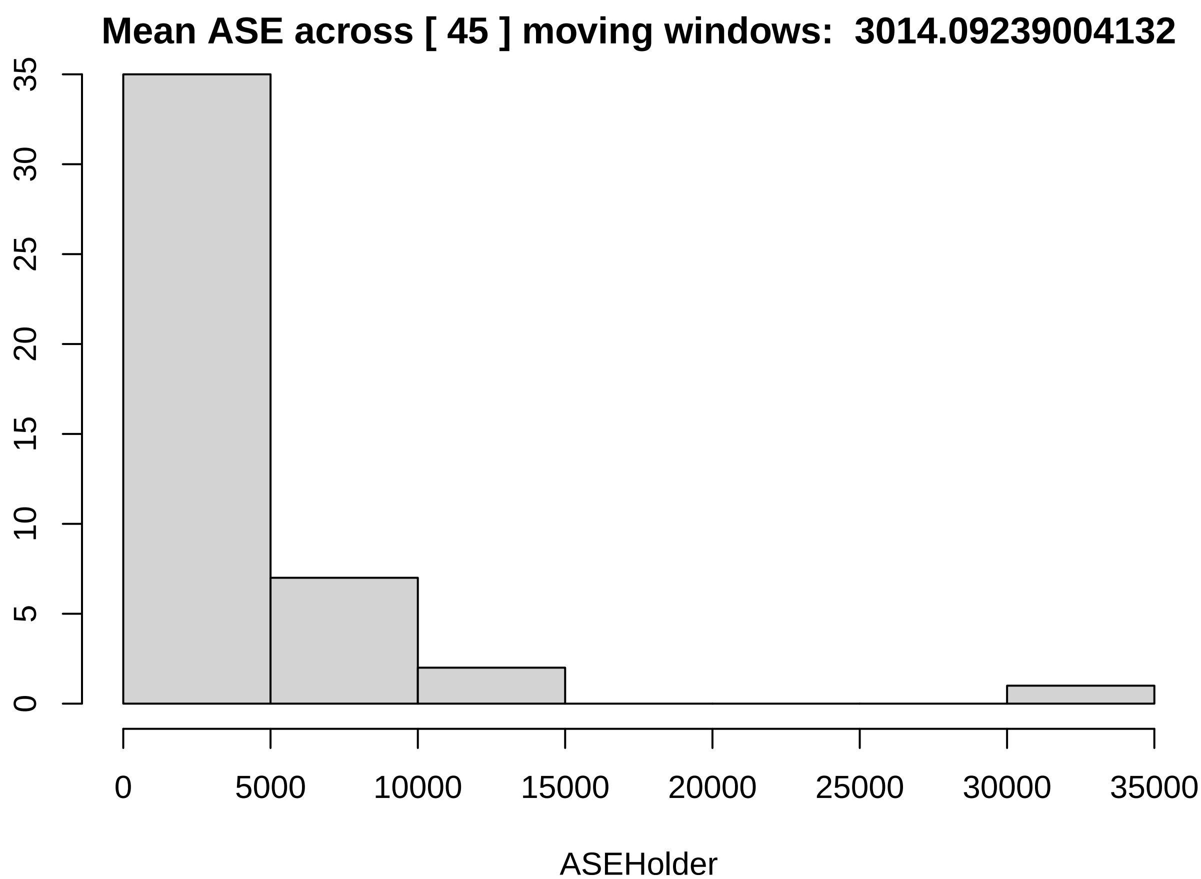



```
#### end ####

#### rolling window ase and viz ####
#rolling window ase: apply model predictions to small segments of the given data and aggregate metrics
trainingSize = 24
horizon = 4 #3months
step_size = 4
n_windows = round((length(tdata)-(trainingSize + horizon))/step_size) #number of whole windows of train
ASEHolder = numeric()

fcastHolder = matrix(nrow = n_windows, ncol = horizon) #create matrix that has the number of rows and c
fUpperHolder = matrix(nrow = n_windows, ncol = horizon) #same for upper and lower intervals
fLowerHolder = matrix(nrow = n_windows, ncol = horizon)
phis = m1$phi
thetas = m1$theta
s = 0
d = 0
for( i in 1:n_windows) #how many "windows" can fit into the total length of time, rounding down
{
  t_start = 1+(step_size*(i-1)) #starting point for each window's training data
  t_end = trainingSize + (step_size*(i-1)) #endpoint
  forecasts = fore.aruma.wge(tdata[t_start:t_end],phi = phis, theta = thetas, s = s, d = d, n.ahead = ho
  ASE = mean((tdata[(t_end + 1):(t_end+horizon)] - forecasts$f)^2)
  ASEHolder[i] = ASE
  fcastHolder[i, ] <- forecasts$f
  fUpperHolder[i, ] <- forecasts$ul
```

```
  fLowerHolder[i, ] <- forecasts$ll
}
summary(ASEHolder)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     1.31   136.74   781.16  3014.09  2970.89 32876.32
```

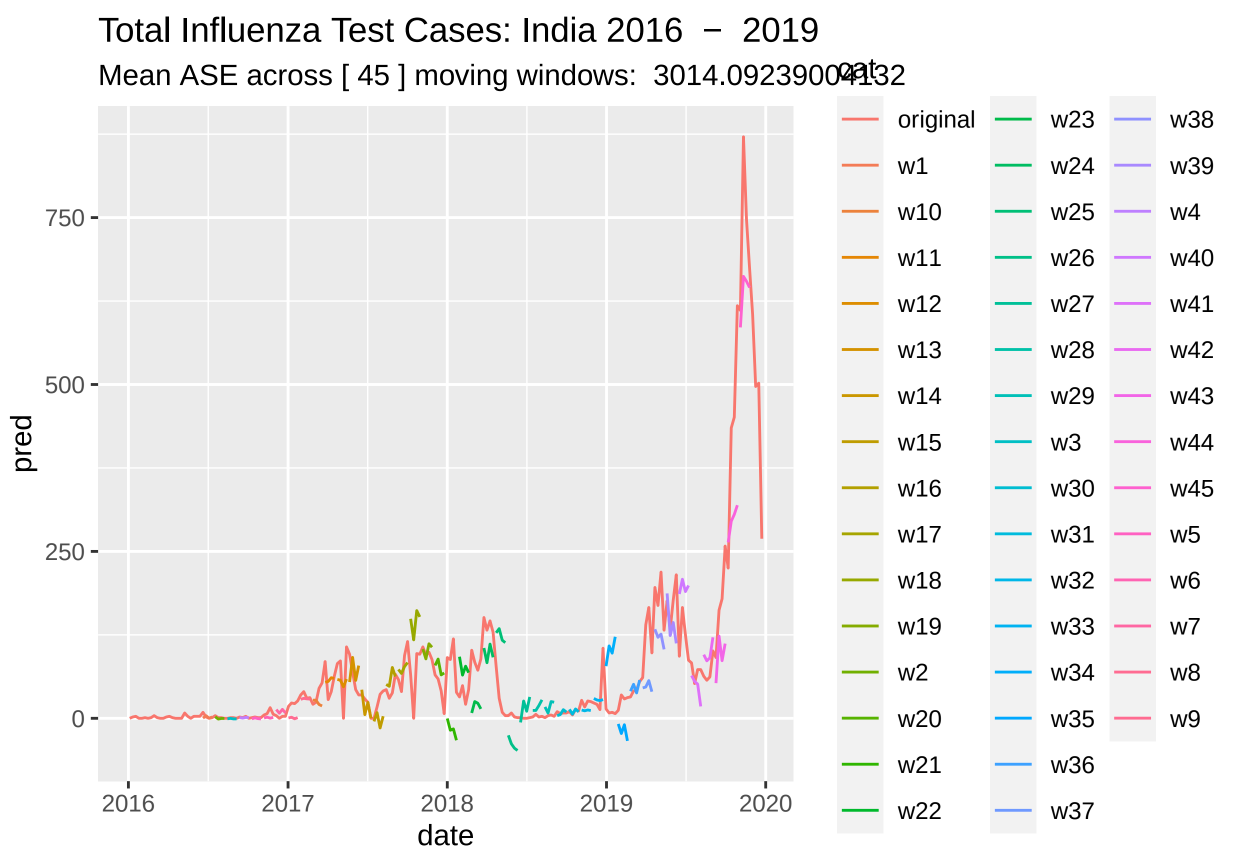
```
WindowedASE = mean(ASEHolder)
hist(ASEHolder, main = paste("Mean ASE across [", n_windows, '] moving windows: ', WindowedASE))
```




```
#### end ####

#### create categorical dataframe to store windowed predictions and visualize with ggplot ####
f_spaced <- fspacer(fcastHolder)
fframe <- list()
for(i in 1:n_windows){
  #name = paste0('df', i)
  lname = paste0('w', i)
  #assign(name, data.frame(date = ttime, pred = f_spaced[i, ]))
  tmp <- data.frame(date = ttime, pred = f_spaced[i, ])
  fframe[[lname]] <- tmp
  }
f_df <- cbind(cat= rep(names(fframe), sapply(fframe, NROW)), do.call(rbind, fframe)) #arrange data with
o_df <- data.frame(cat = rep('original', length(tdata)), date = ttime, pred = tdata) #add actual data a
```

```r
f_df <- rbind(f_df, o_df)

ggplot(f_df, aes(date,pred, color= cat)) + geom_line() +
  ggtitle(paste("Total Influenza Test Cases: India", year_range[1:n_years][n_years], ' - ', year_range[
```

```
## Warning: Removed 9180 row(s) containing missing values (geom_path).
```



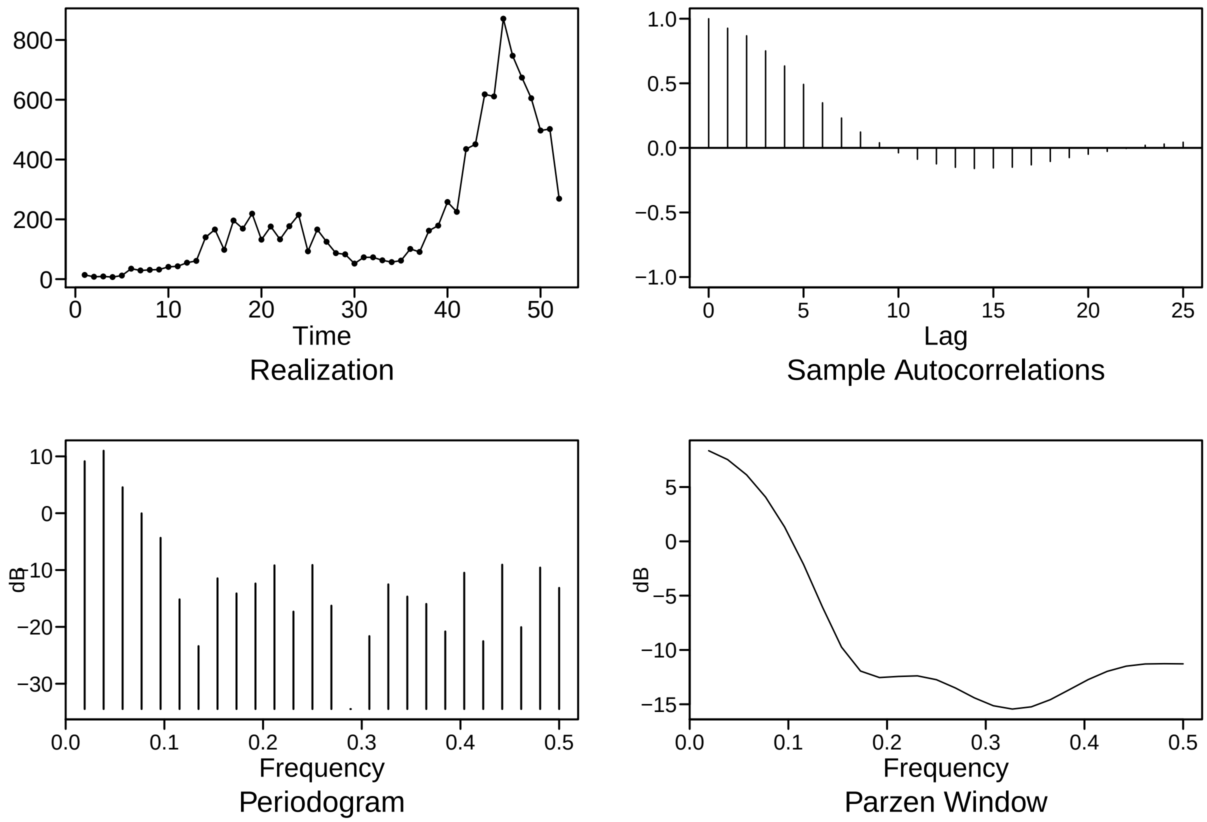

```r
#### end ####
```

#determine optimal # of years for forecast with ASE

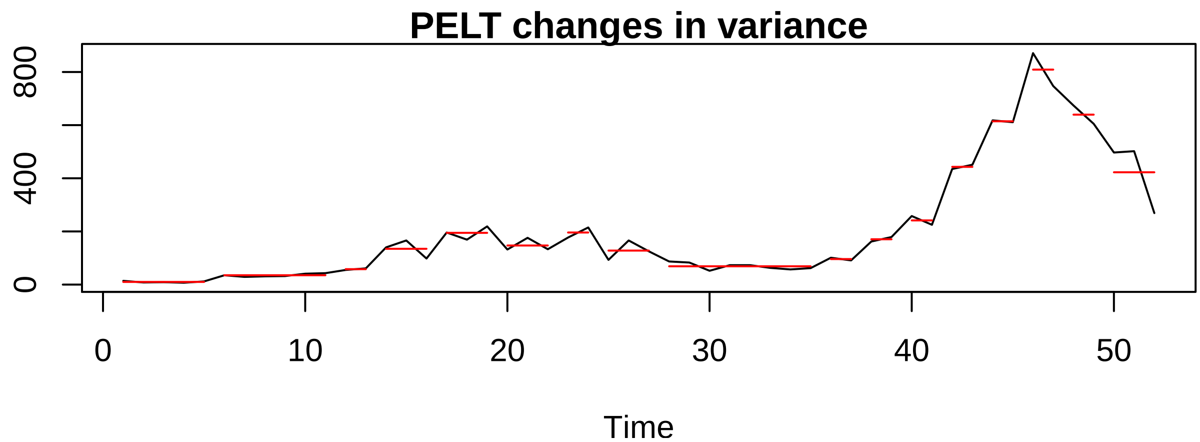
```r
#iterating on iterable year range
#dataset generation
S_Asia <- raw_data[which(raw_data$FLUREGION == 'Southern Asia'), ]
t_ind <- S_Asia[which(S_Asia['ï..Country'] == 'India'), ] #leave na, convert to zero
t_ind$ALL_INF[is.na(t_ind$ALL_INF)] <- 0
year_range = 2019:1995 #call with year_range[1], ect

meta_ep_ase = vector() #store the endpoint and rolling ase for each run of n years in the data
meta_roll_ase = vector()
meta_counter = 1

for(i in 1:length(year_range)){
  n_years = i #create dataset with 1 years data, 2 years, ect, ect
```

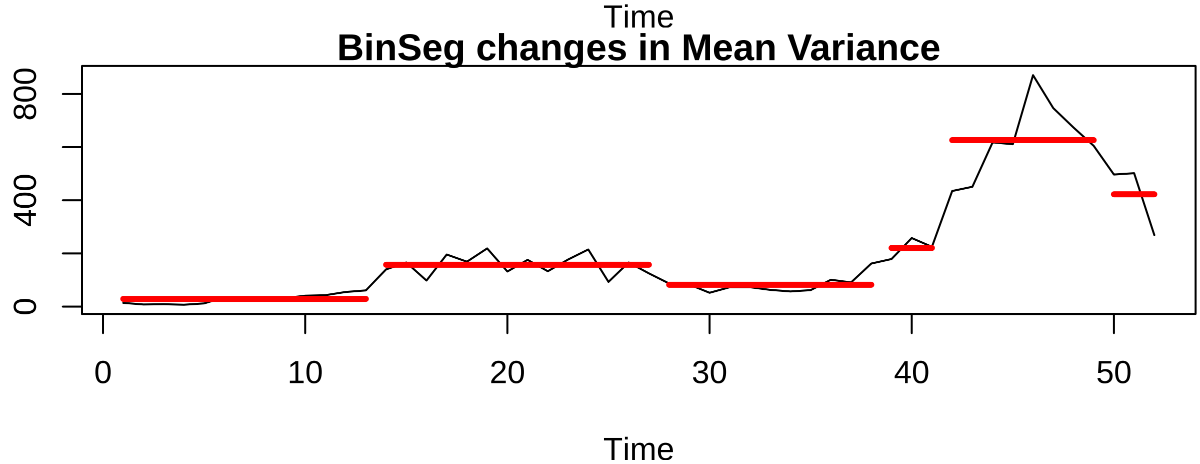
```
range_data = vector()
for(j in 1:n_years){
  range_data <- rbind(range_data, t_ind[which(t_ind$Year == year_range[j]), ])
  } #there we go, creates a df based on specified index years
tdata <- rev(range_data$ALL_INF) #set primary ts to all influenza cases #IMPORTANT to reverse the dat
ttime <- rev(range_data$SDATE) %>% as.Date("%m/%d/%Y") %>% sort()#because it's in descending year fro

#### model diagnostics ####
invisible(plotts.sample.wge(tdata)) #wrap objects in invisible() to hide the long console output and
#stationarity testin
adf.test(tdata)
#pacf(tdata)
#changepoint
par(mfrow = c(2,1))
v2 <- tdata #changepoint visualization
v2.pelt <- cpt.meanvar(v2, test.stat = 'Poisson', method = "PELT")
plot(v2.pelt, main = 'PELT changes in variance')
cpts.ts(v2.pelt)
v2.bs <- cpt.meanvar(v2, test.stat = 'Poisson', method = 'BinSeg')
plot(v2.bs, cpt.width = 3, main = 'BinSeg changes in Mean Variance')
cpts.ts(v2.bs)
#### end ####

#### model fitting ####
pspan = 0:5 #range of values to look for possible p and q coefficients for AR and MA
qspan = 0:5
difference = 0
aic_results <- aic5.wge(tdata, p = pspan, q = qspan) #cut bic results
#aic_results
m1 = est.arma.wge(tdata, p = aic_results[1, 1], q = aic_results[1, 2], factor = TRUE) #feed top AIC i
m1$phi
m1$theta
#### end ####

#### endpoint forcast ####
#simple single forecast and test
weeks_compare = 24 #how many weeks to reserve for testing : 6months
f1 <- tdata[1:(length(tdata)-weeks_compare)] %>% fore.aruma.wge(phi = m1$phi, theta = m1$theta, d = 0
ase <- mean((tdata[((length(tdata)-weeks_compare)+1):(length(tdata))] - f1$f)^2)
paste('Average Squared Error: ', ase)
#additional metrics
a_metrics <- accuracy(f1$f, tdata[((length(tdata)-weeks_compare)+1):(length(tdata))] )

#change x and y line names to match whatever dataframe you've stored the time series in
timeFrame <- data.frame(date = ttime, inf_cases = tdata)
g4.2 <- ggplot(timeFrame)+
  geom_line(aes(x = date, y = tdata, color = 'black'), size = 0.5)+
  geom_line(aes(x = date, y = c(rep(NA, (length(tdata)-weeks_compare)), f1$f), color = 'red'), size=
  scale_color_discrete(name = "total_patients", labels = c('actual', 'predicted')) +
  ggtitle(paste('ARIMA (',aic_results[1, 1],',',aic_results[1, 2],',',difference,')', 'Forecast of ',
g4.2
#### end ####
```

```
#### rolling window ase and viz ####
#rolling window ase: apply model predictions to small segments of the given data and aggregate metric
trainingSize = 24
horizon = 4 #3months
step_size = 4
n_windows = round((length(tdata)-(trainingSize + horizon))/step_size) #number of whole windows of tra
ASEHolder = numeric()

fcastHolder = matrix(nrow = n_windows, ncol = horizon) #create matrix that has the number of rows and
fUpperHolder = matrix(nrow = n_windows, ncol = horizon) #same for upper and lower intervals
fLowerHolder = matrix(nrow = n_windows, ncol = horizon)
phis = m1$phi
thetas = m1$theta
s = 0
d = 0
for( i in 1:n_windows) #how many "windows" can fit into the total length of time, rounding down
  {
  t_start = 1+(step_size*(i-1)) #starting point for each window's training data
  t_end = trainingSize + (step_size*(i-1)) #endpoint
  forecasts = fore.aruma.wge(tdata[t_start:t_end],phi = phis, theta = thetas, s = s, d = d, n.ahead =
  ASE = mean((tdata[(t_end + 1):(t_end+horizon)] - forecasts$f)^2)
  ASEHolder[i] = ASE
  fcastHolder[i, ] <- forecasts$f
  fUpperHolder[i, ] <- forecasts$ul
  fLowerHolder[i, ] <- forecasts$ll
  }
summary(ASEHolder)
WindowedASE = mean(ASEHolder)
hist(ASEHolder, main = paste("Mean ASE across [", n_windows, '] moving windows: ', WindowedASE))
#### end ####

#### create categorical dataframe to store windowed predictions and visualize with ggplot ####
f_spaced <- fspacer(fcastHolder)
fframe <- list()
for(i in 1:n_windows){
  lname = paste0('w', i)
  tmp <- data.frame(date = ttime, pred = f_spaced[i, ])
  fframe[[lname]] <- tmp
  }
f_df <- cbind(cat= rep(names(fframe), sapply(fframe, NROW)), do.call(rbind, fframe)) #arrange data wi
o_df <- data.frame(cat = rep('original', length(tdata)), date = ttime, pred = tdata) #add actual data
f_df <- rbind(f_df, o_df)

g4.3 <- ggplot(f_df, aes(date,pred, color= cat)) + geom_line() +
ggtitle(paste("Total Influenza Test Cases: India", year_range[1:n_years][n_years], ' - ', year_range[
g4.3
#### end ####

meta_ep_ase[meta_counter] <- ase
meta_roll_ase[meta_counter] <- WindowedASE
meta_counter <- meta_counter+1

}
```

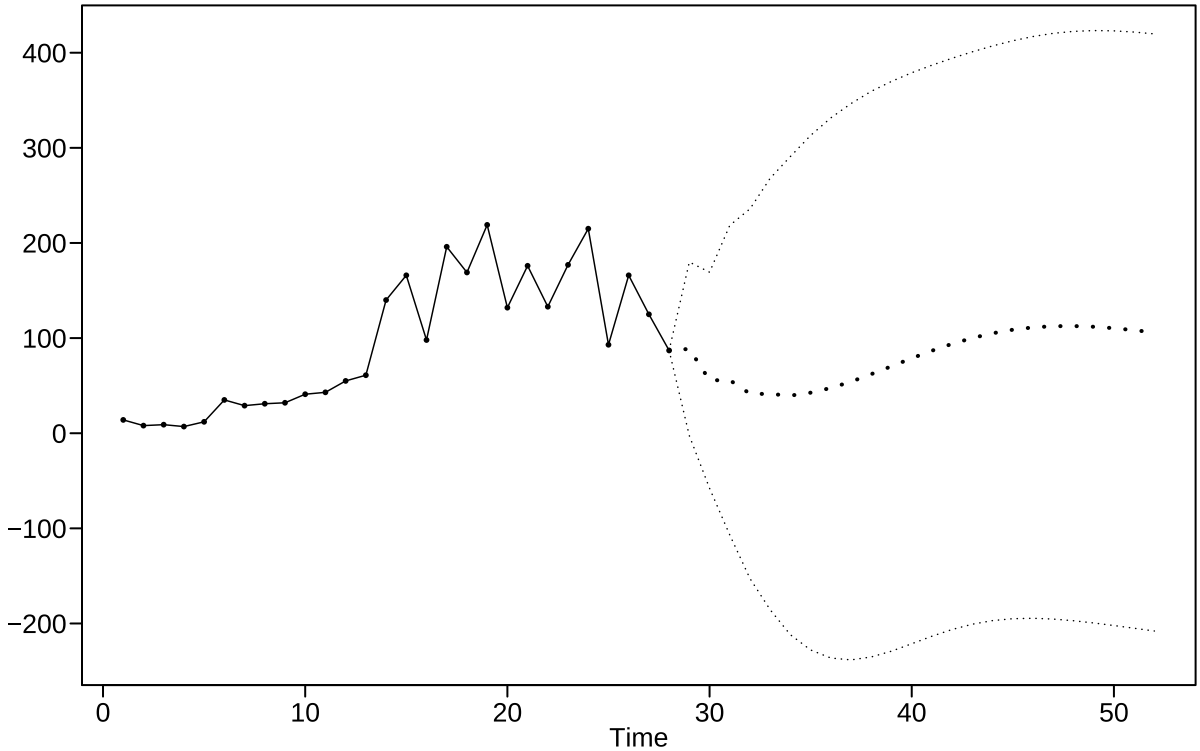


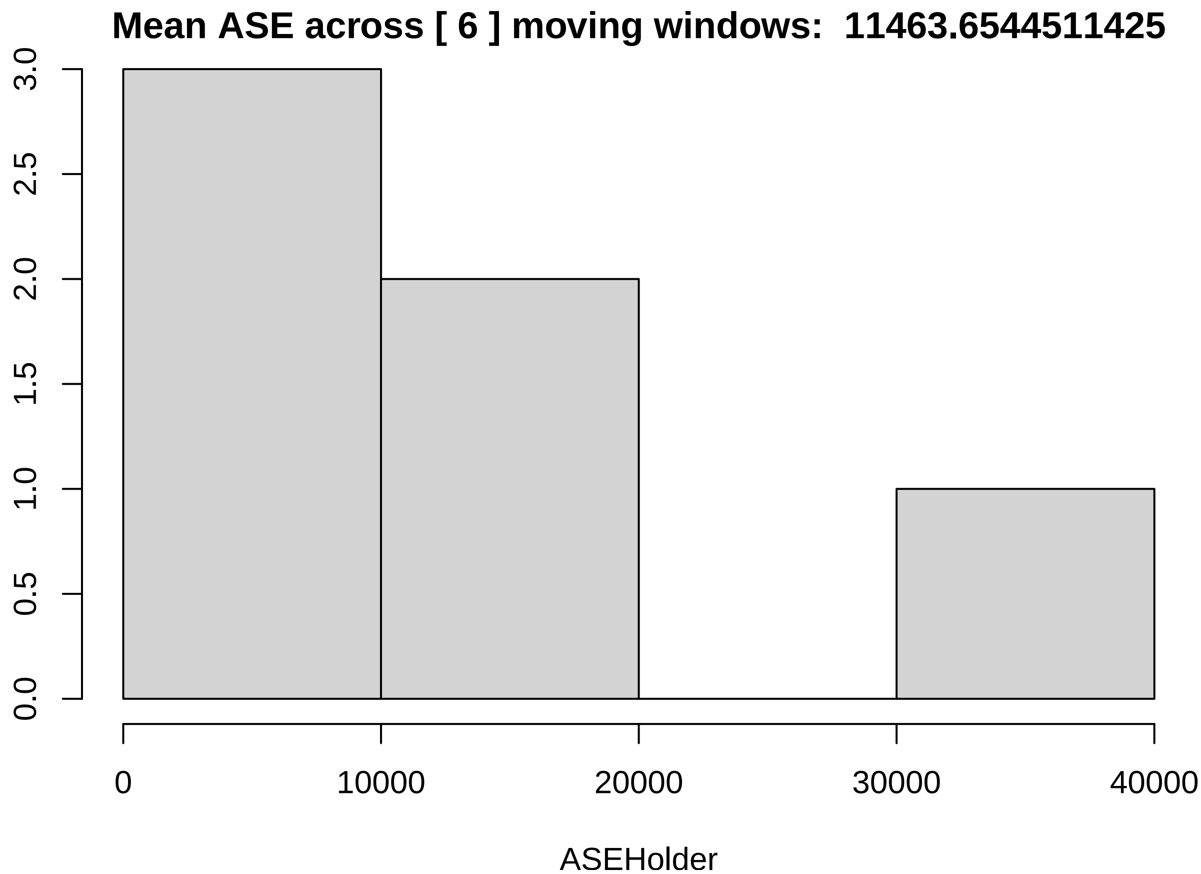

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

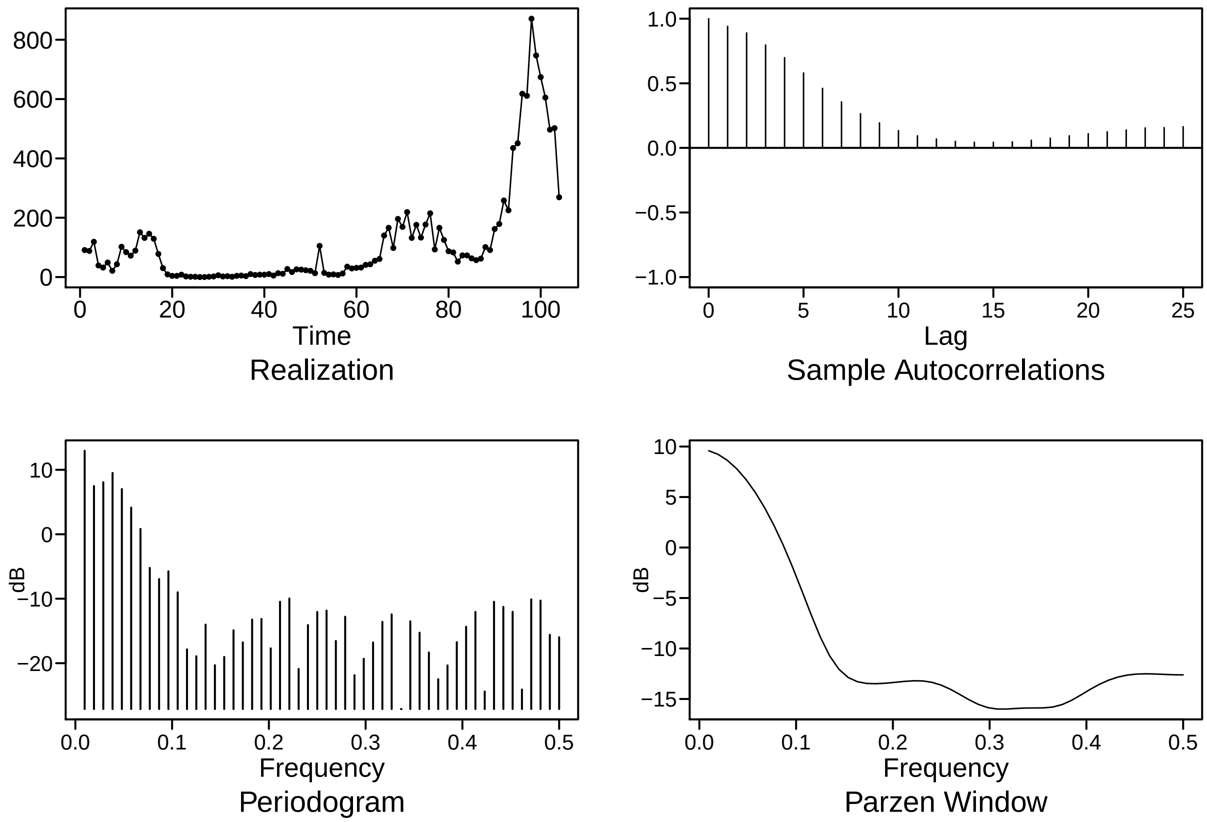




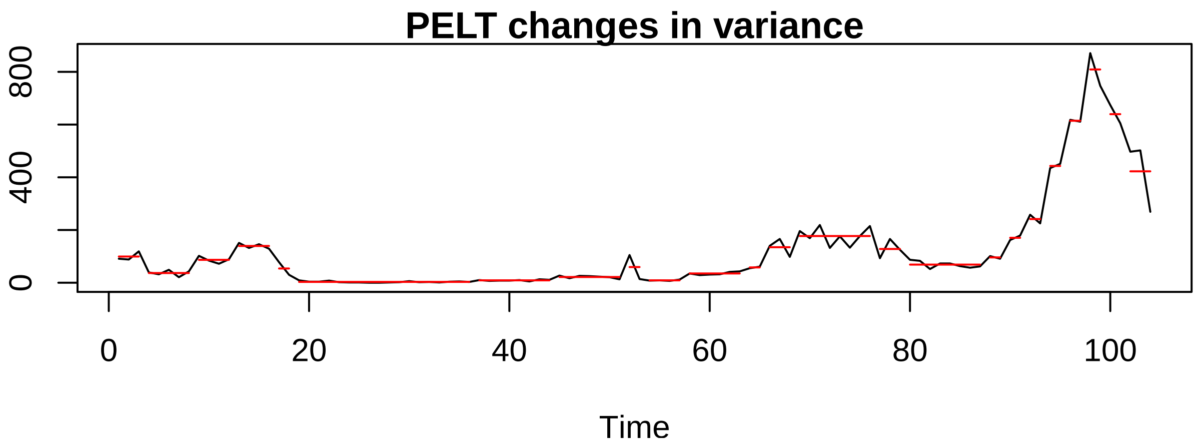

```
## ---------WORKING... PLEASE WAIT...
##
##
## Error in aic calculation at 3 1
## Error in aic calculation at 3 3
## Error in aic calculation at 4 2
## Error in aic calculation at 5 1
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.7403 0.7288 -0.3219 -0.2704
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.7958B+0.8477B^2    1.0592+-0.2403i      0.9207       0.0355
## 1+1.0555B+0.3190B^2   -1.6543+-0.6307i      0.5648       0.4420
##
##
```

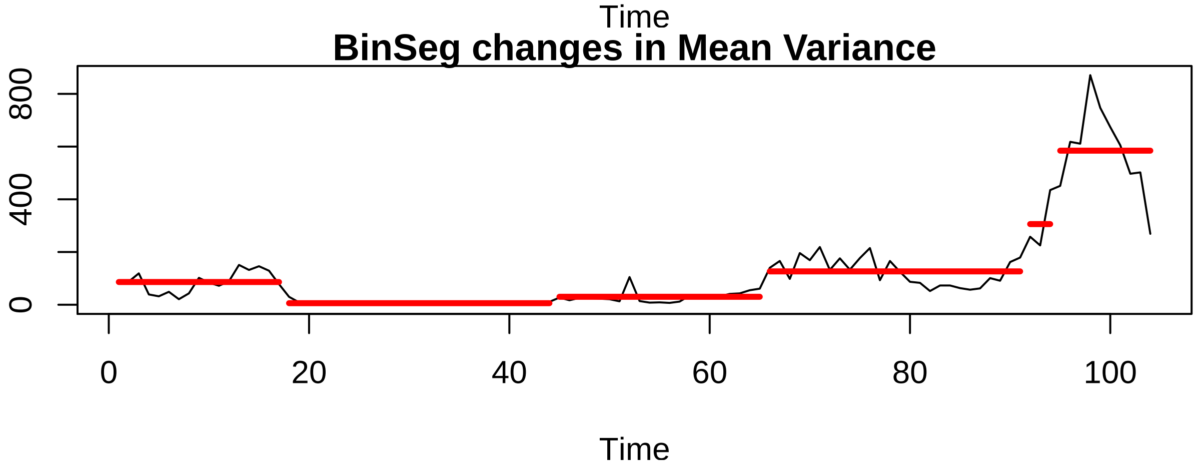

400
300
200
100
0
−100
−200
0
10
20
30
40
50
Time


Mean ASE across [6] moving windows: 11463.6544511425
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
10000
20000
30000
40000
ASEHolder

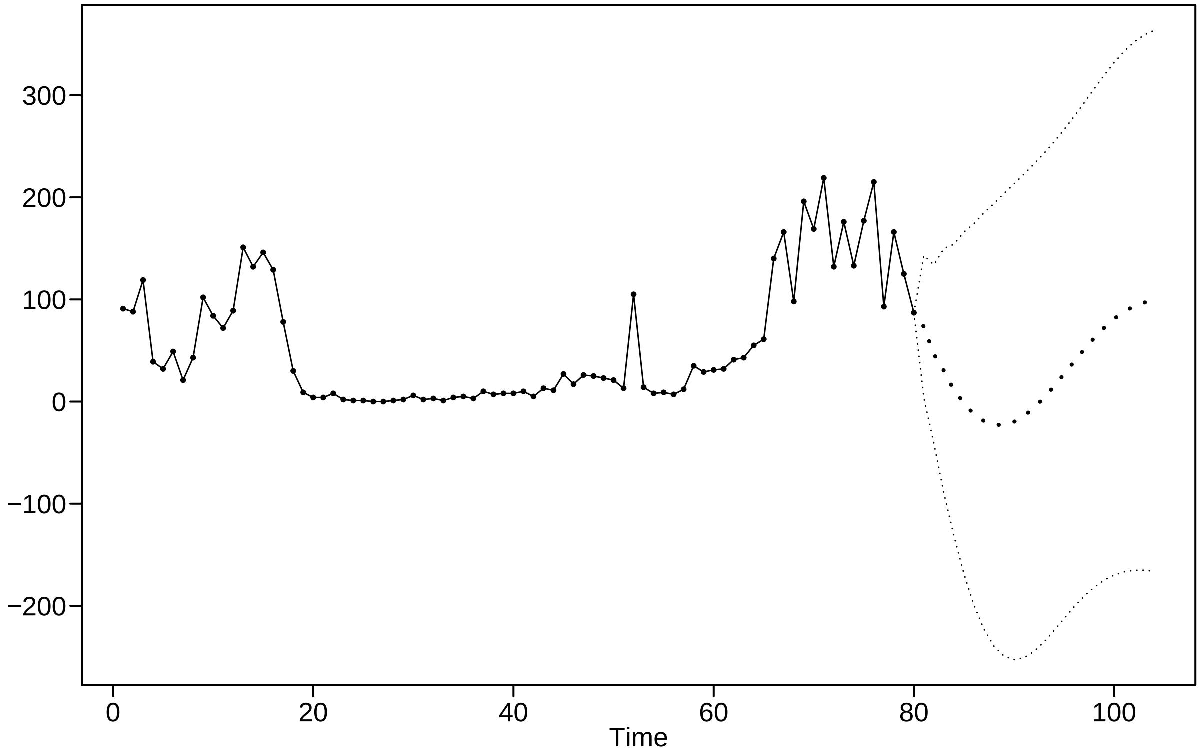



```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

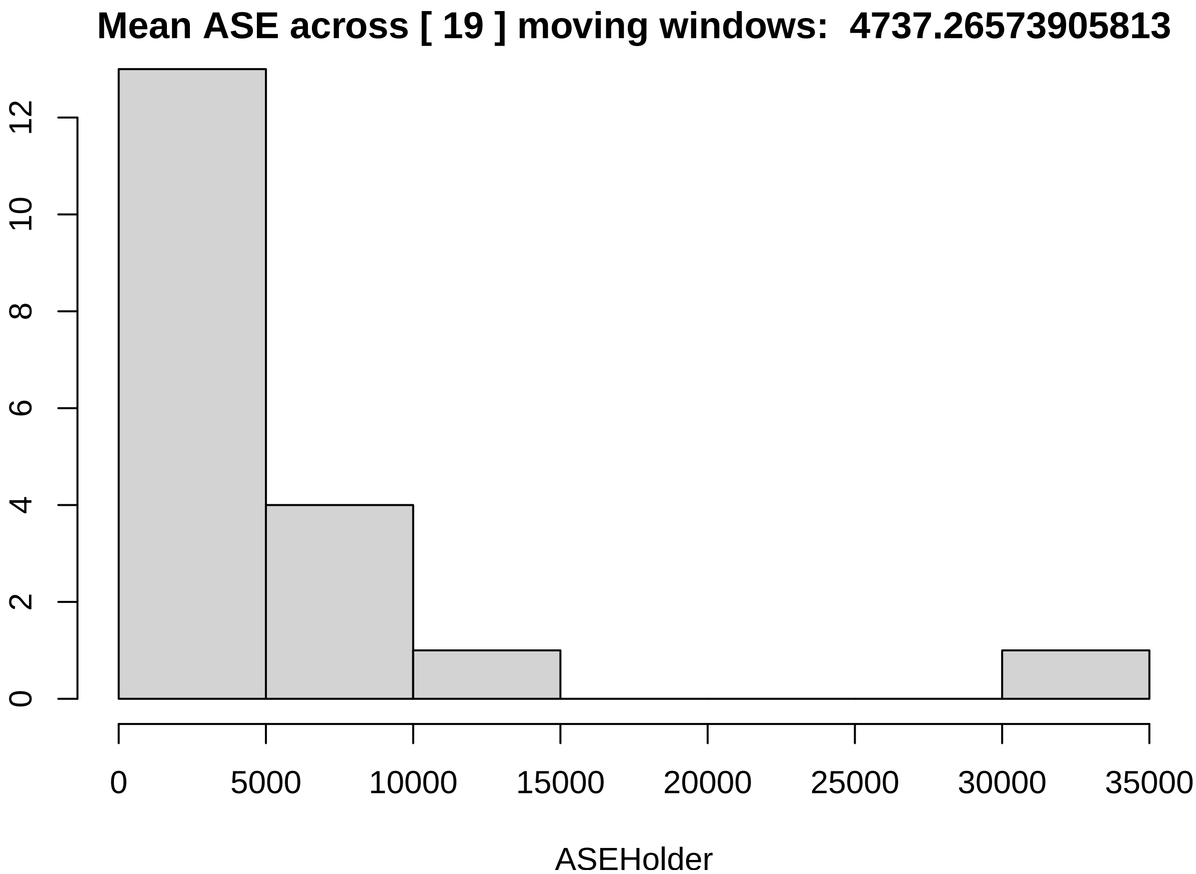




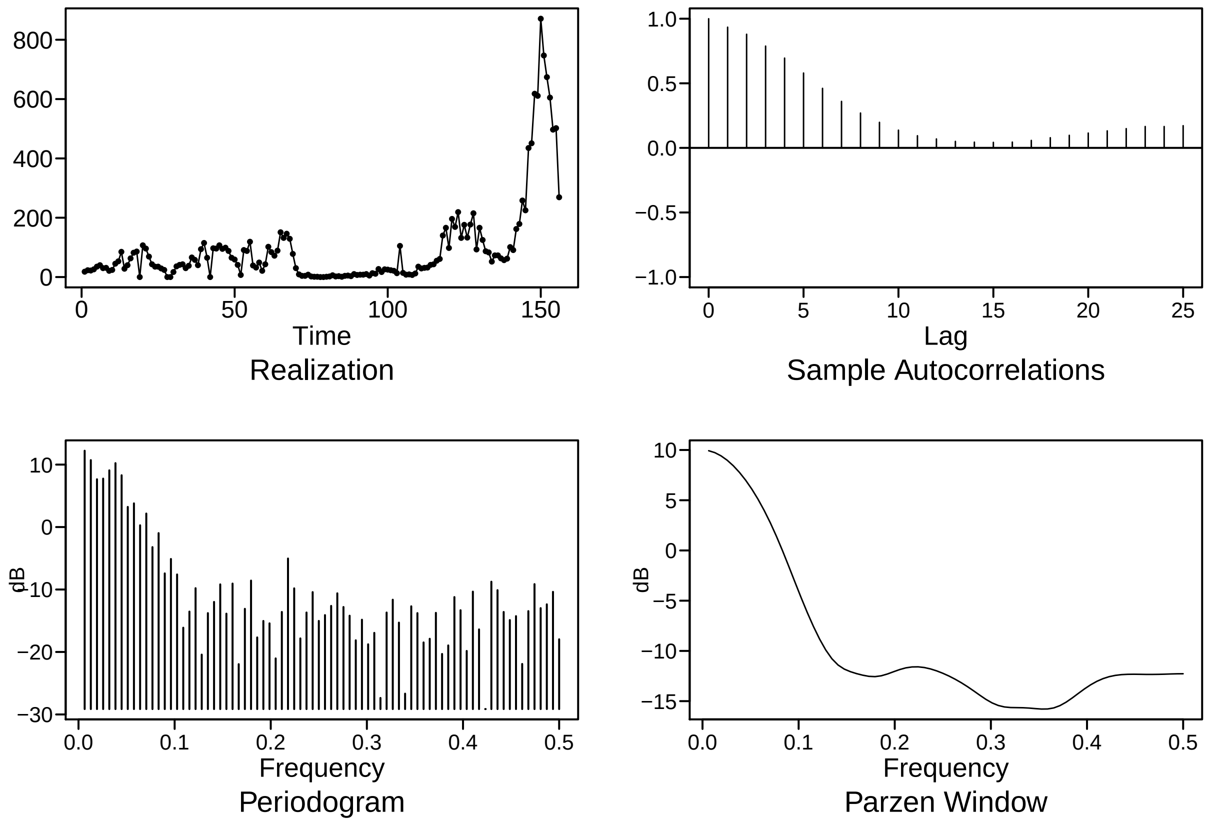

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 1.2806 0.2128 -0.5556
##
## Factor                     Roots                  Abs Recip    System Freq
## 1-1.8832B+0.9221B^2       1.0212+-0.2041i        0.9602       0.0314
## 1+0.6026B                -1.6595                 0.6026       0.5000
##
##
```



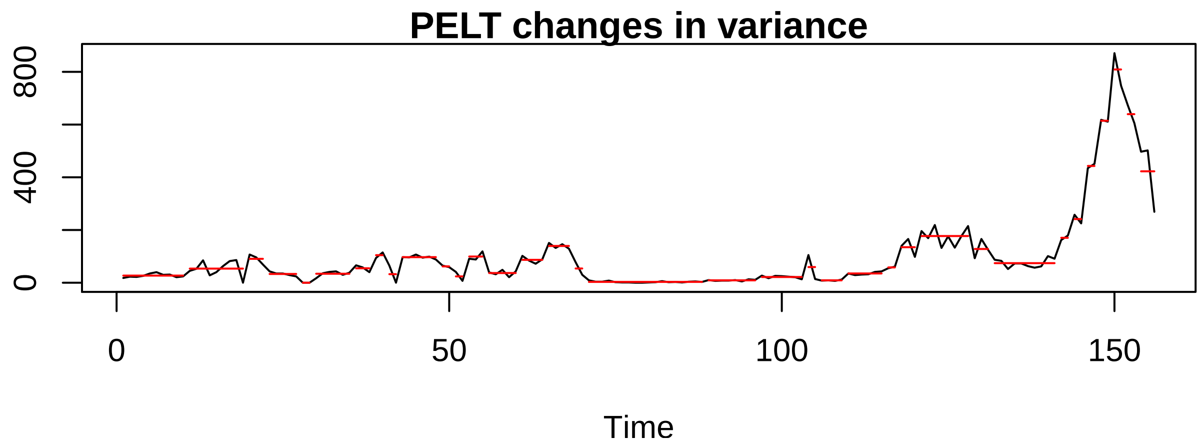
300
200
100
0
−100
−200
0
20
40
60
80
100
Time

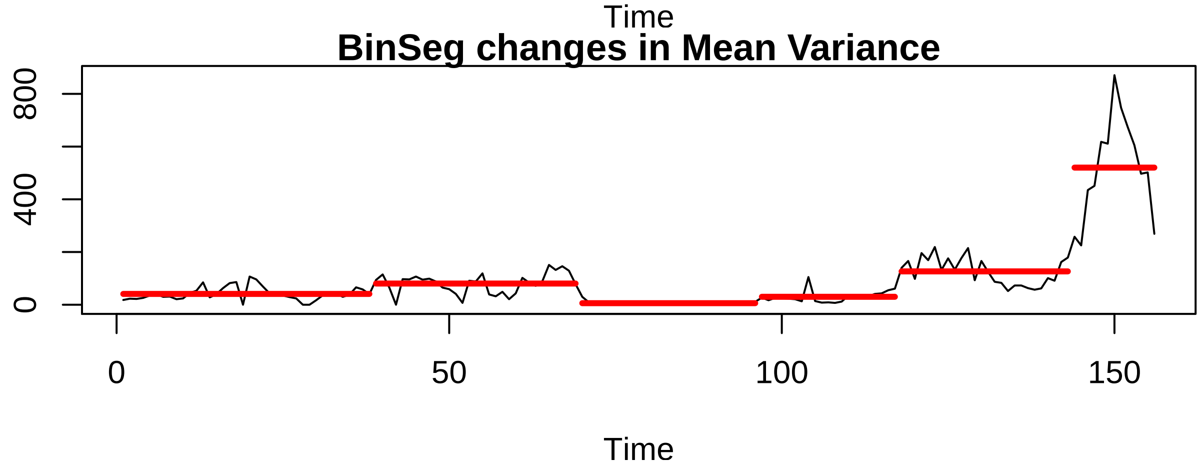



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

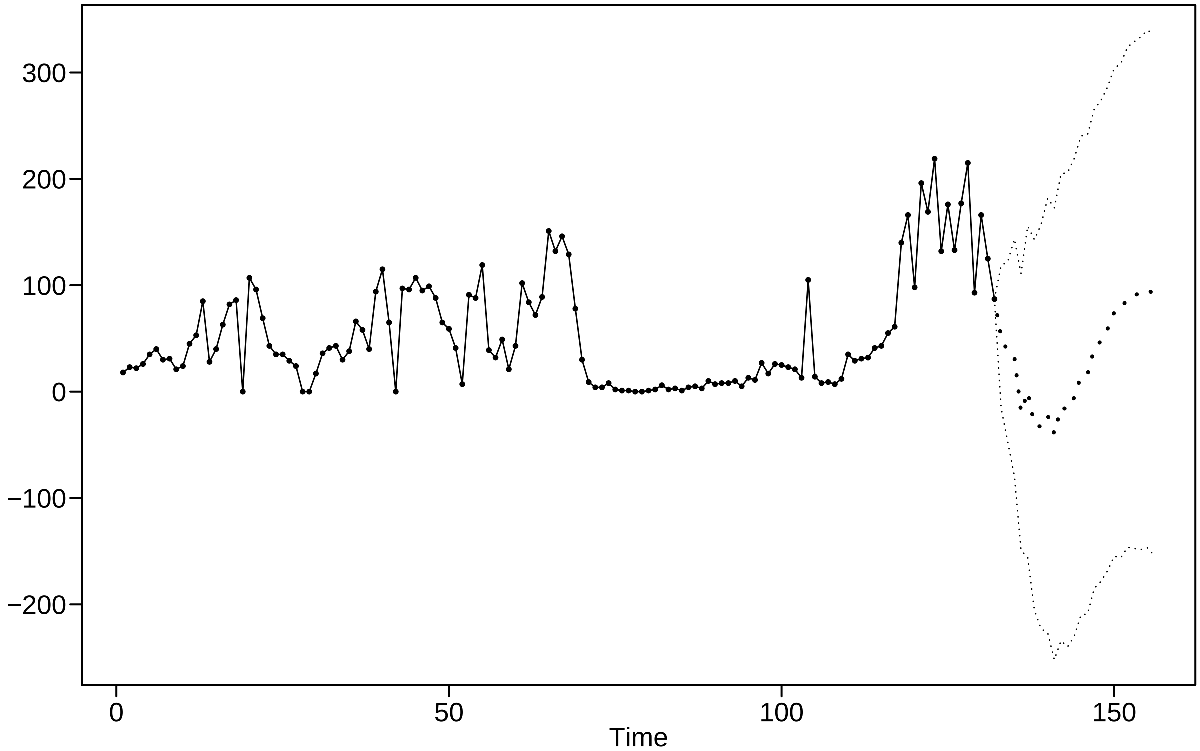


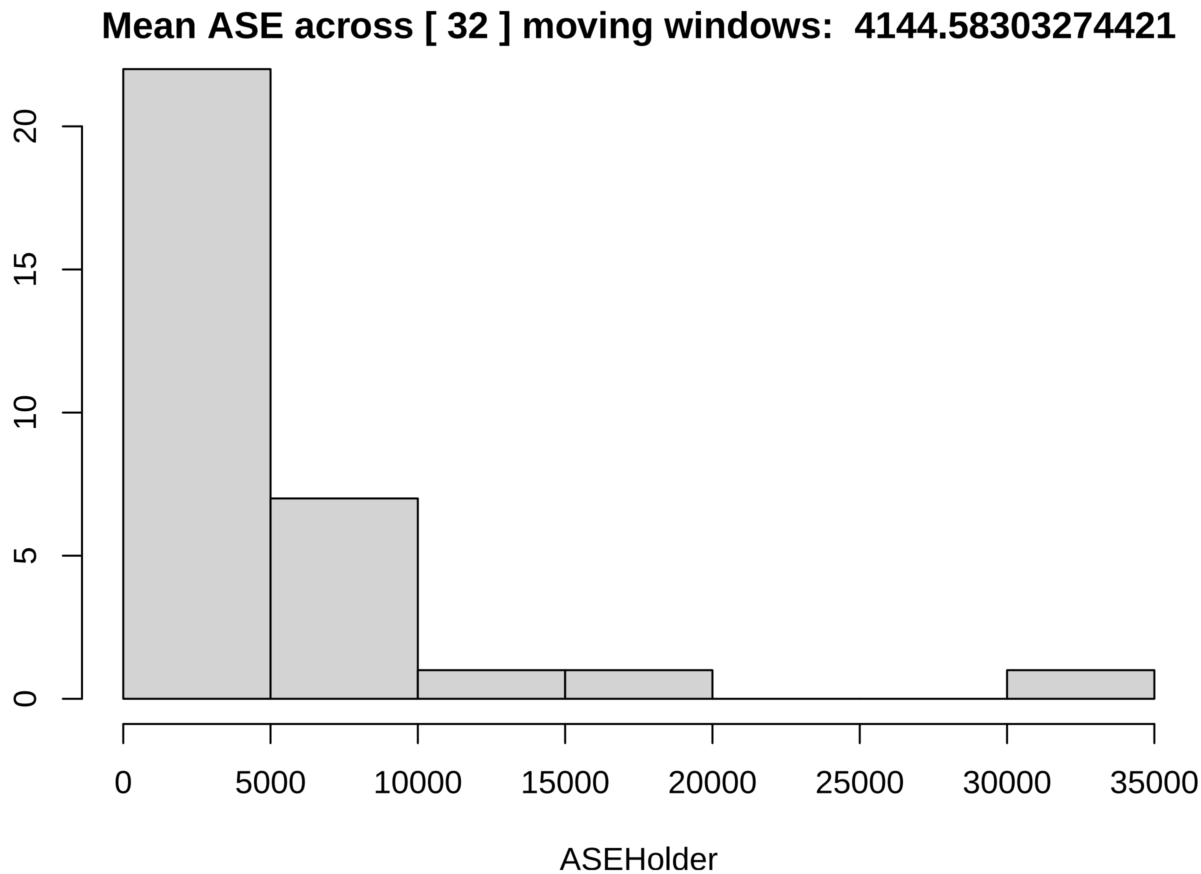

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



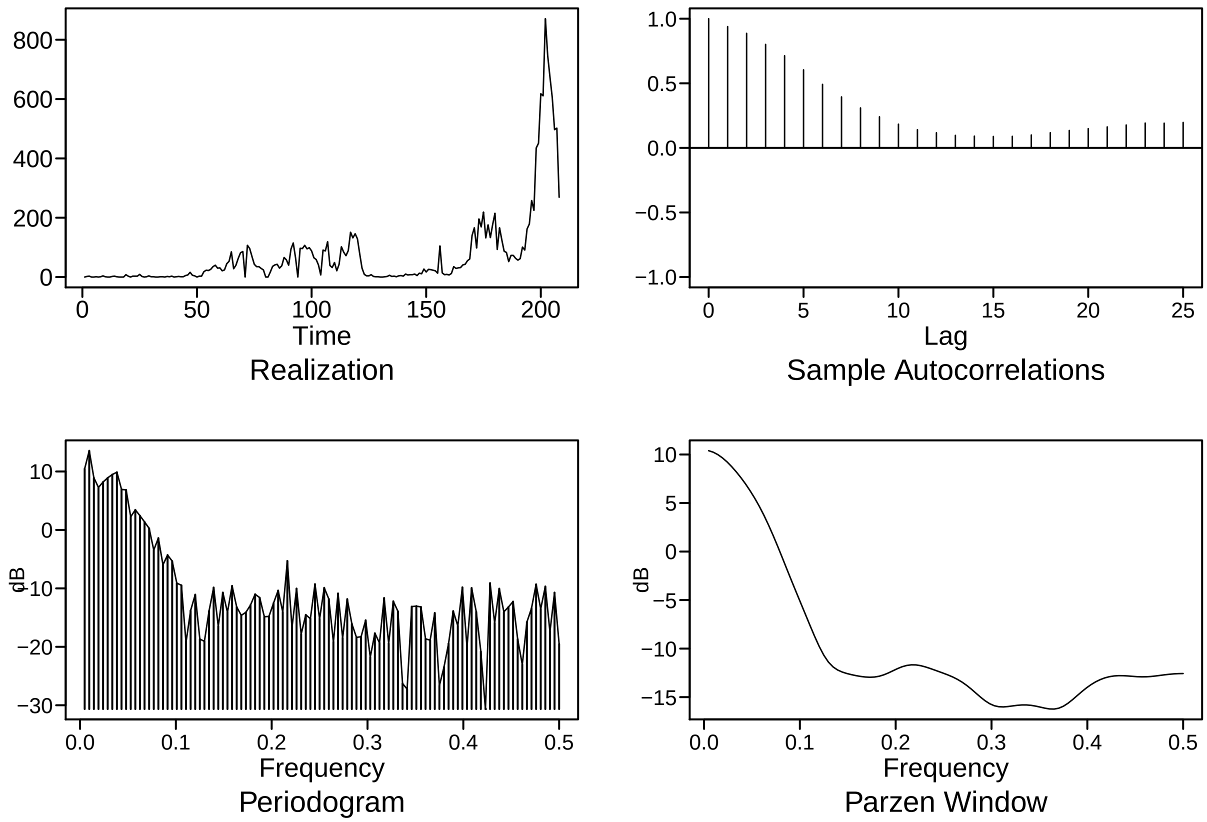



```
## ---------WORKING... PLEASE WAIT...
##
##
## Error in aic calculation at 4 3
## Error in aic calculation at 5 4
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.4329 0.9503 0.1997 -0.7247
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.8605B+0.9044B^2    1.0285+-0.2185i      0.9510       0.0333
## 1+1.4276B+0.8013B^2   -0.8908+-0.6741i      0.8952       0.3969
##
##
```

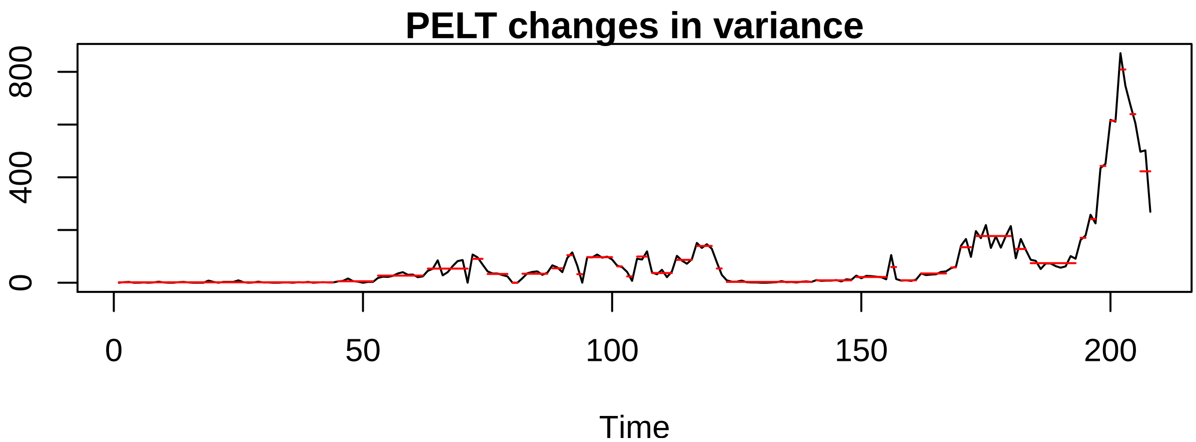

300
200
100
0
−100
−200
0
50
100
150
Time

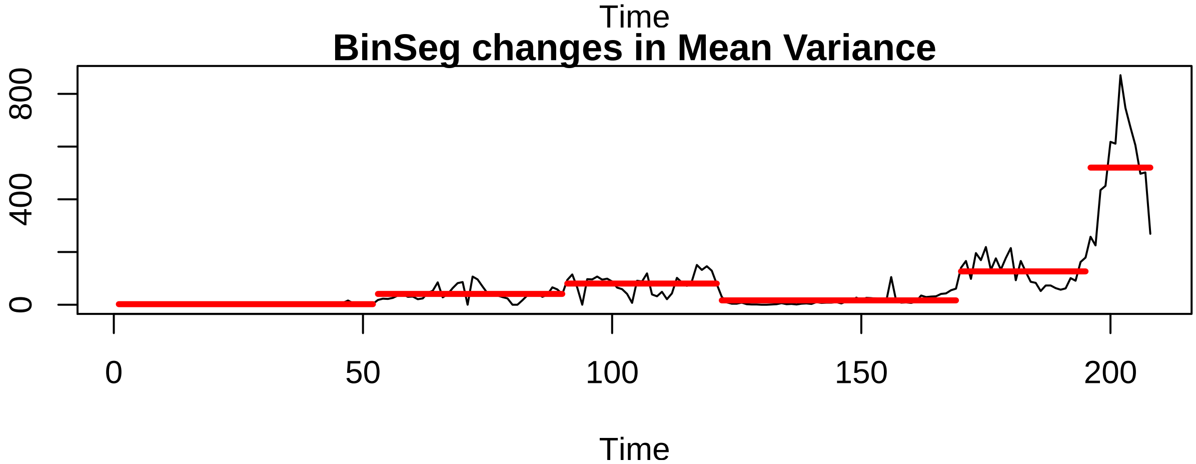



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

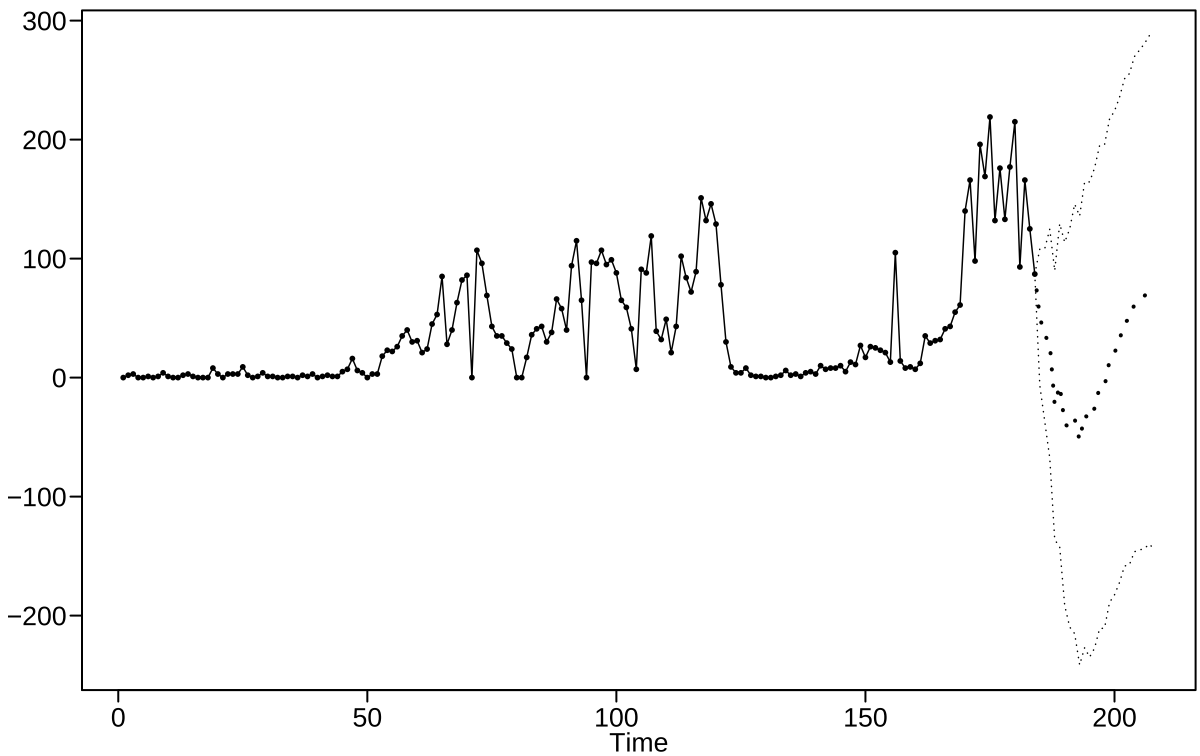


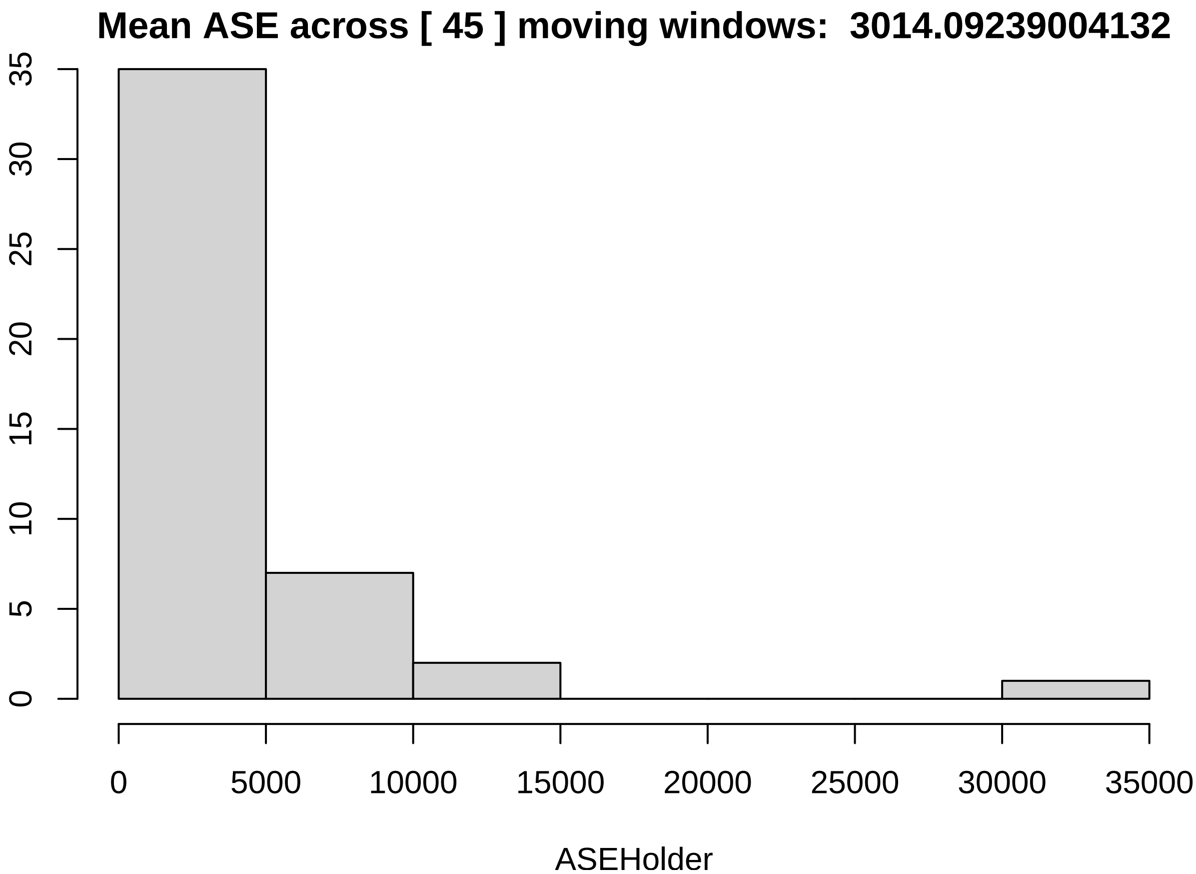

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

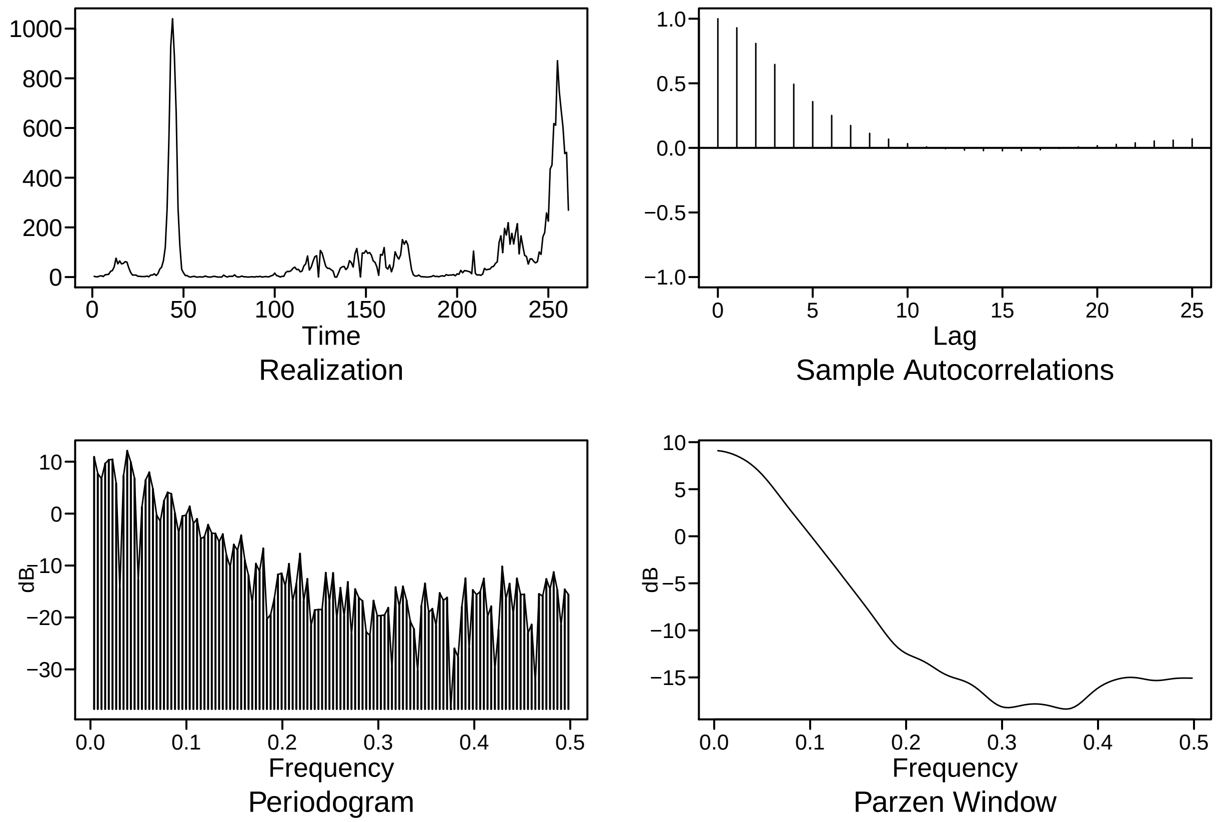




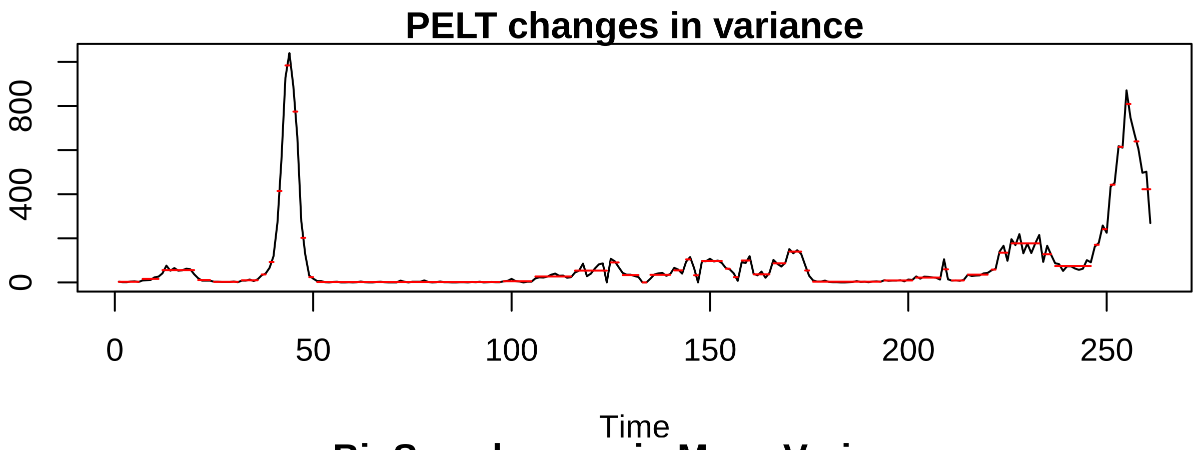

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.4298 0.9433 0.1995 -0.7044
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.8492B+0.8903B^2    1.0386+-0.2112i      0.9435       0.0319
## 1+1.4194B+0.7912B^2   -0.8970+-0.6777i      0.8895       0.3970
##
##
```

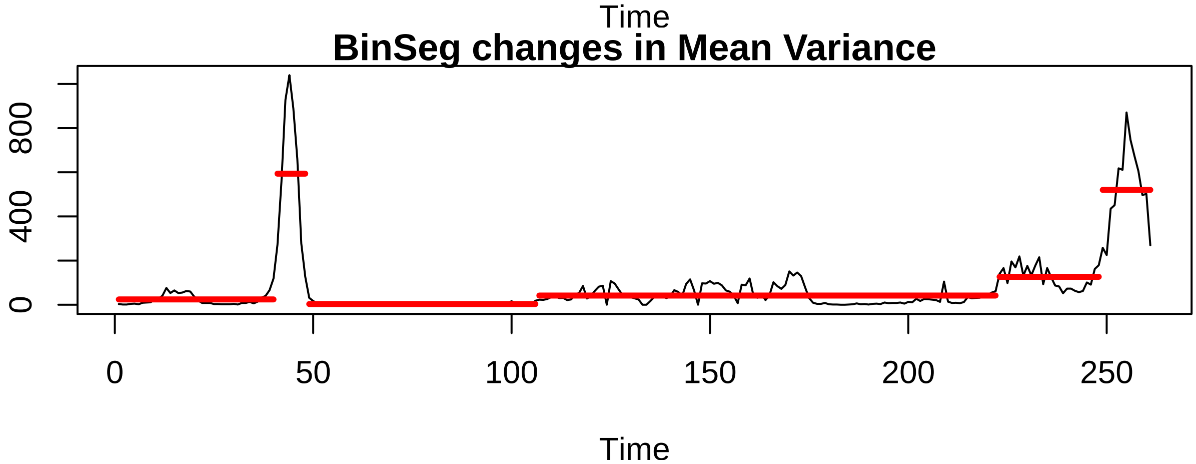

300
200
100
0
−100
−200
0
50
100
150
200
Time


Mean ASE across [45] moving windows: 3014.09239004132
35
30
25
20
15
10
5
0
0
5000
10000
15000
20000
25000
30000
35000
ASEHolder

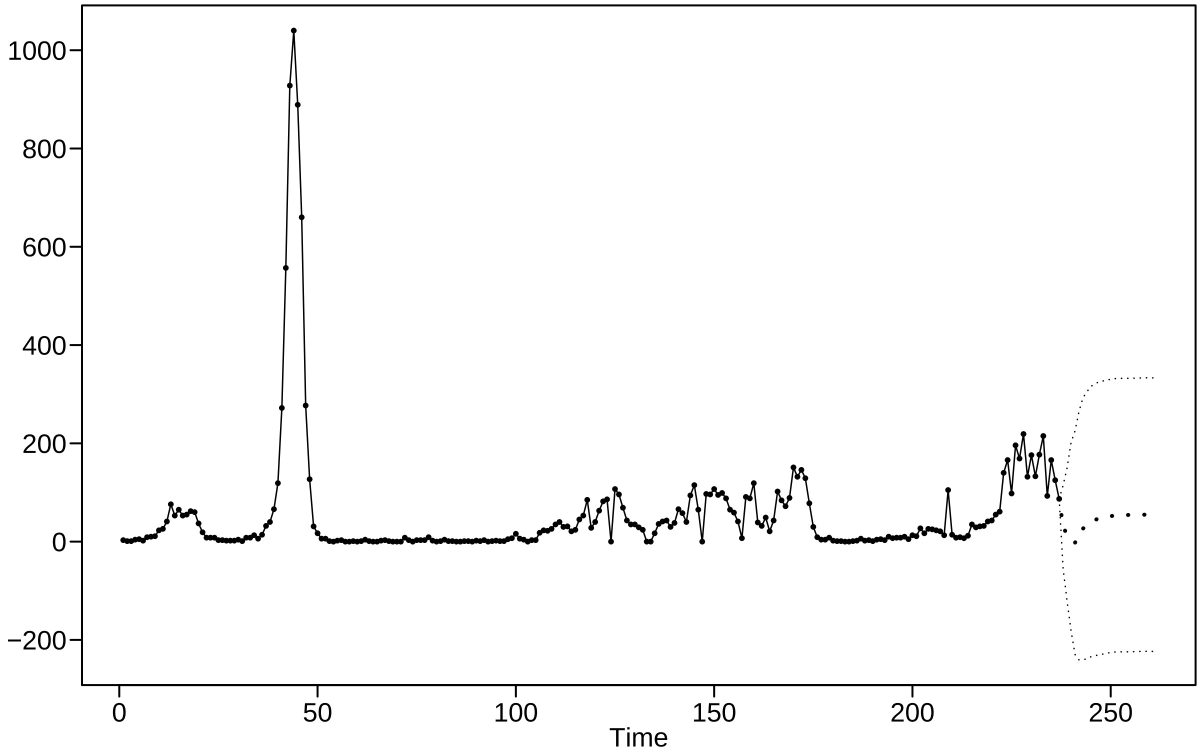


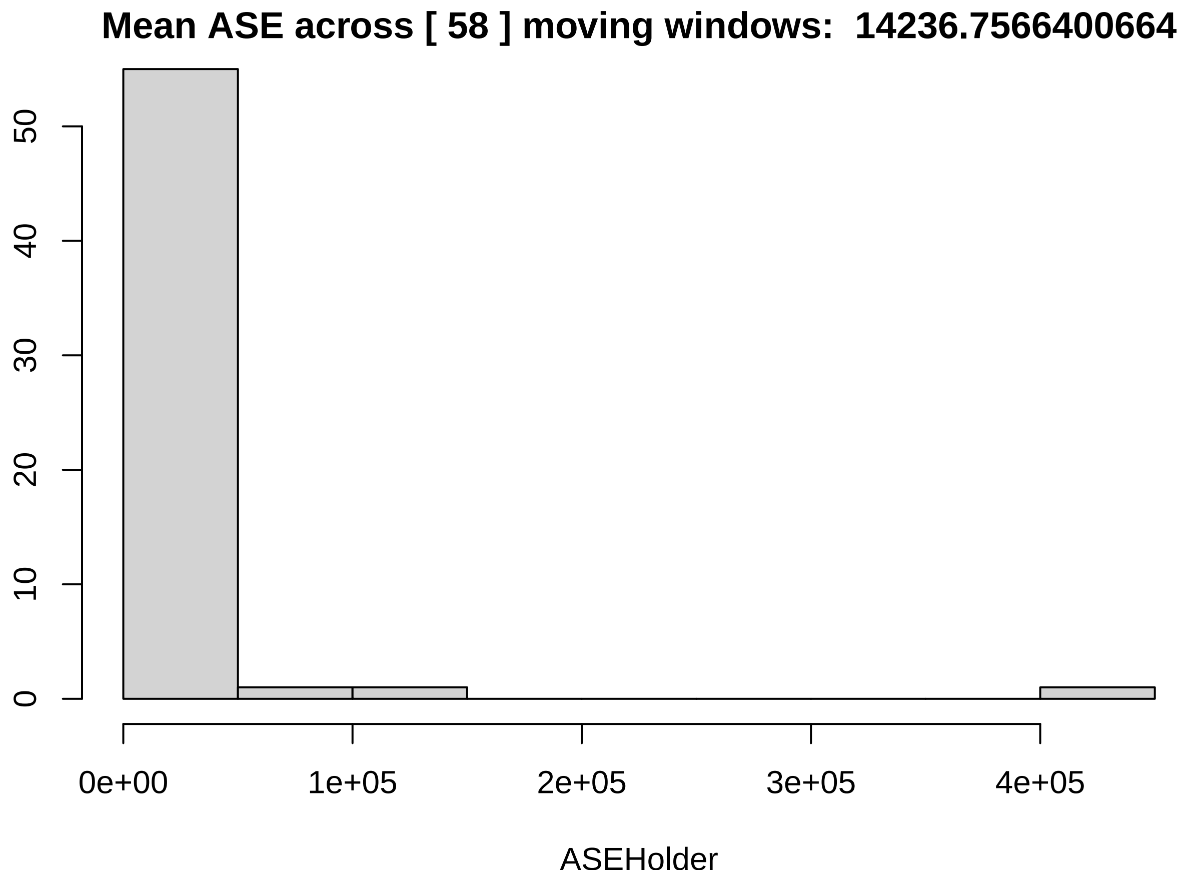

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



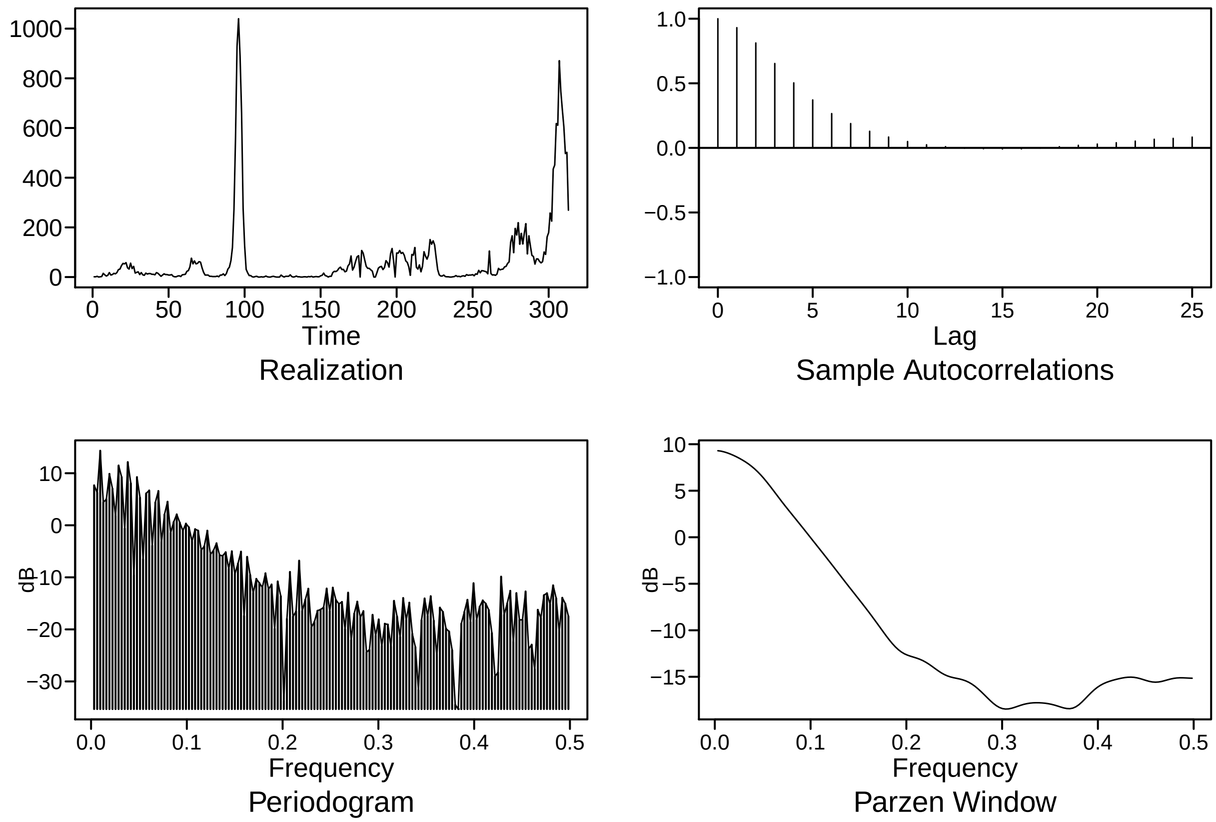



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## -0.2051 0.6517
##
## Factor                 Roots                Abs Recip    System Freq
## 1+0.9163B             -1.0913               0.9163       0.5000
## 1-0.7112B              1.4061               0.7112       0.0000
##
##
```

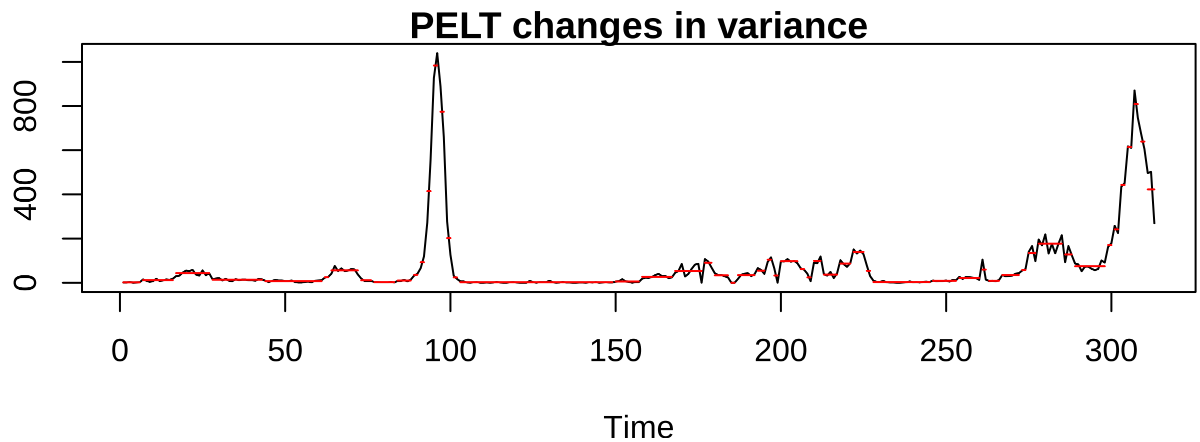

1000
800
600
400
200
0
−200
0
50
100
150
200
250
Time

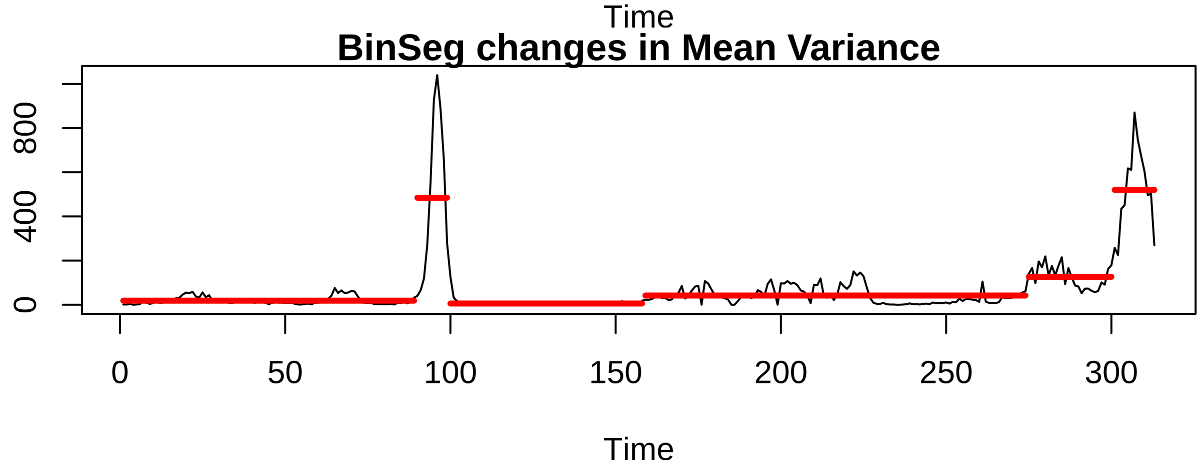



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

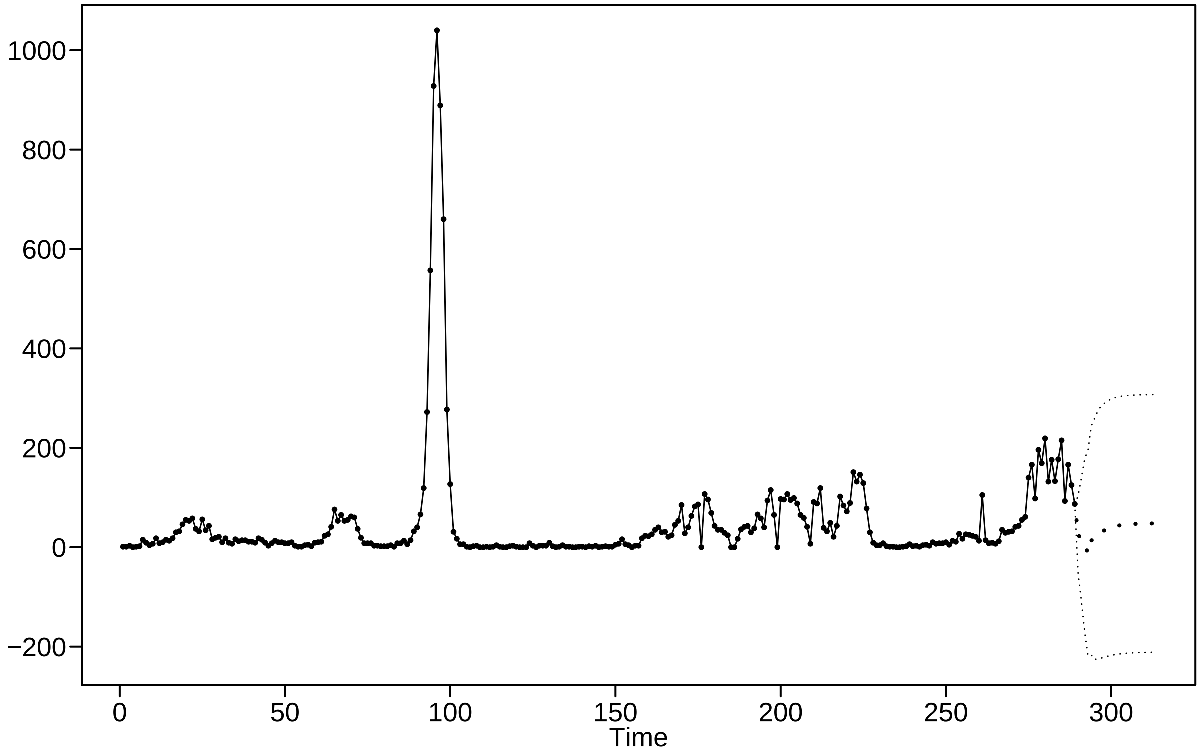


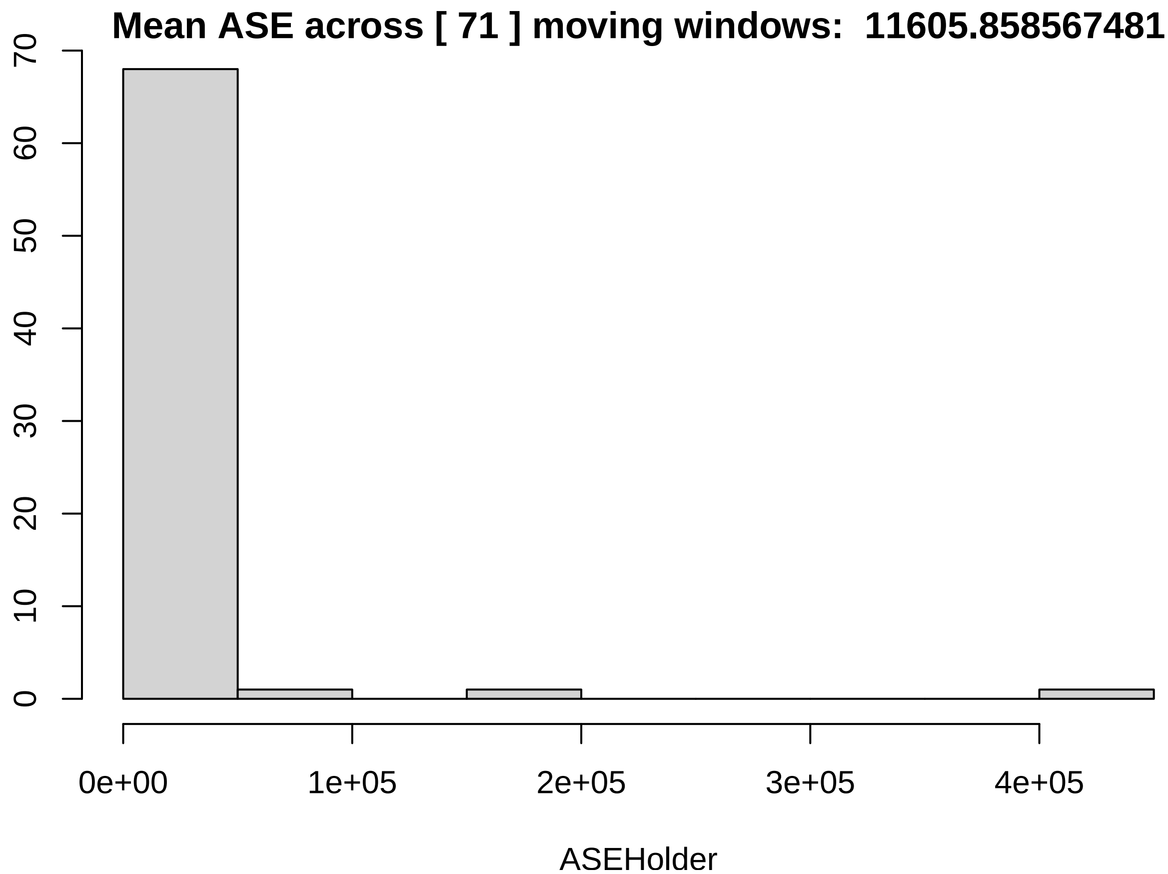

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



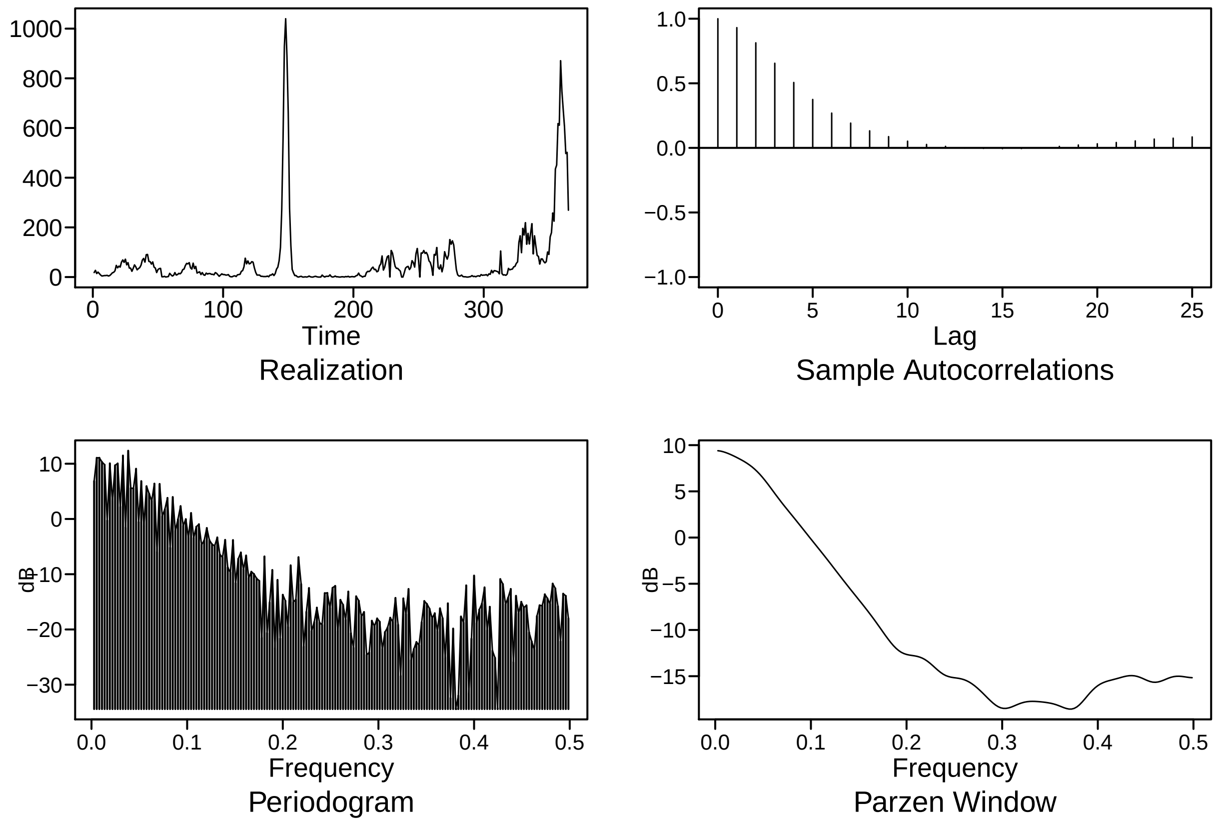



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## -0.5416 0.6012 0.3166
##
## Factor                 Roots                Abs Recip    System Freq
## 1+0.7978B             -1.2534               0.7978       0.5000
## 1-0.7709B              1.2971               0.7709       0.0000
## 1+0.5147B             -1.9427               0.5147       0.5000
##
##
```

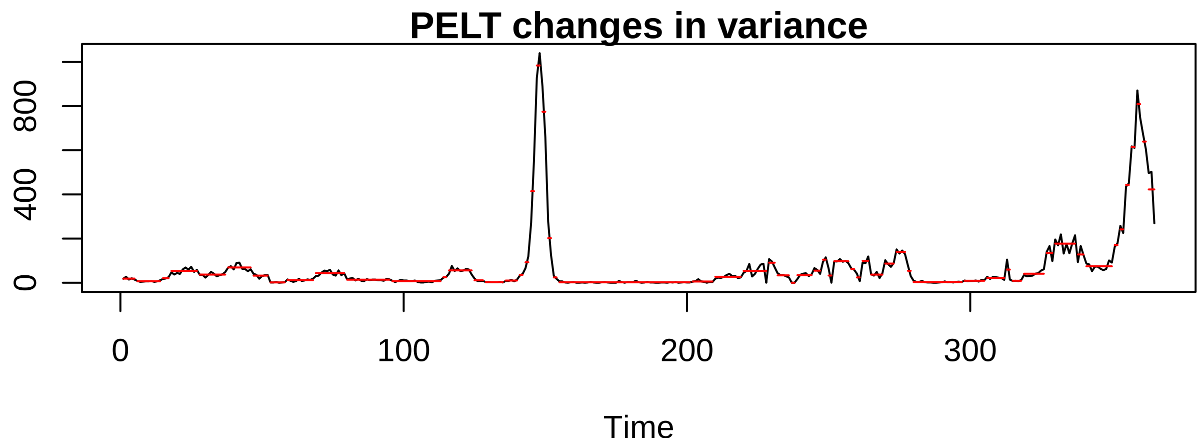

1000
800
600
400
200
0
−200
0
50
100
150
200
250
300
Time

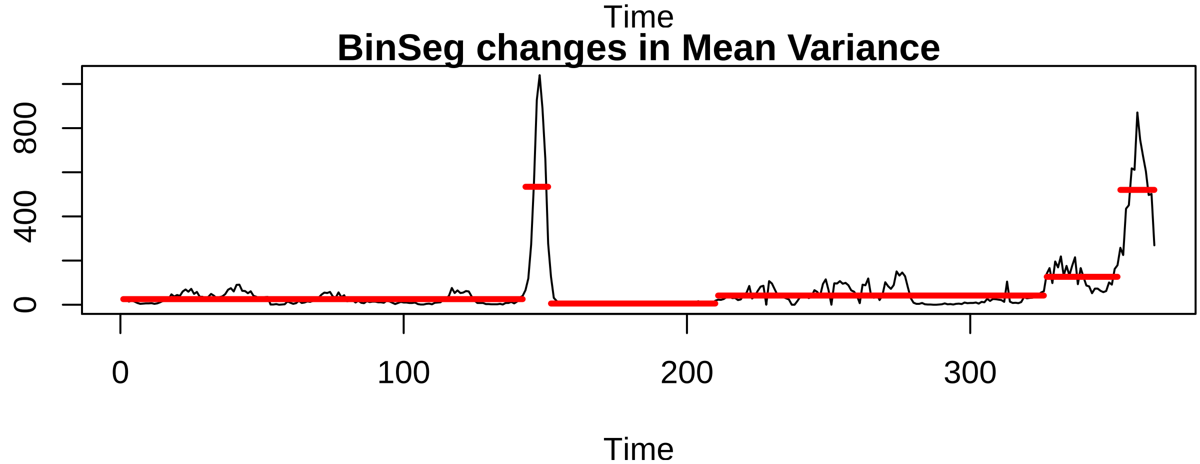



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

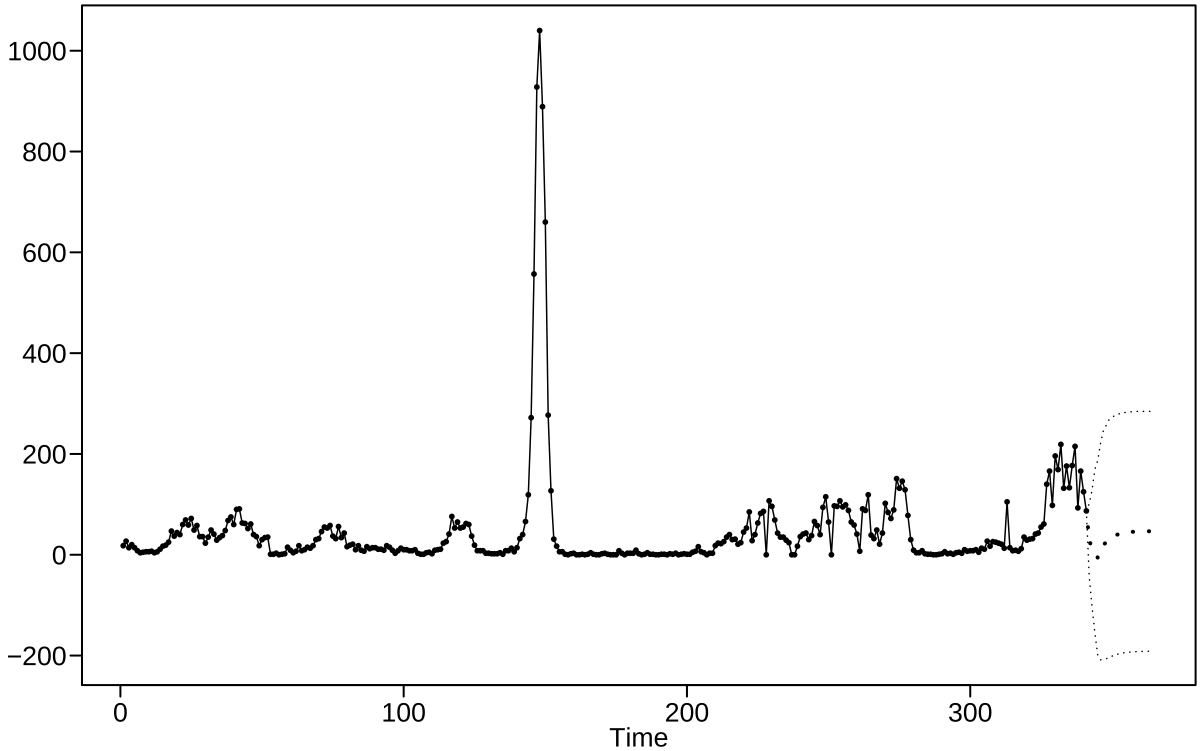



```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

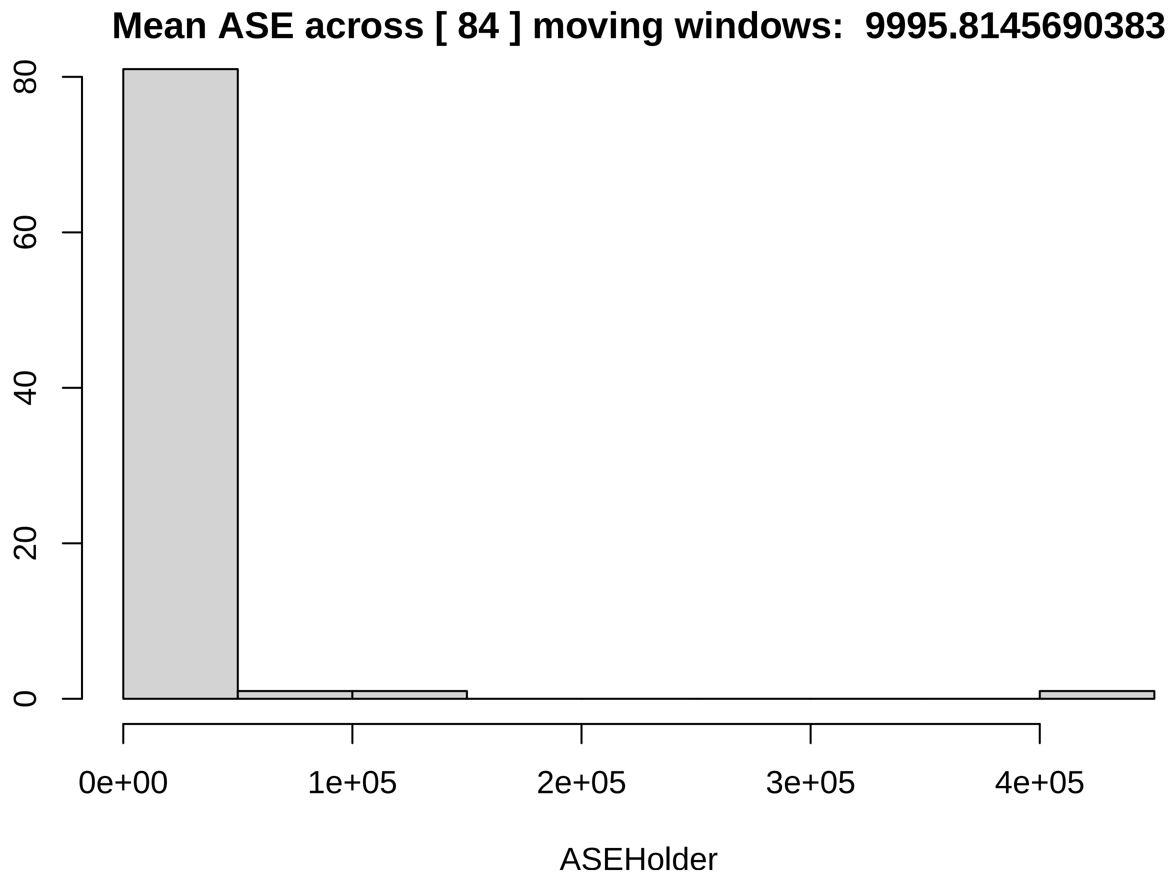




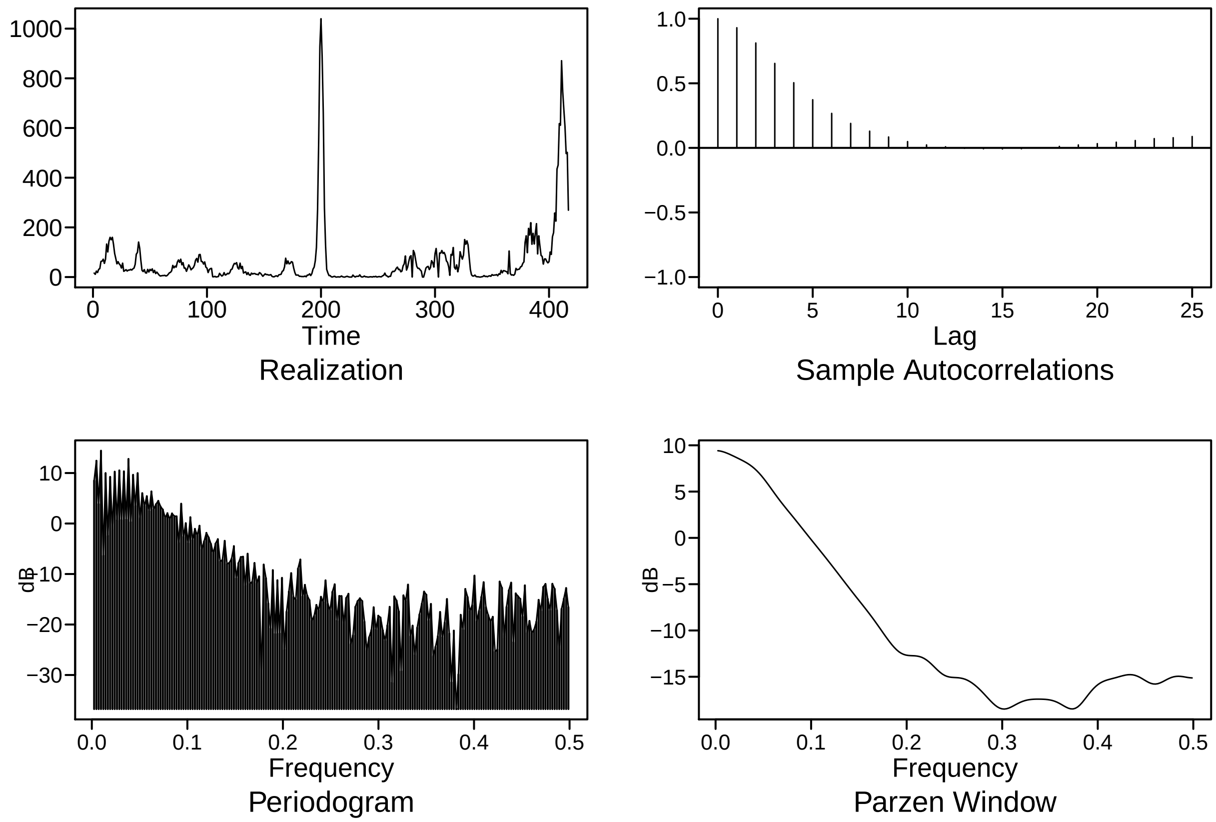

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## -0.1949 0.6654
##
## Factor                 Roots                Abs Recip    System Freq
## 1+0.9190B             -1.0881               0.9190       0.5000
## 1-0.7241B              1.3811               0.7241       0.0000
##
##
```



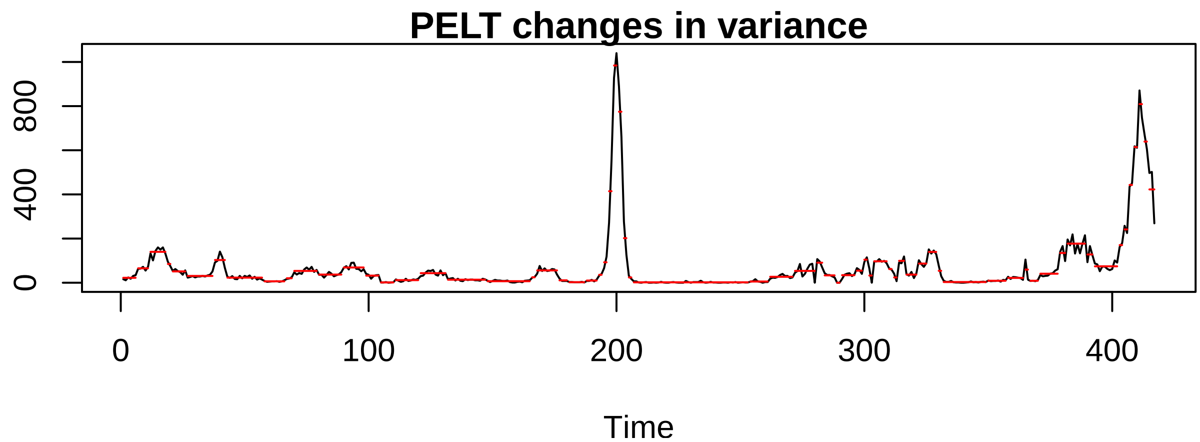
1000
800
600
400
200
0
−200
0
100
200
300
Time

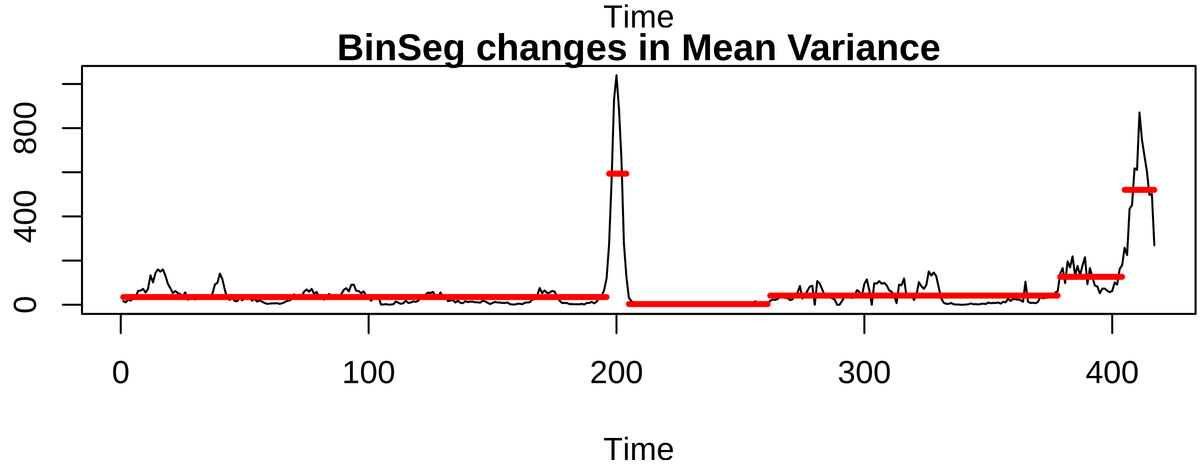



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

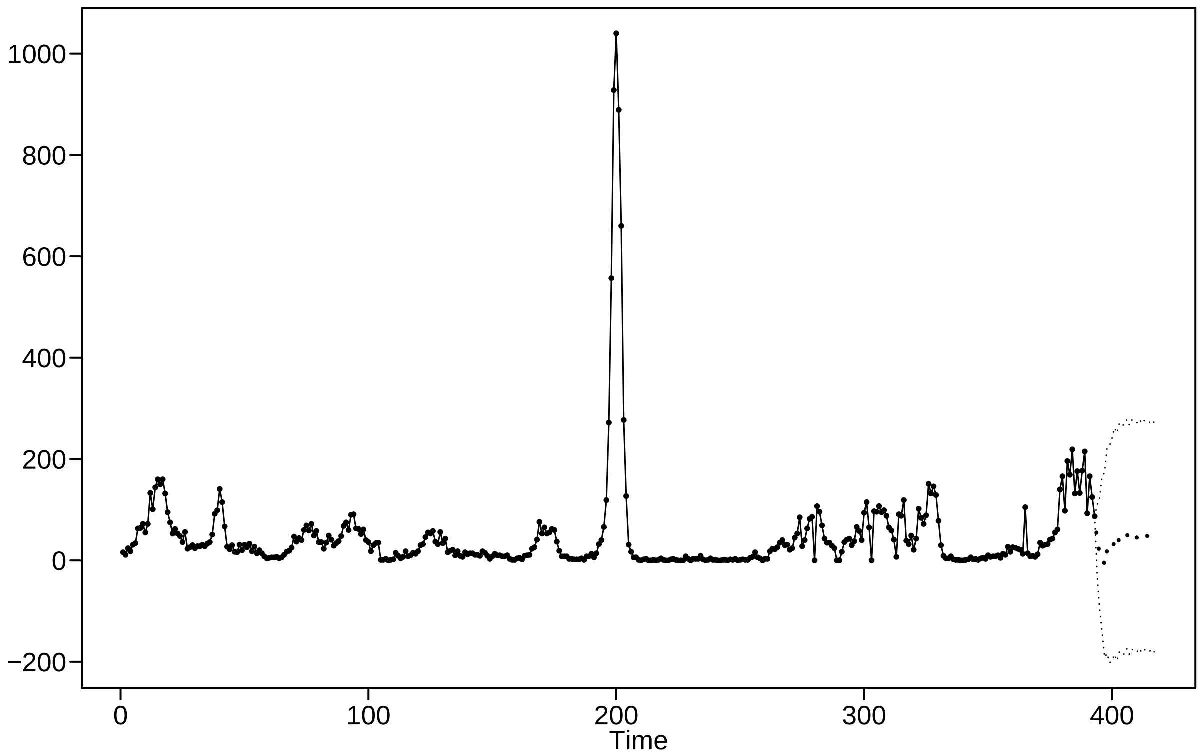


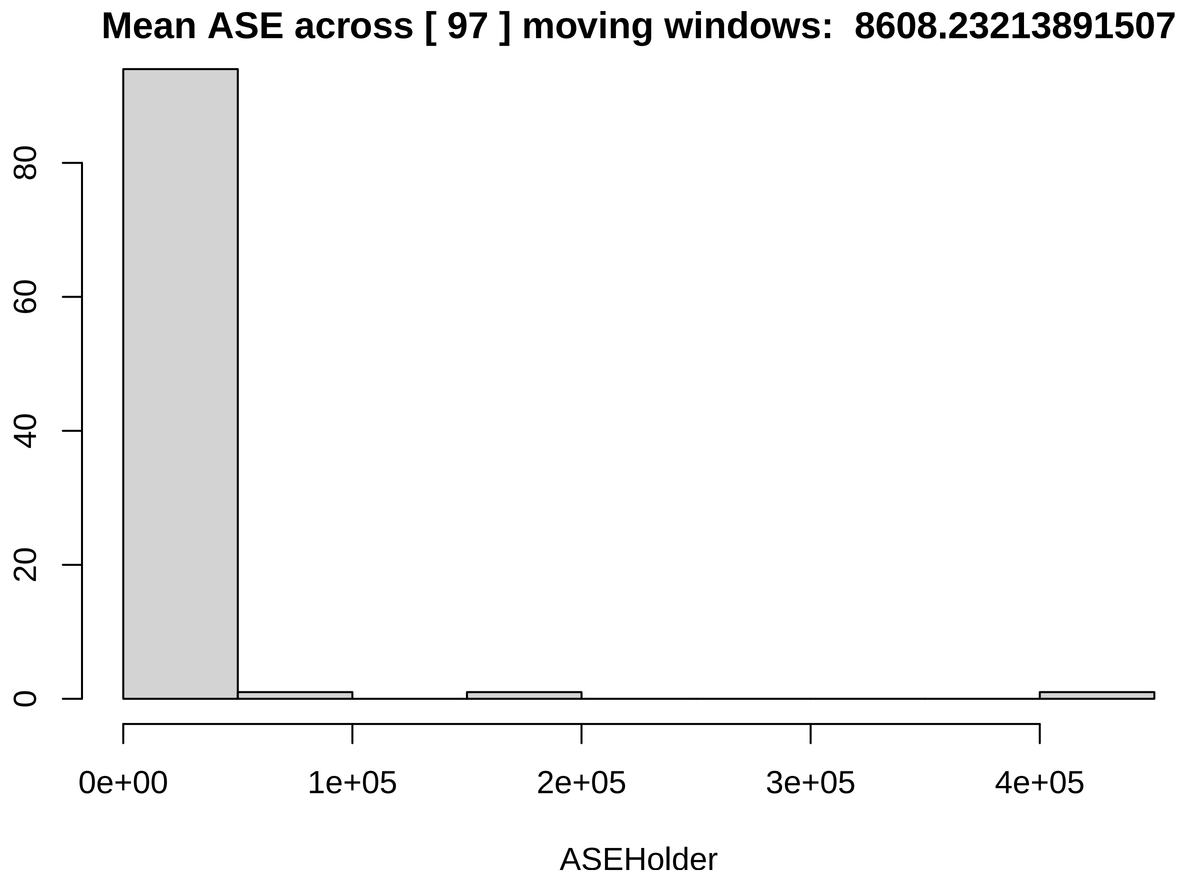

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



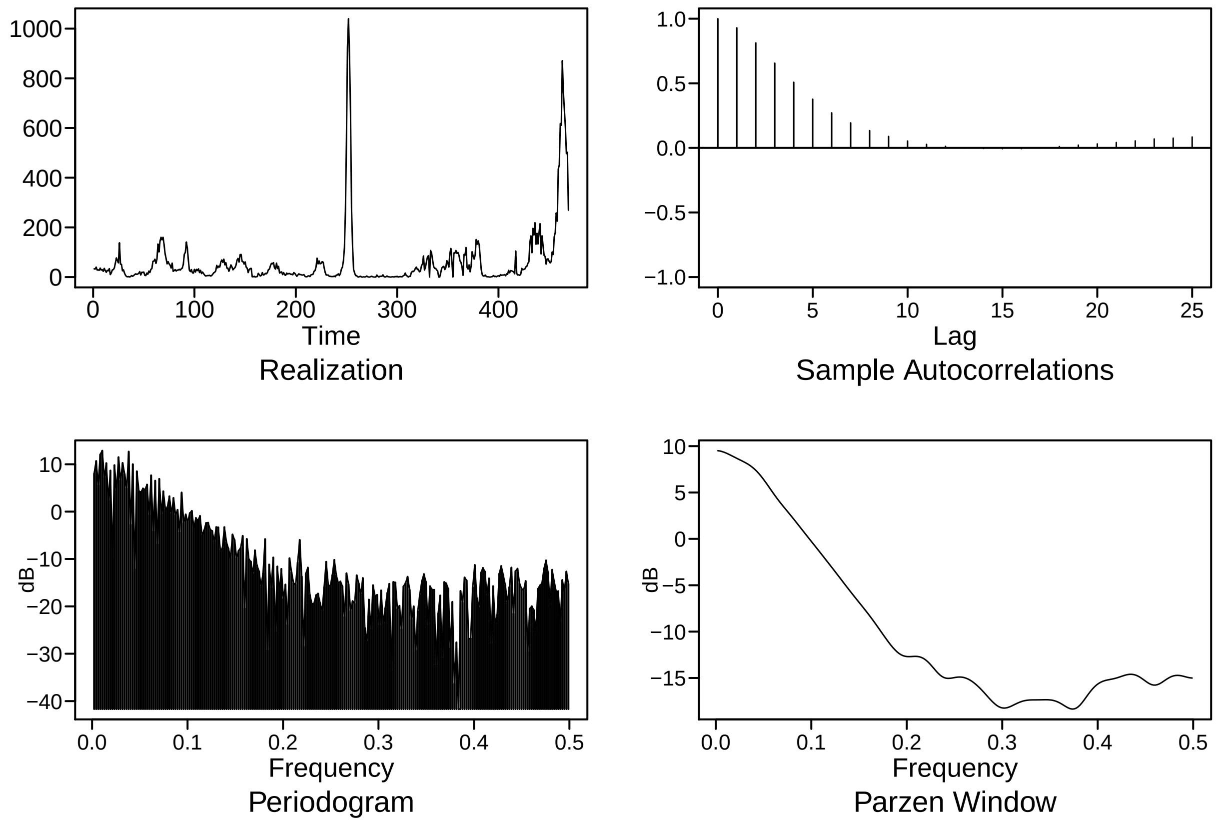



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## -0.1143 0.6302 0.4753 -0.4029
##
## Factor                 Roots                Abs Recip    System Freq
## 1+1.4581B+0.8613B^2    -0.8464+-0.6668i      0.9281       0.3938
## 1-1.3438B+0.4678B^2     1.4362+-0.2735i      0.6840       0.0299
##
##
```

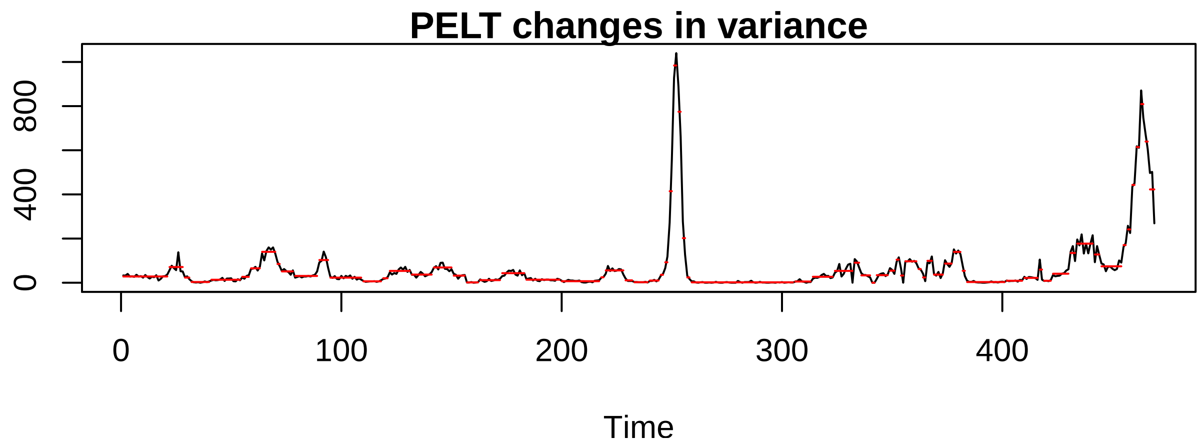

1000
800
600
400
200
0
−200
0
100
200
300
400
Time

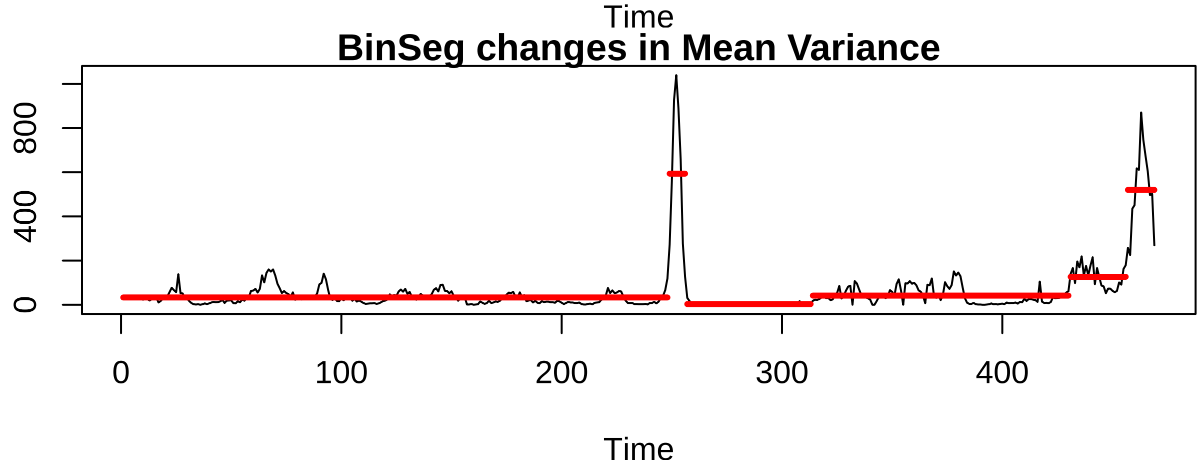



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

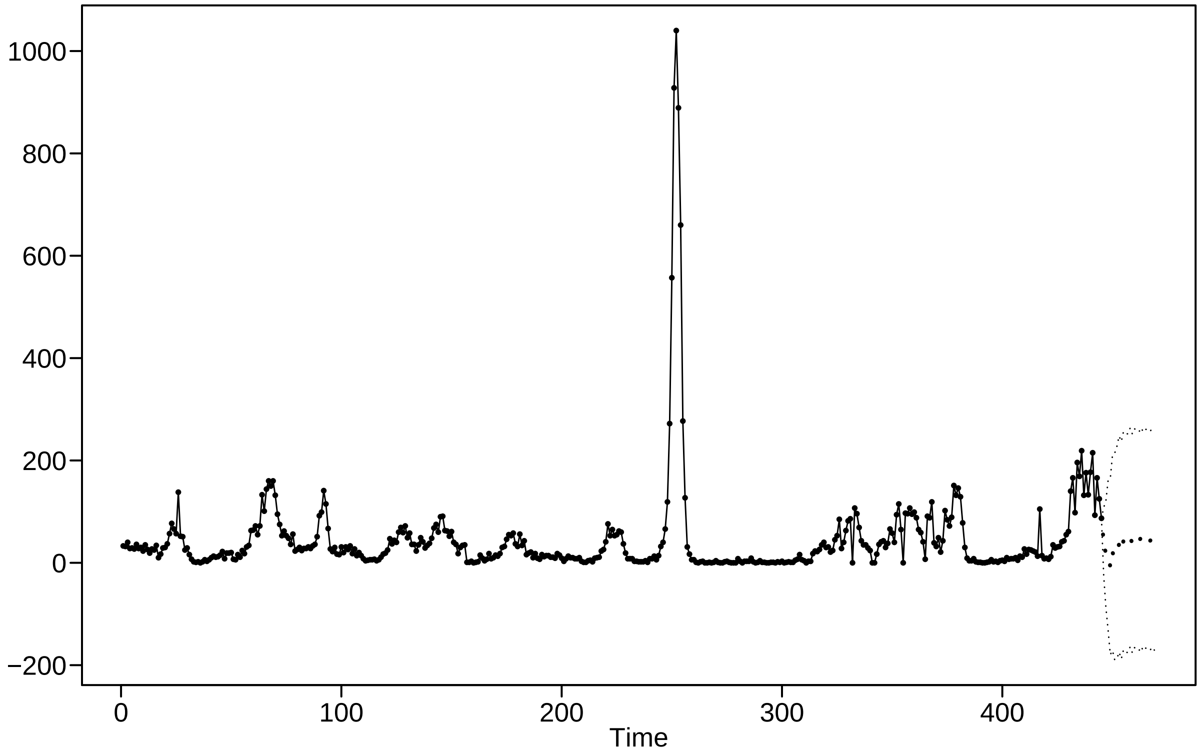


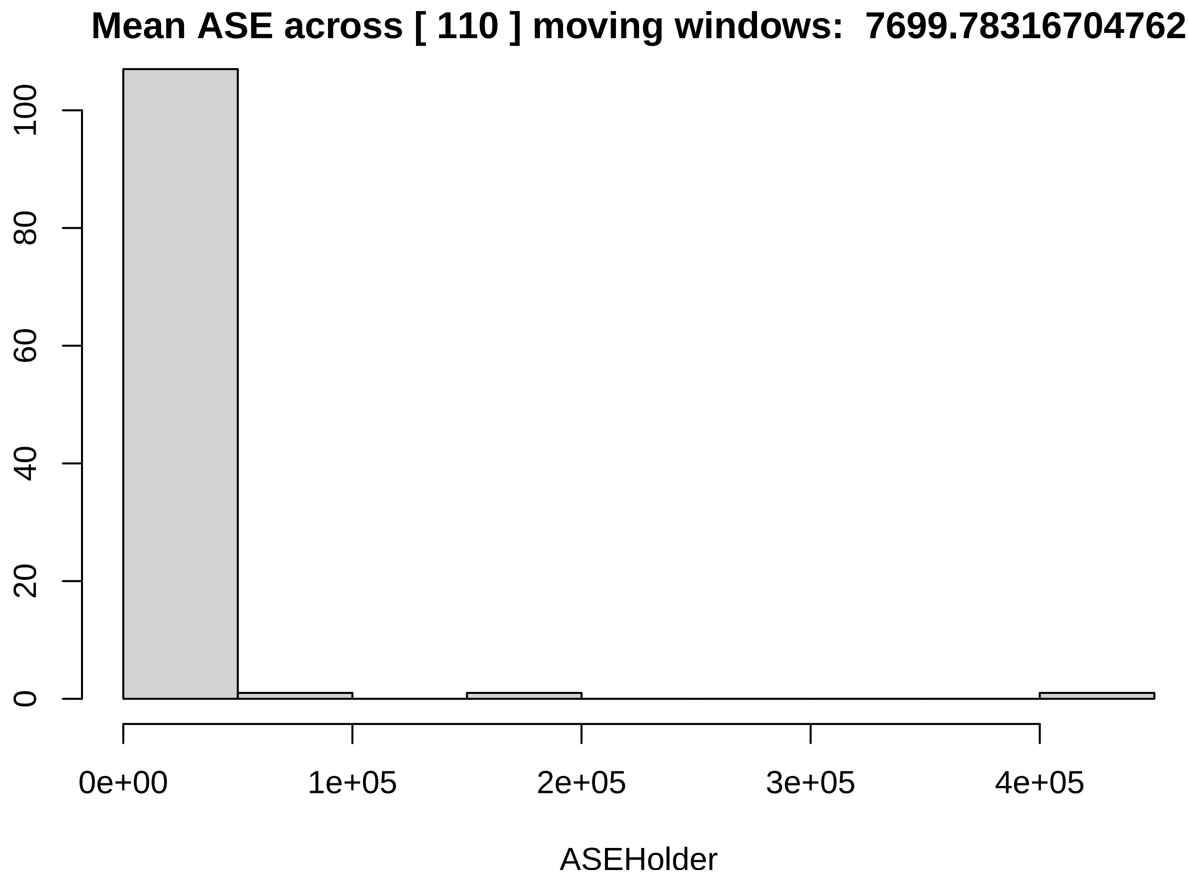

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



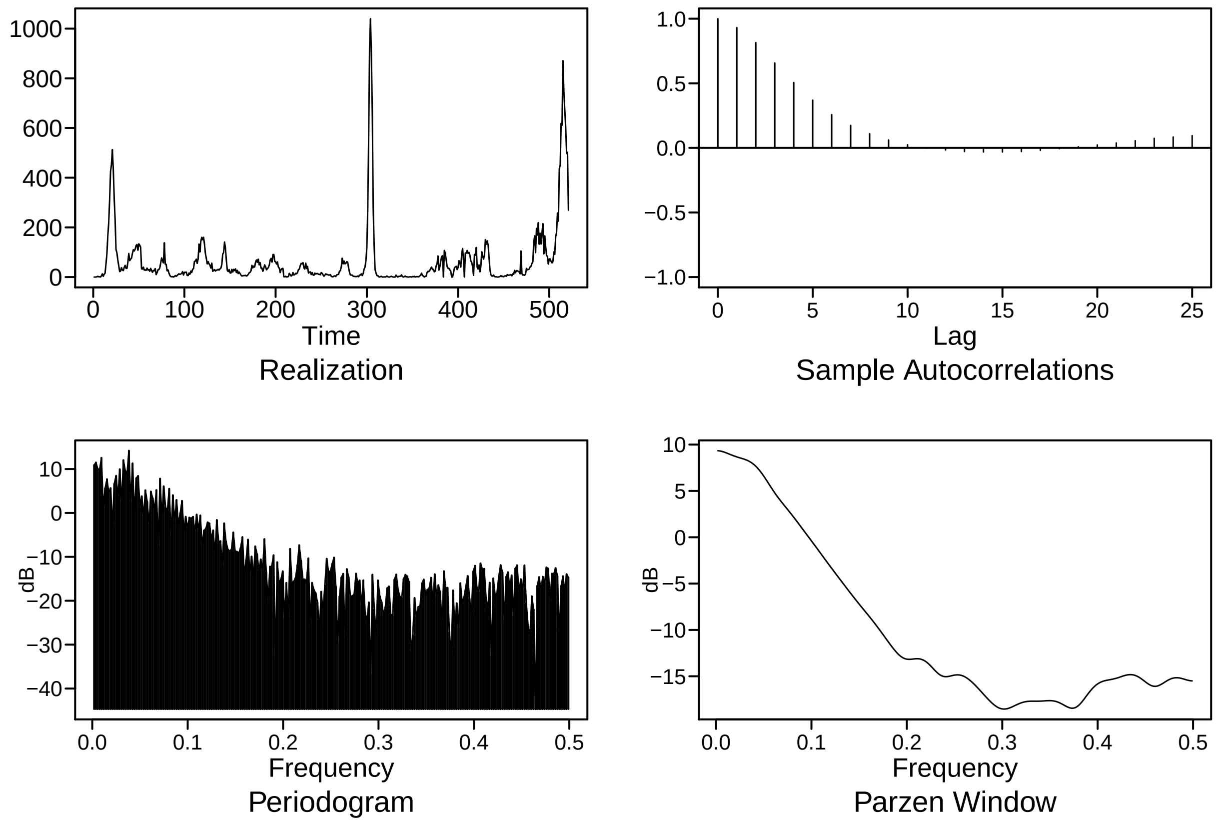



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## -0.1165 0.6216 0.4895 -0.4040
##
## Factor                 Roots                Abs Recip    System Freq
## 1+1.4580B+0.8697B^2   -0.8382+-0.6688i      0.9326       0.3928
## 1-1.3415B+0.4645B^2    1.4440+-0.2603i      0.6815       0.0284
##
##
```

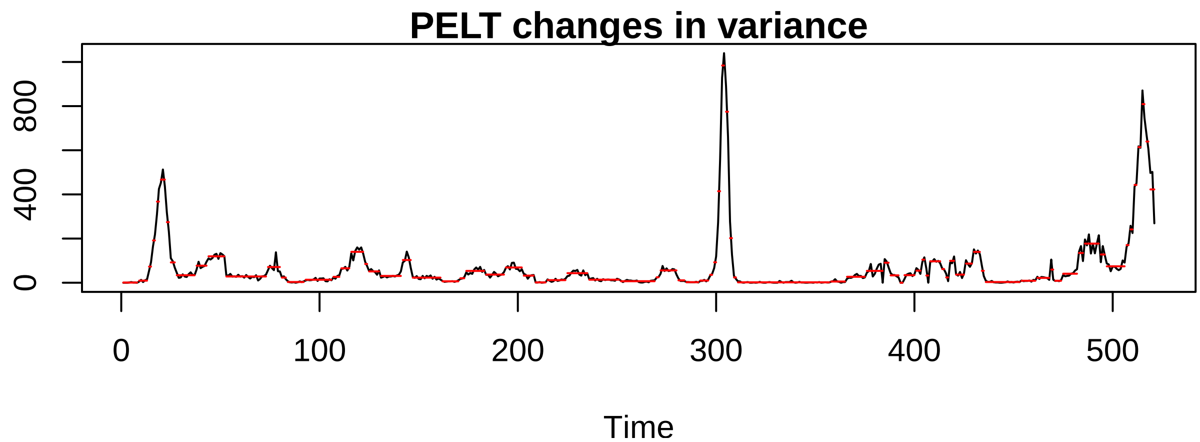

1000
800
600
400
200
0
−200
0
100
200
300
400
Time

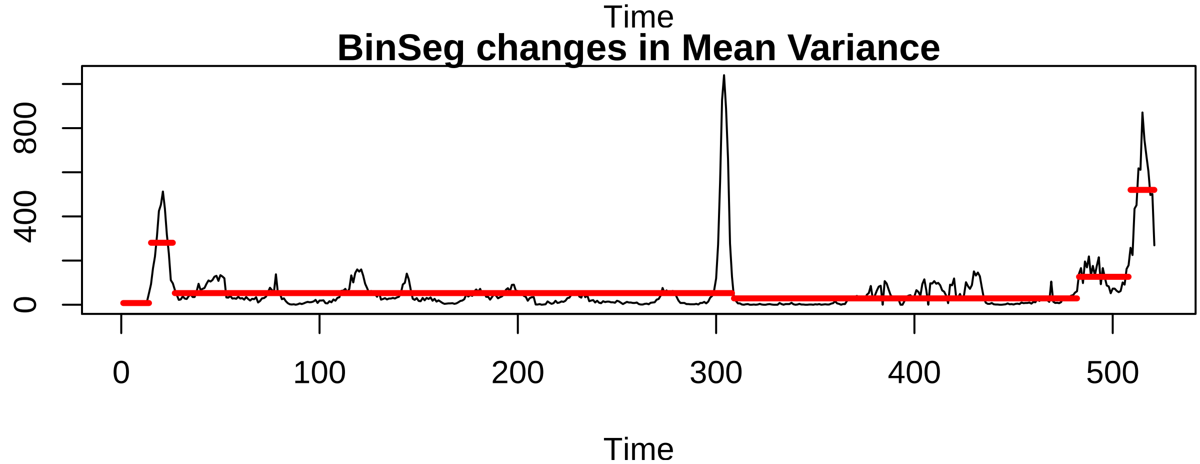



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

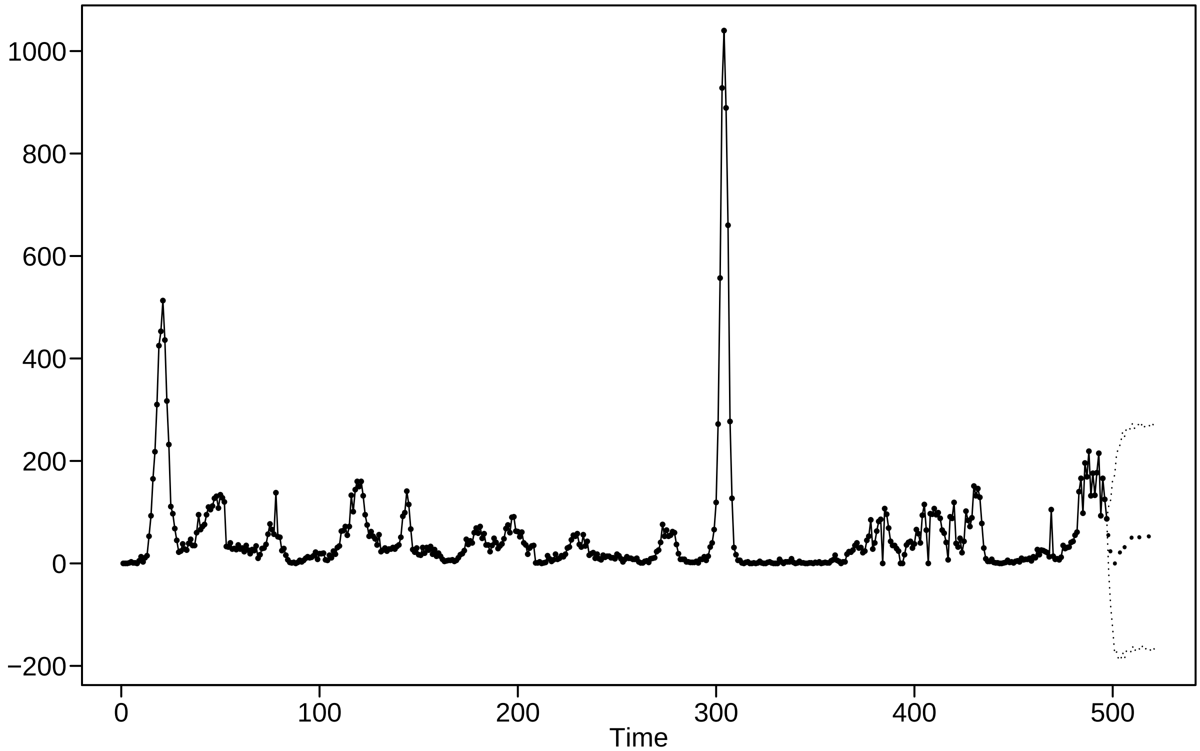


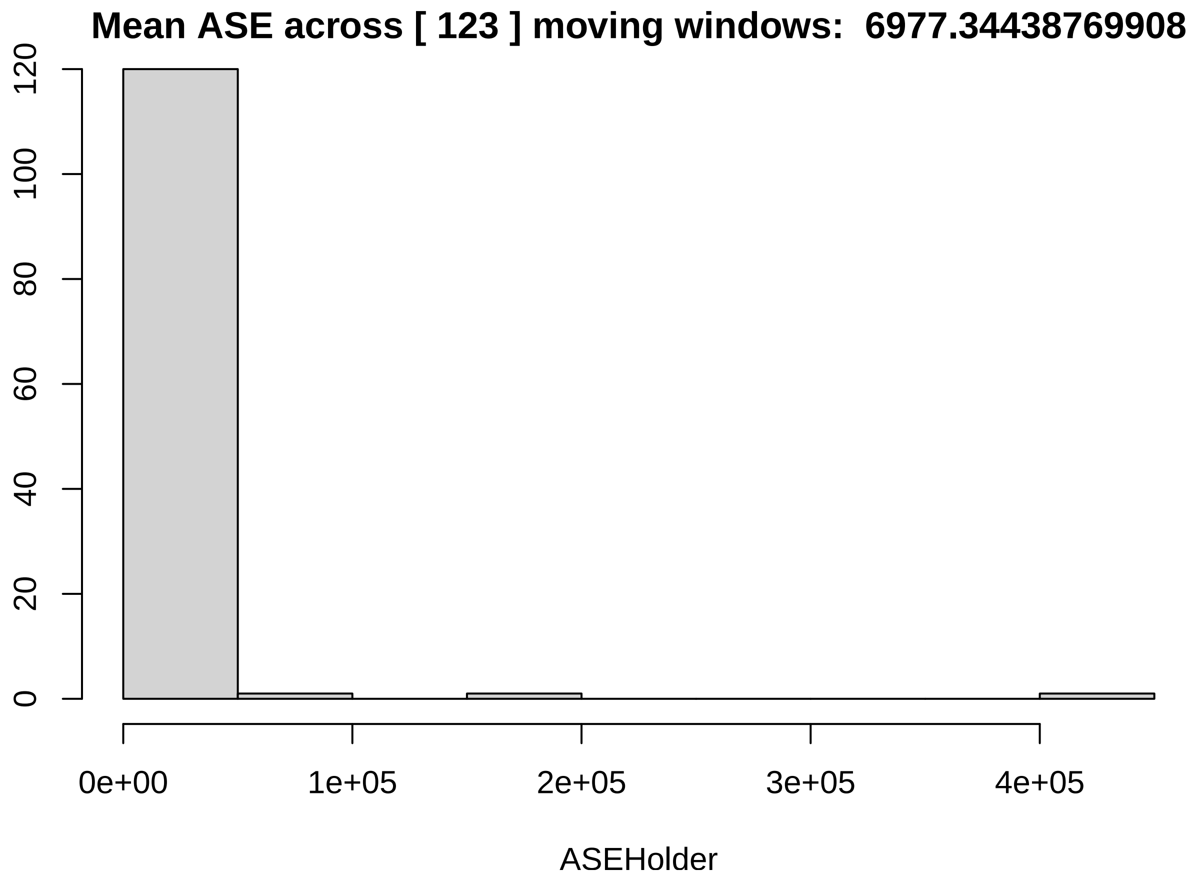

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



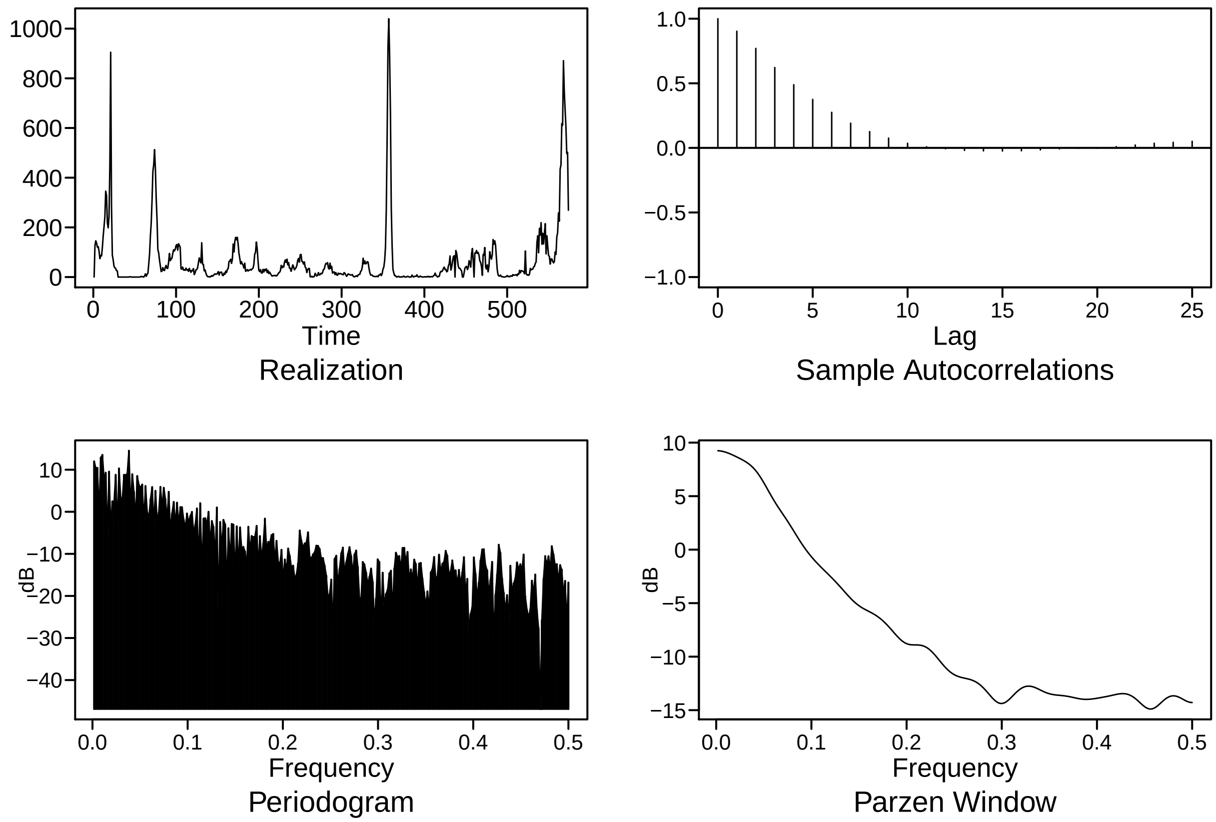



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## -0.0711 0.6272 0.4848 -0.4360
##
## Factor                 Roots                Abs Recip    System Freq
## 1+1.4504B+0.8753B^2   -0.8286+-0.6753i      0.9356       0.3912
## 1-1.3794B+0.4982B^2    1.3845+-0.3011i      0.7058       0.0341
##
##
```

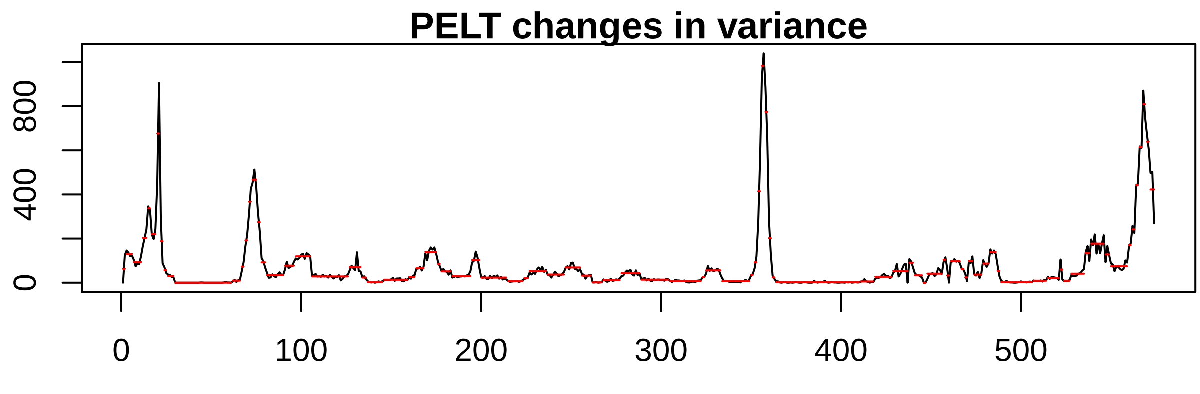

1000
800
600
400
200
0
−200
0
100
200
300
400
500
Time

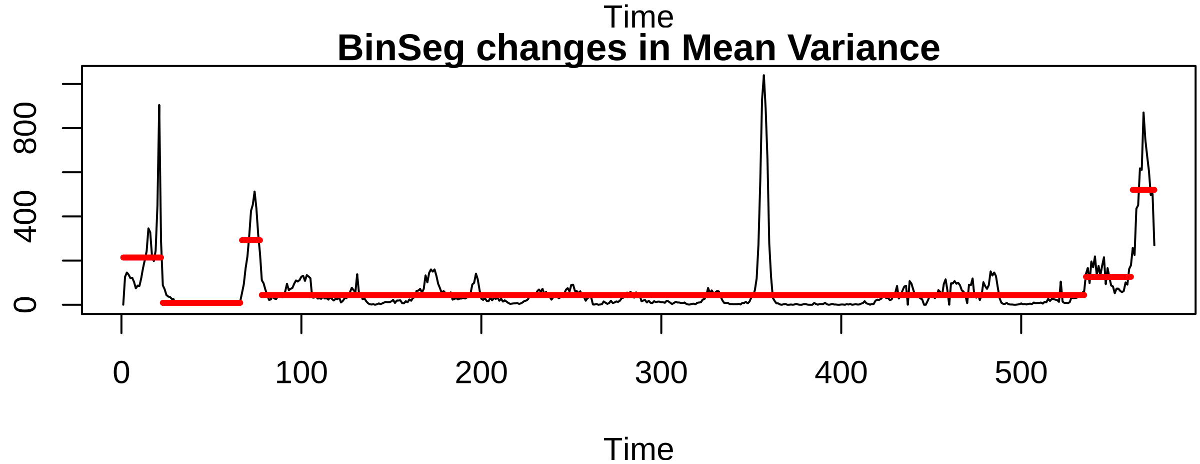



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

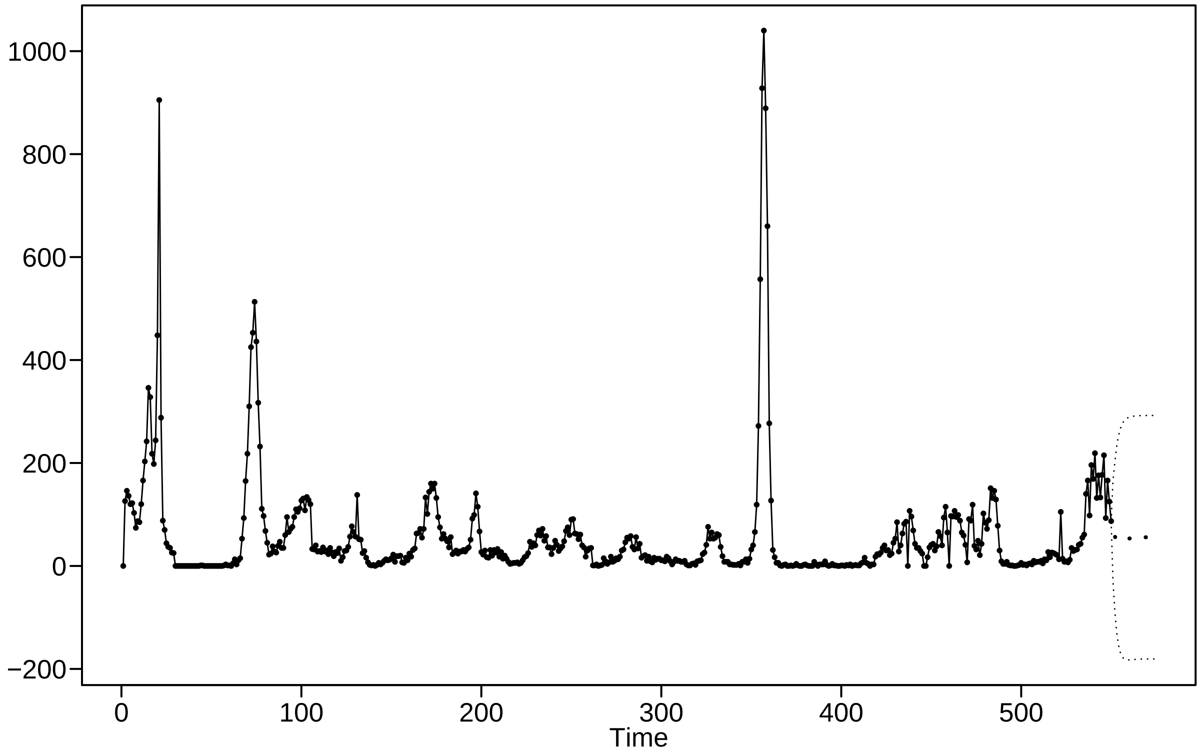


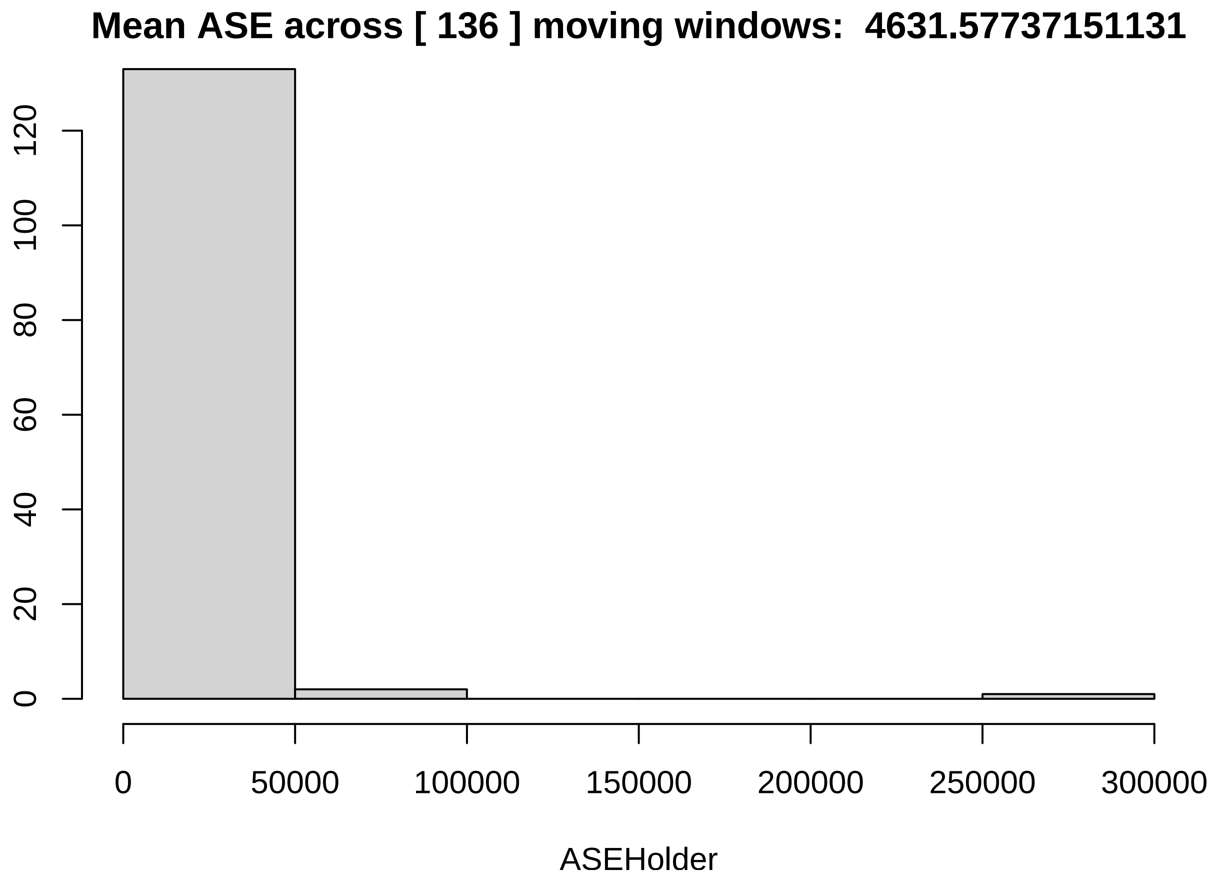

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



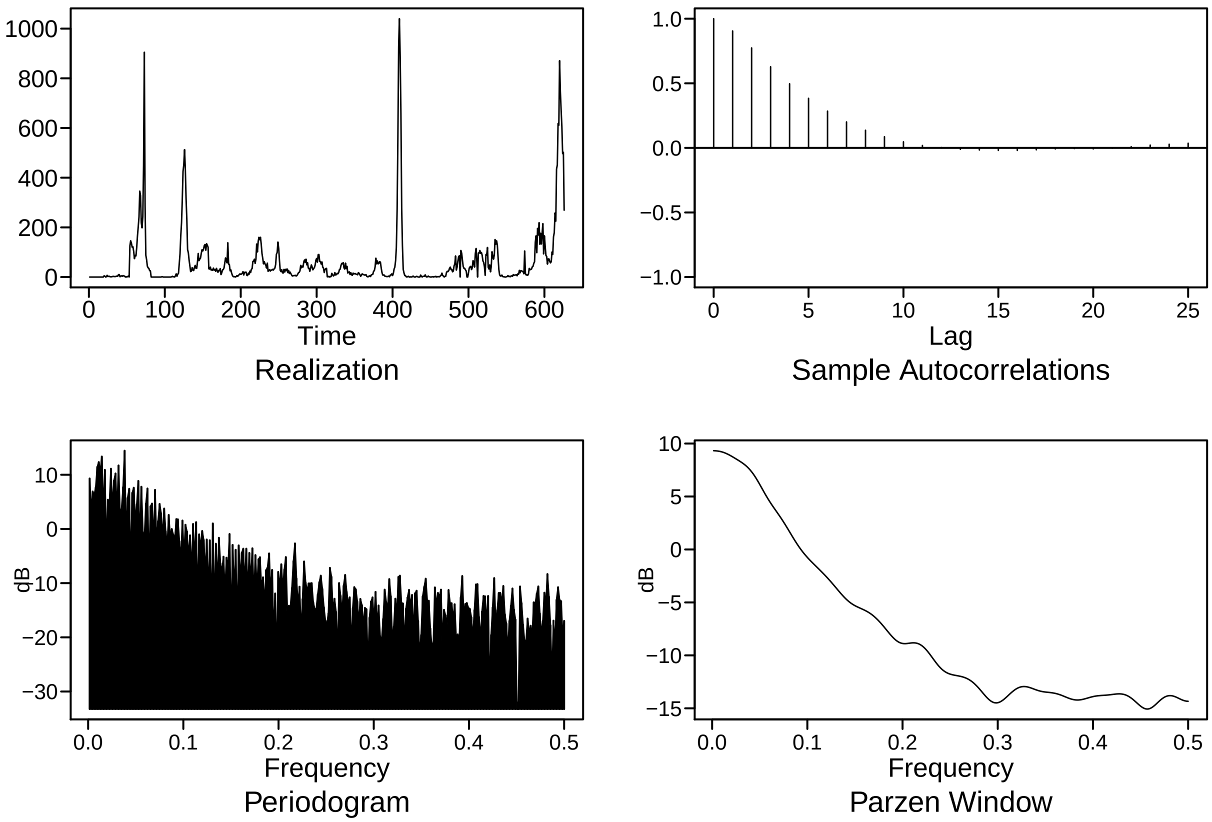



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 1.0929 -0.0890 -0.1389
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.3796B+0.4845B^2    1.4236+-0.1927i      0.6961       0.0214
## 1+0.2867B             -3.4878               0.2867       0.5000
##
##
```

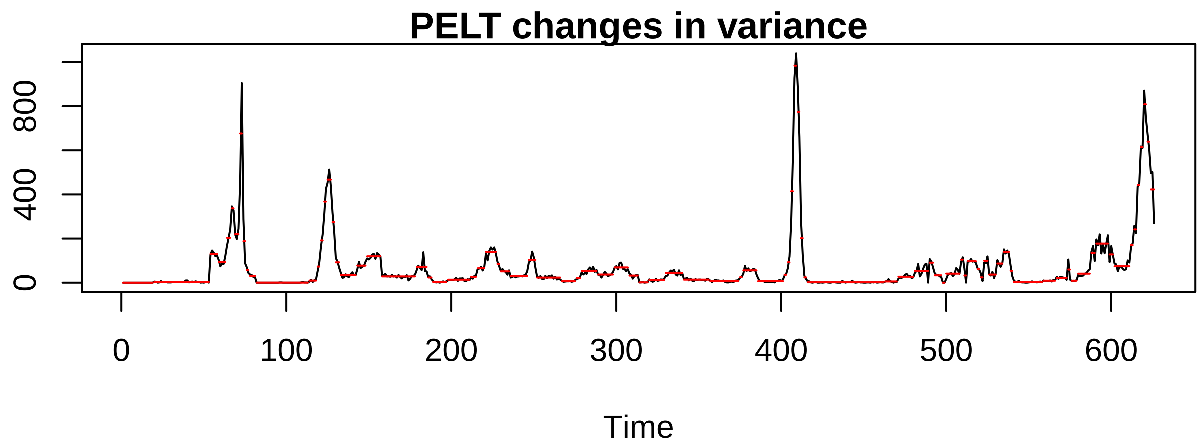

1000
800
600
400
200
0
−200
0
100
200
300
400
500
Time

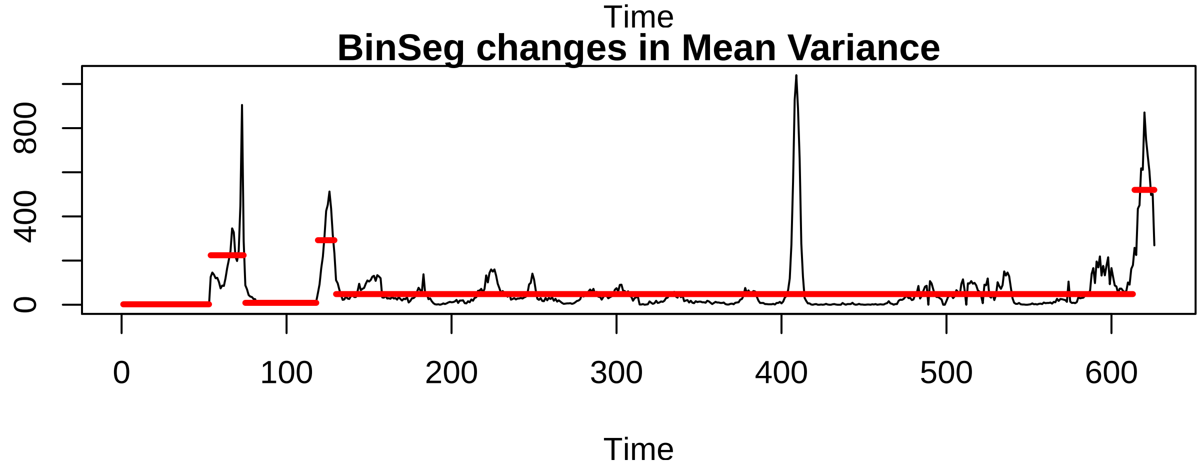



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

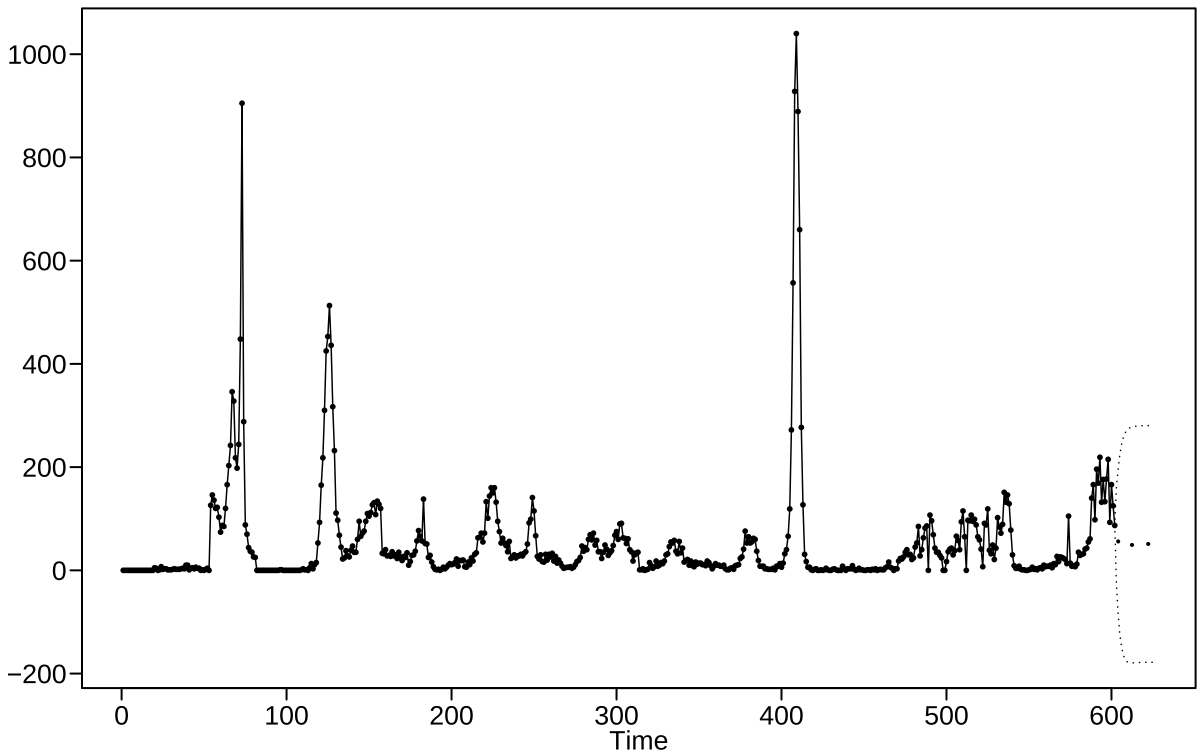


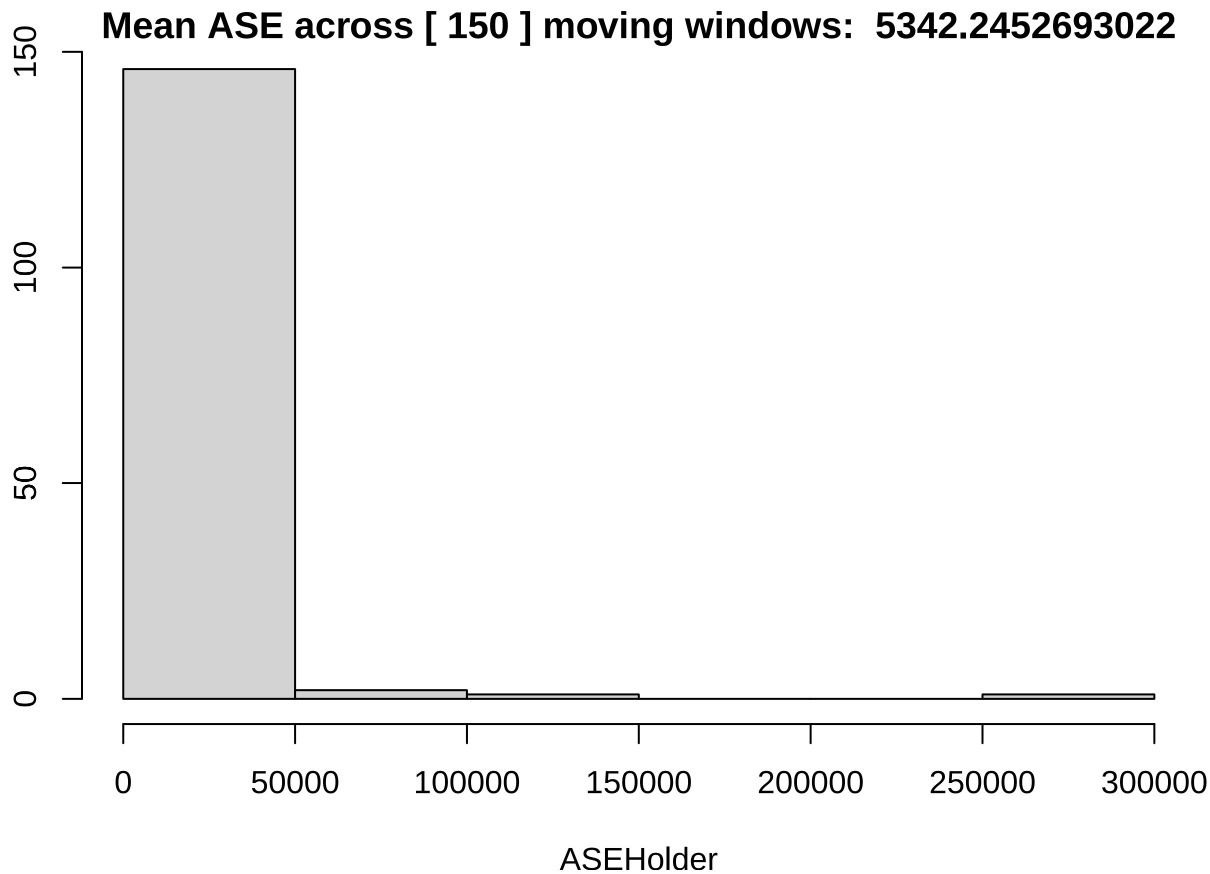

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



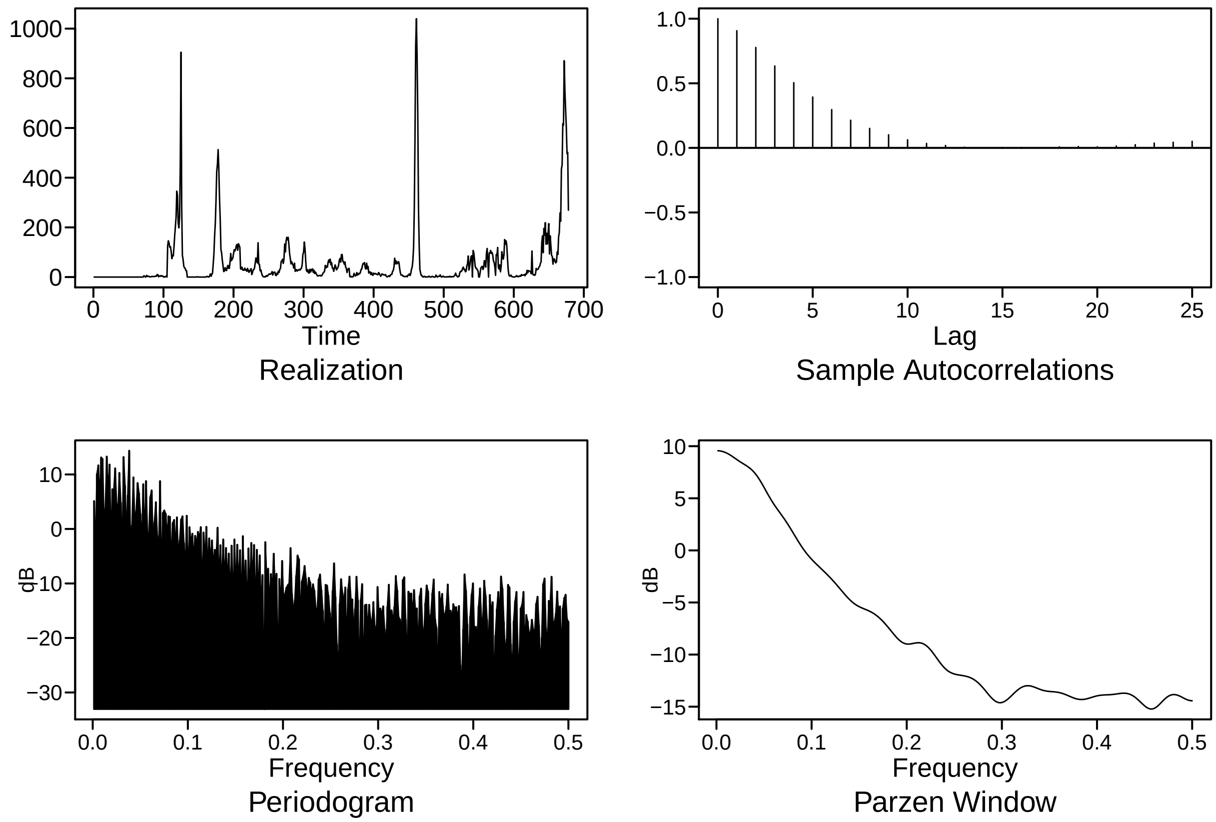



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 1.0930 -0.0882 -0.1367
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.3776B+0.4803B^2    1.4340+-0.1596i      0.6931       0.0176
## 1+0.2846B             -3.5135               0.2846       0.5000
##
##
```

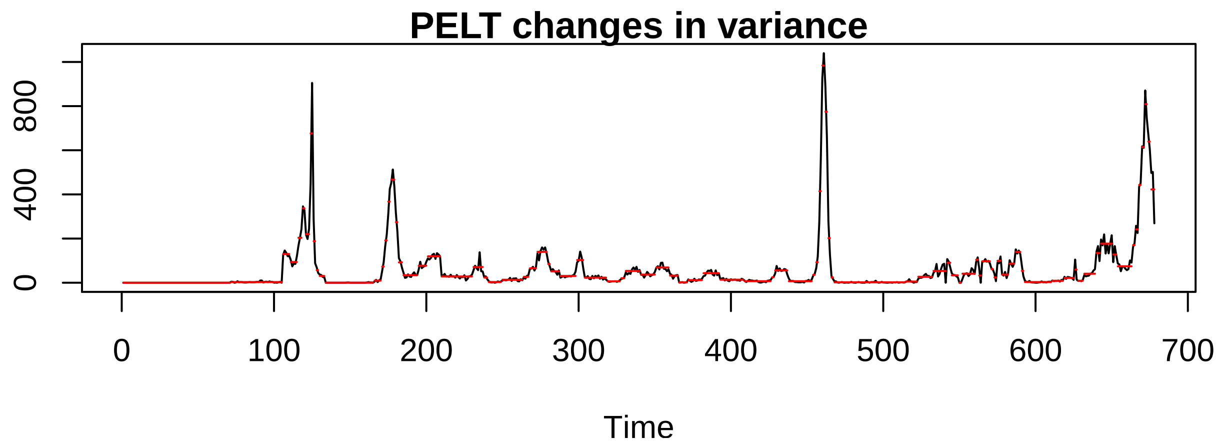

1000
800
600
400
200
0
−200
0
100
200
300
400
500
600
Time

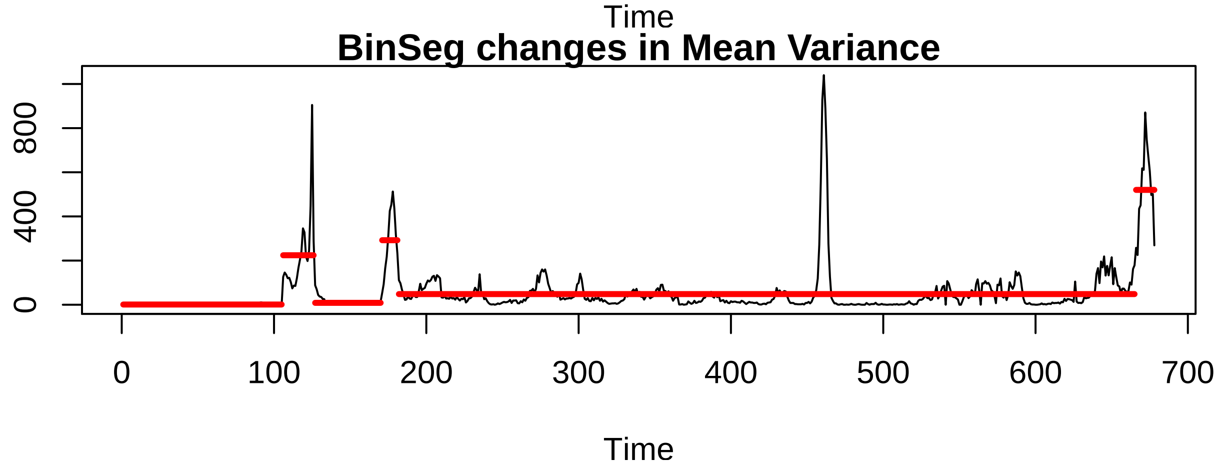



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

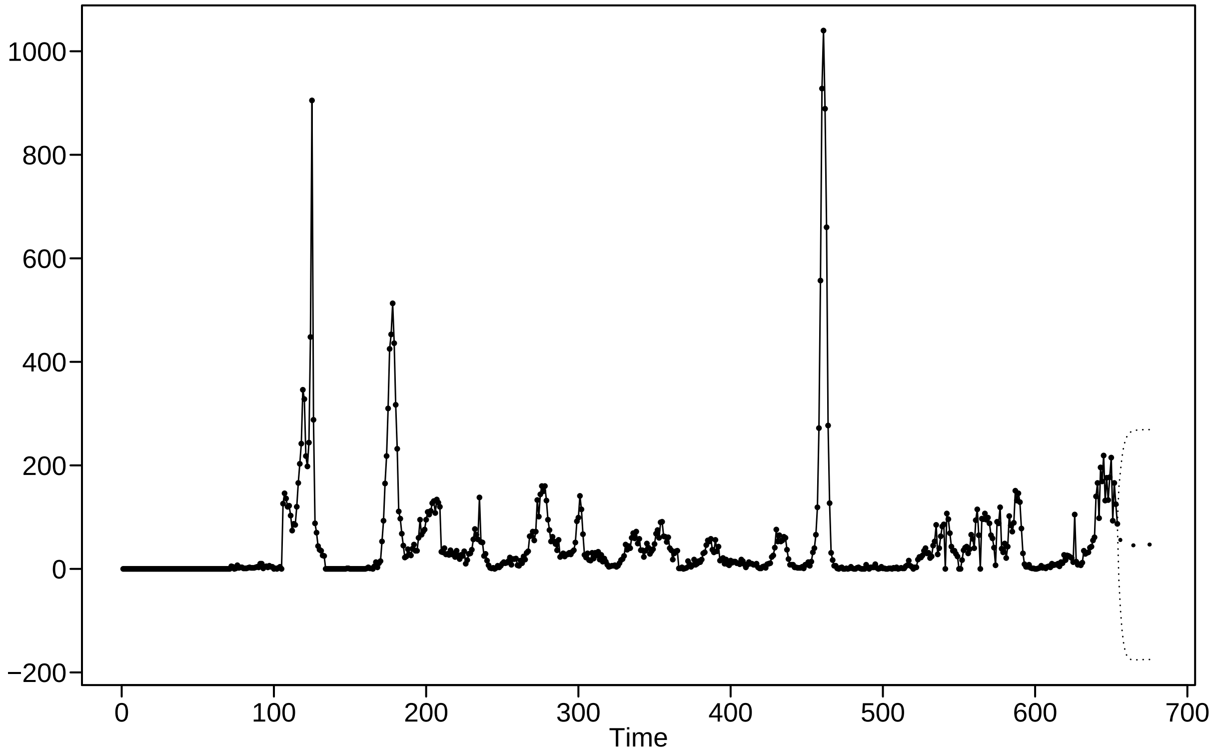


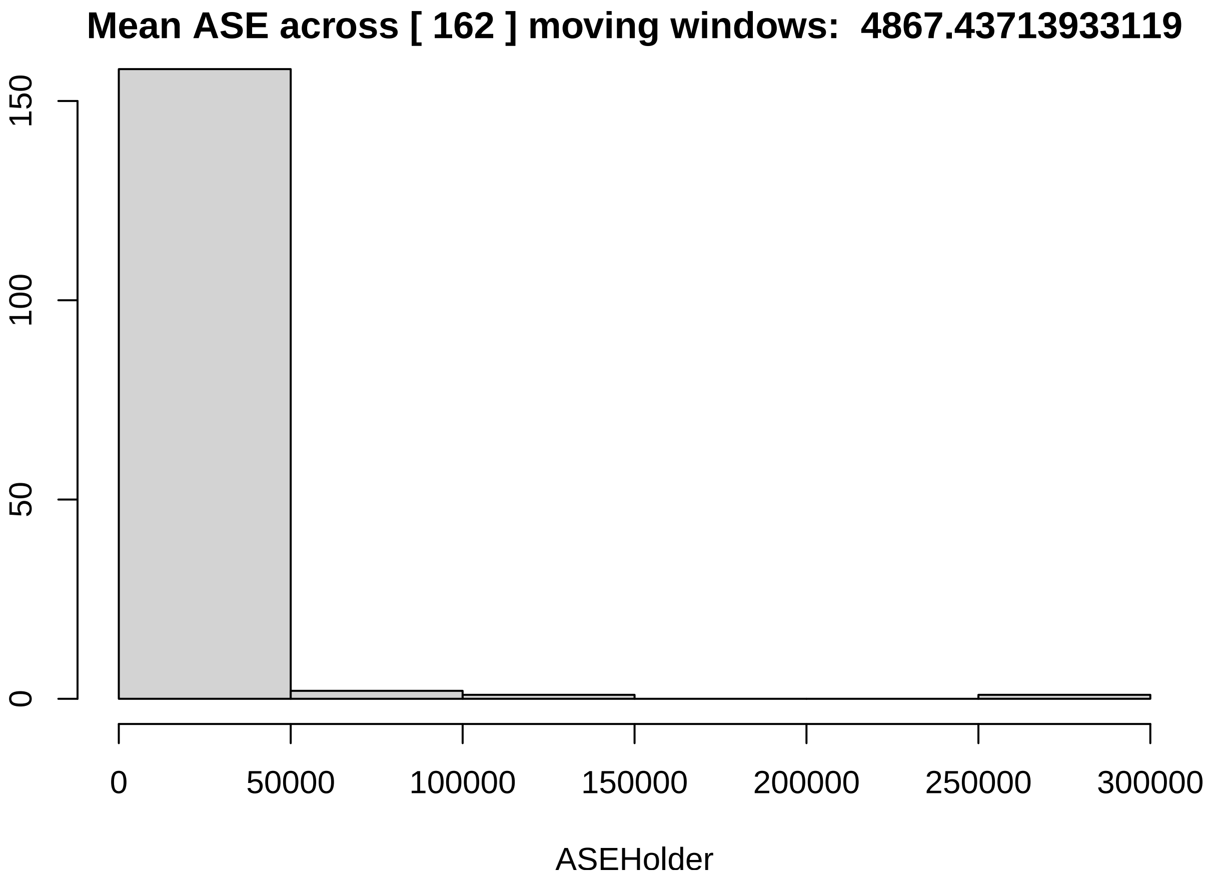

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



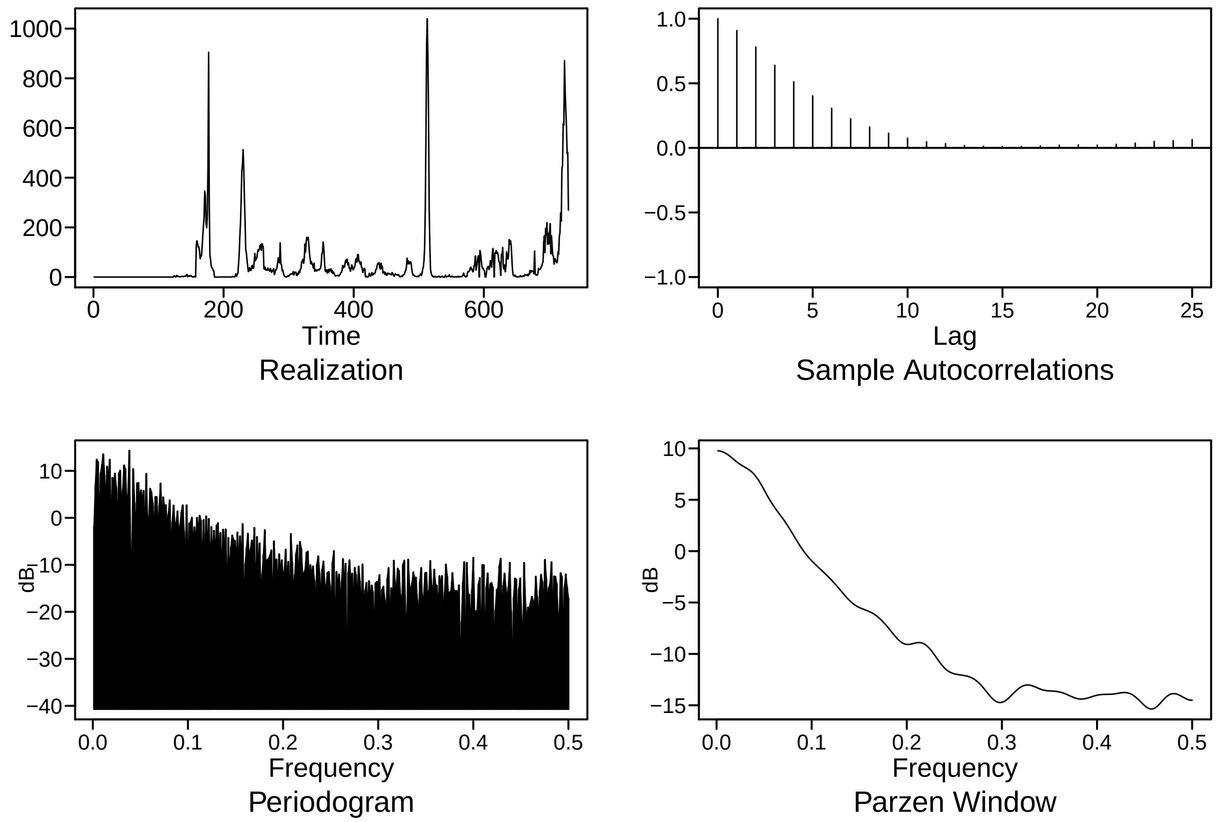



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 1.0948 -0.0889 -0.1351
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.3773B+0.4781B^2    1.4406+-0.1287i      0.6914       0.0142
## 1+0.2826B             -3.5389               0.2826       0.5000
##
##
```

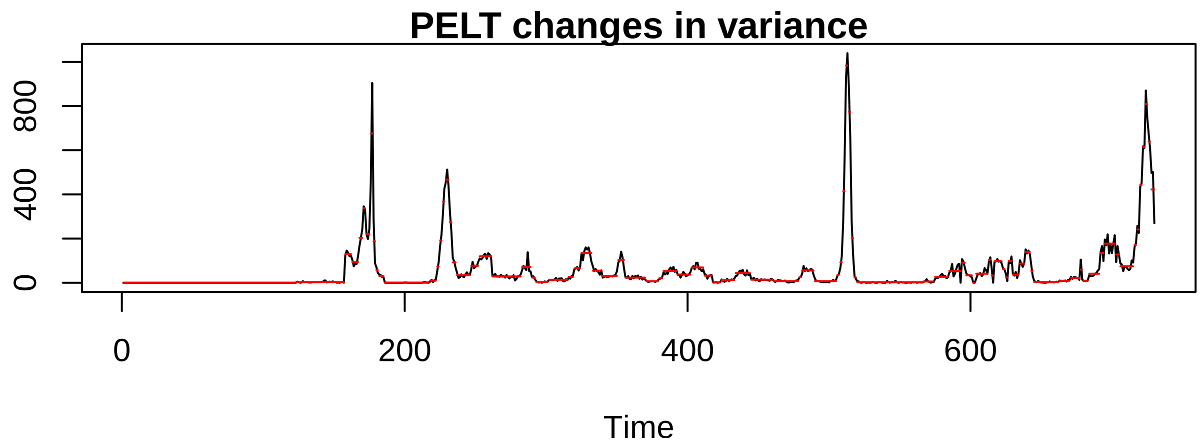

1000
800
600
400
200
0
−200
0
100
200
300
400
500
600
700
Time

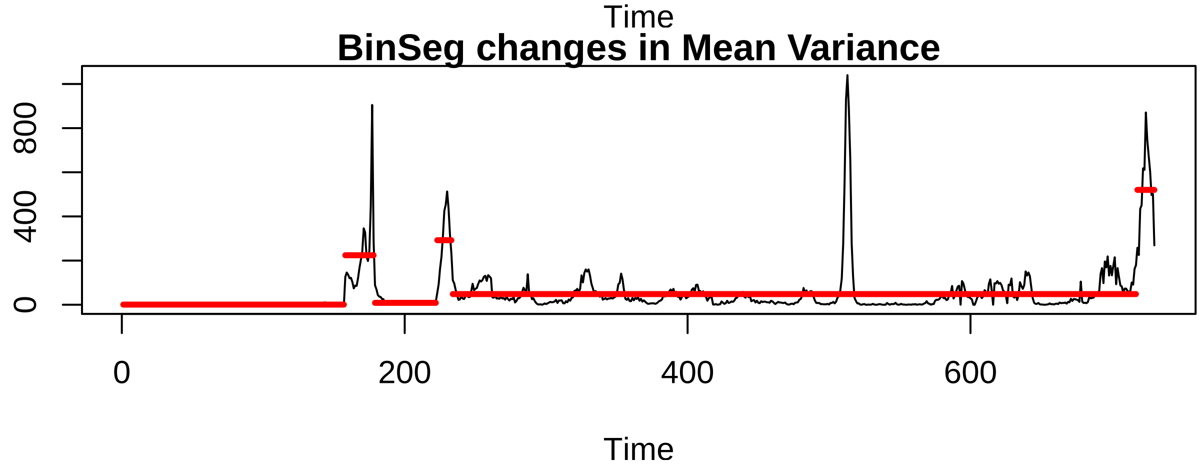



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

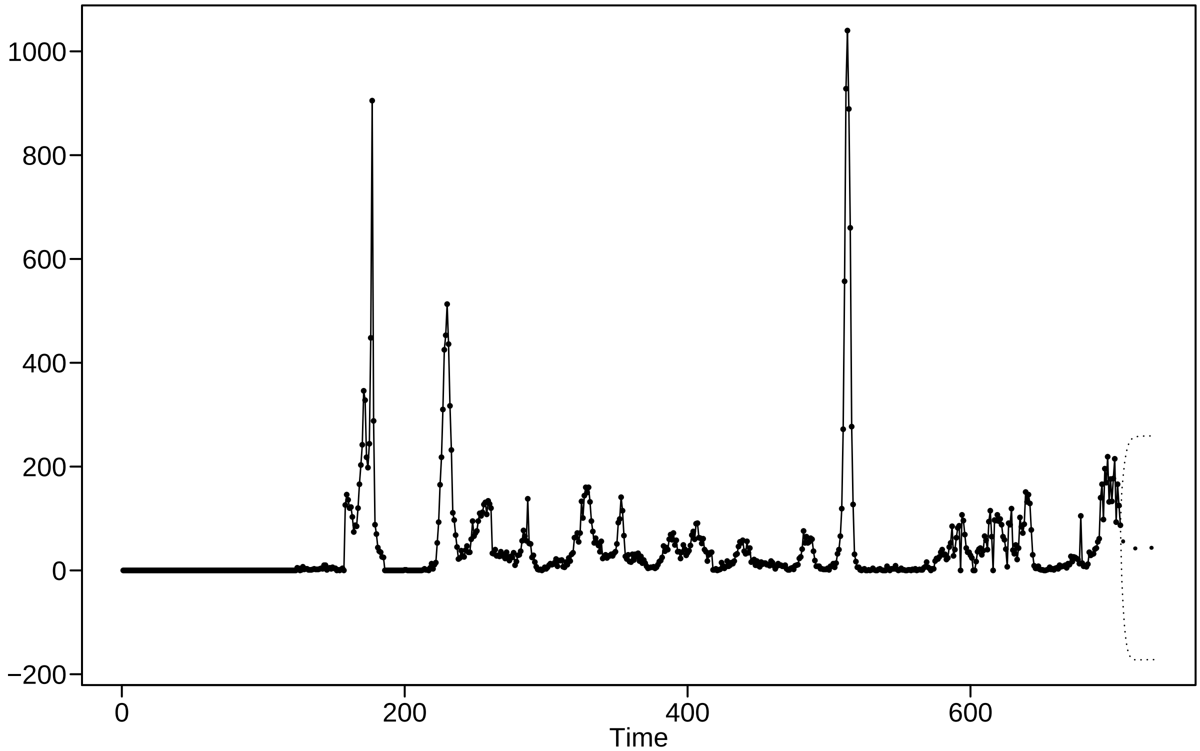


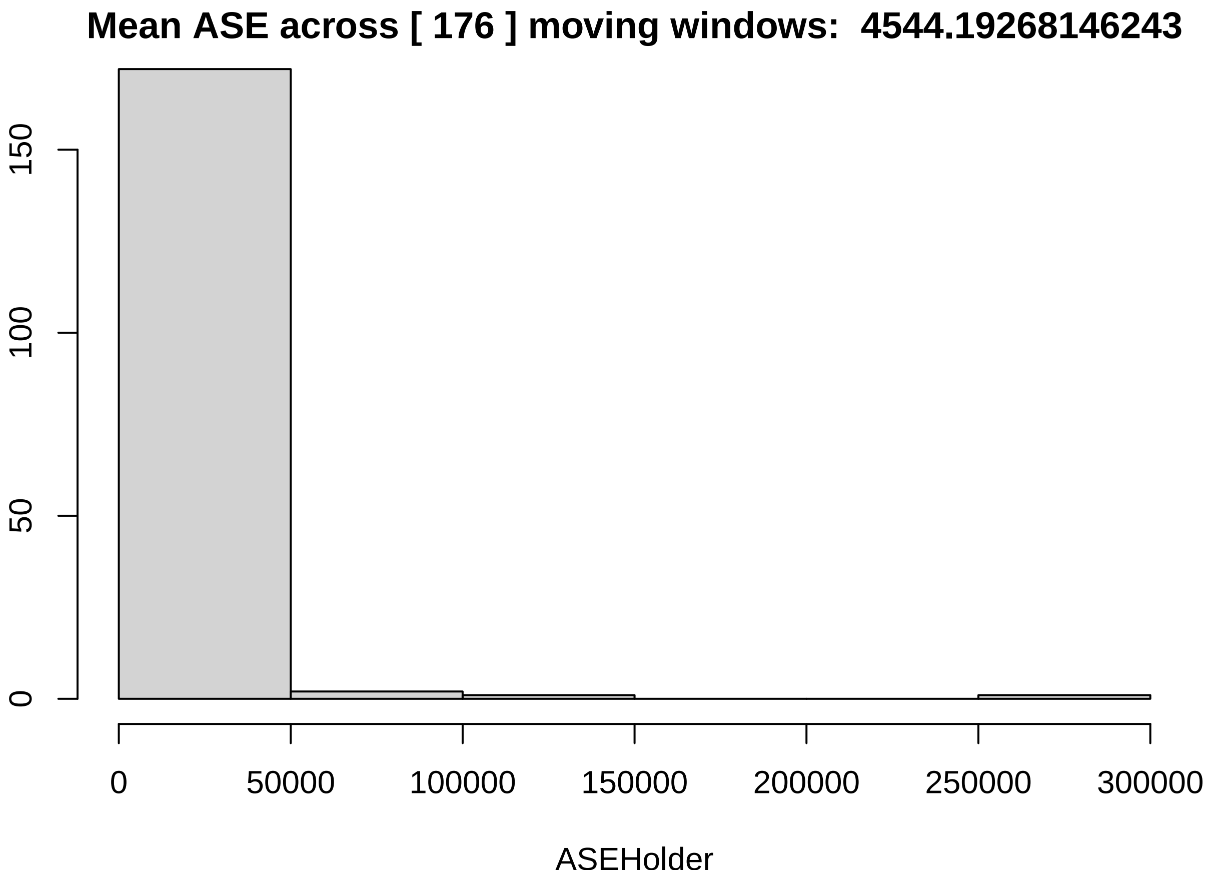

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



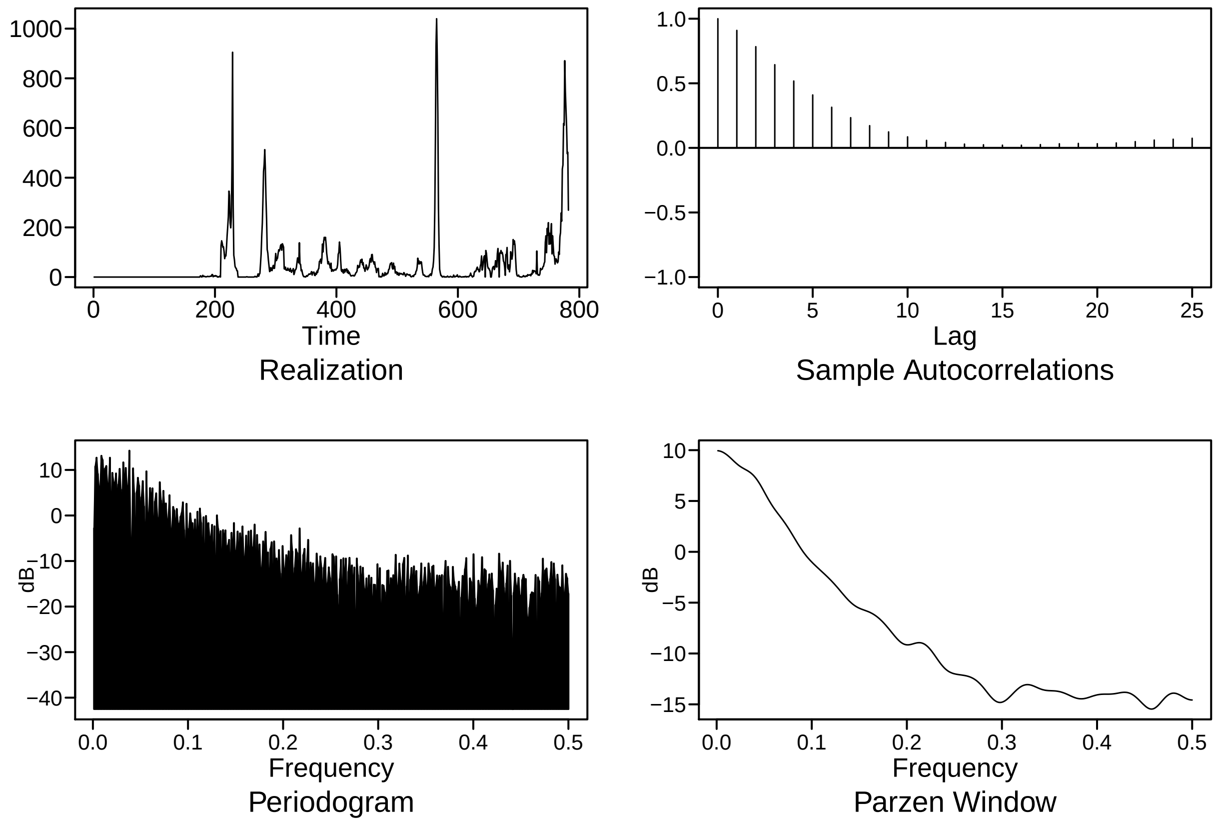



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 1.0963 -0.0894 -0.1338
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.3771B+0.4762B^2    1.4460+-0.0953i      0.6901       0.0105
## 1+0.2809B             -3.5601               0.2809       0.5000
##
##
```

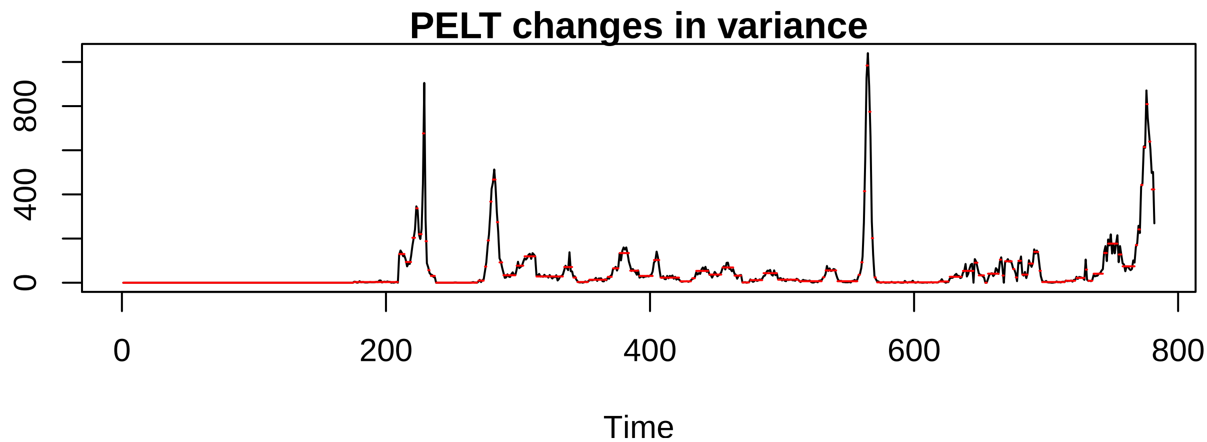

1000
800
600
400
200
0
−200
0
200
400
600
Time

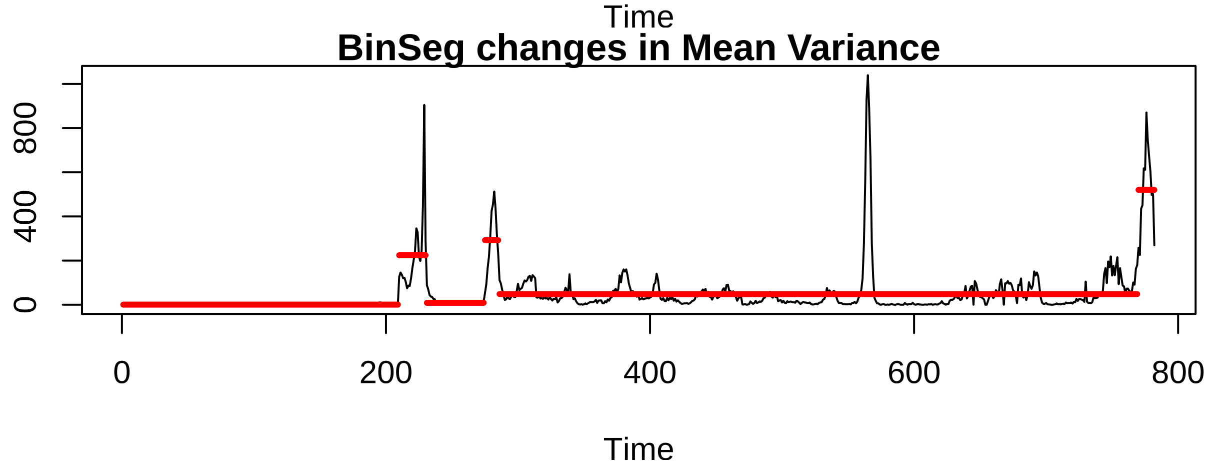



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

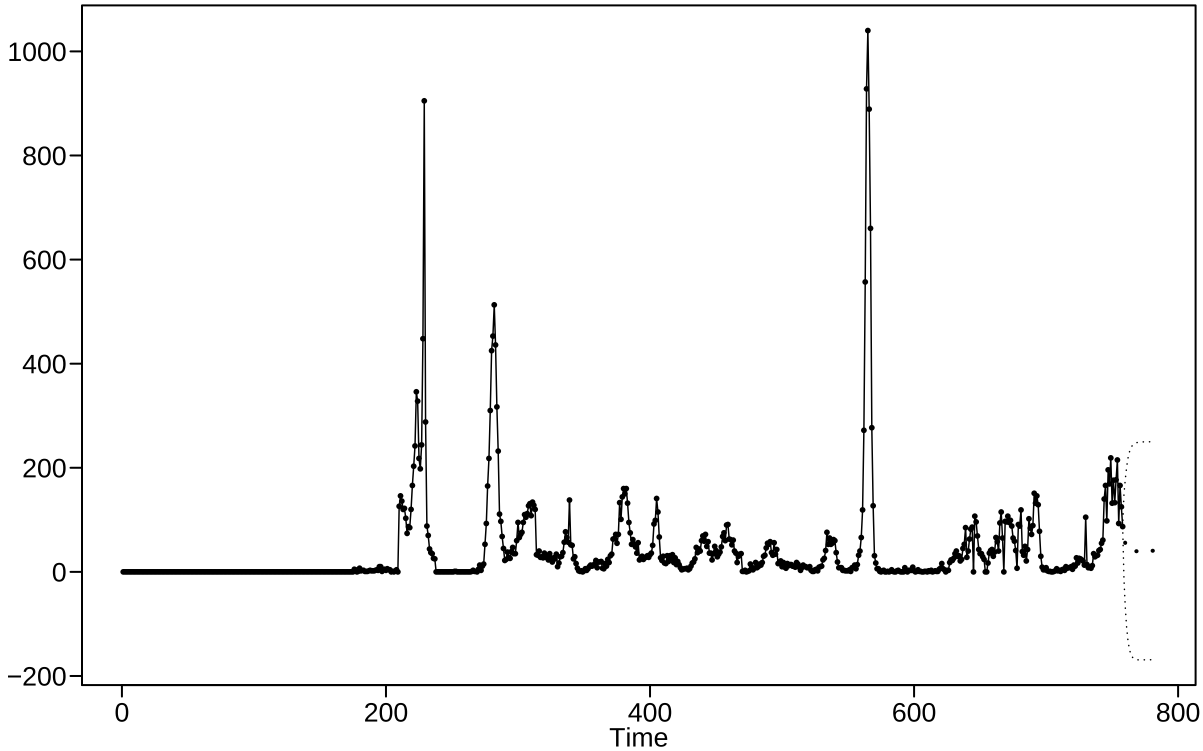


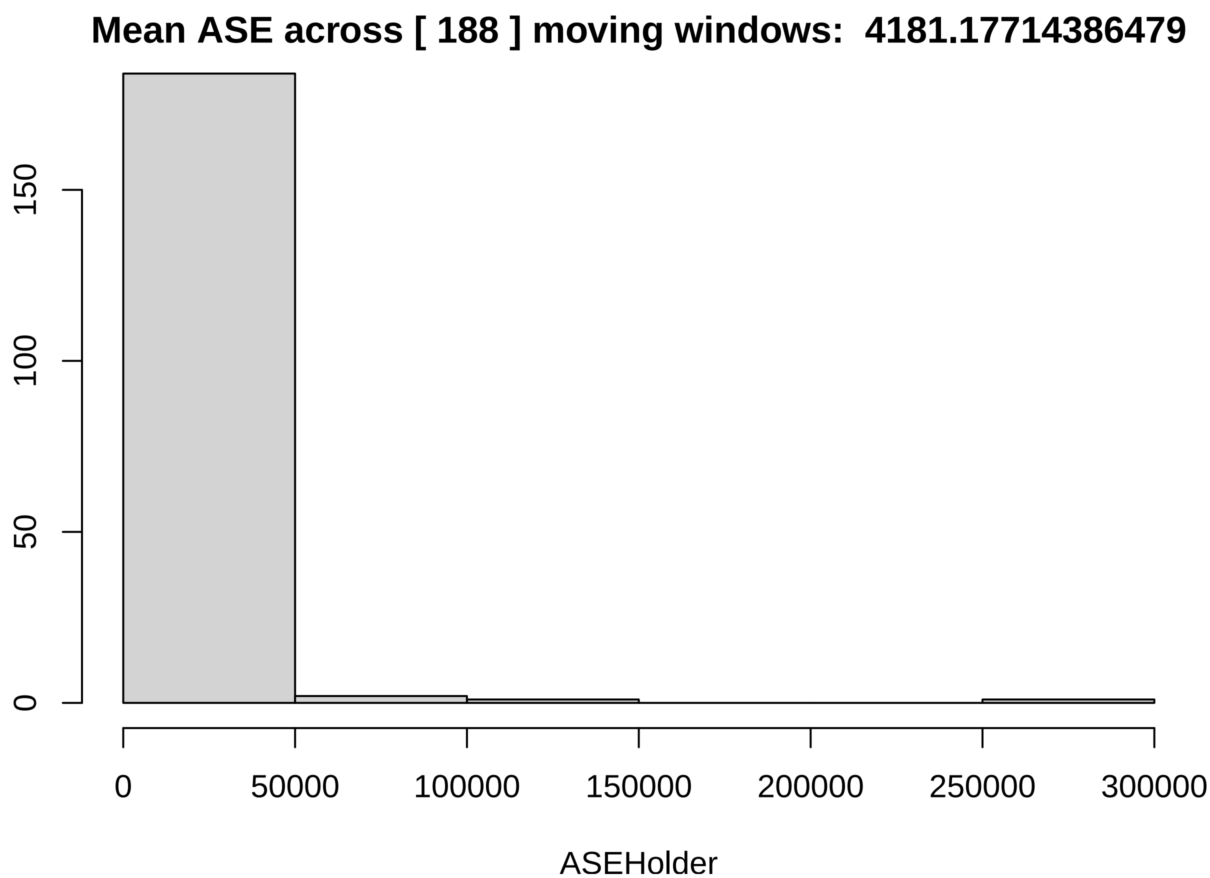

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



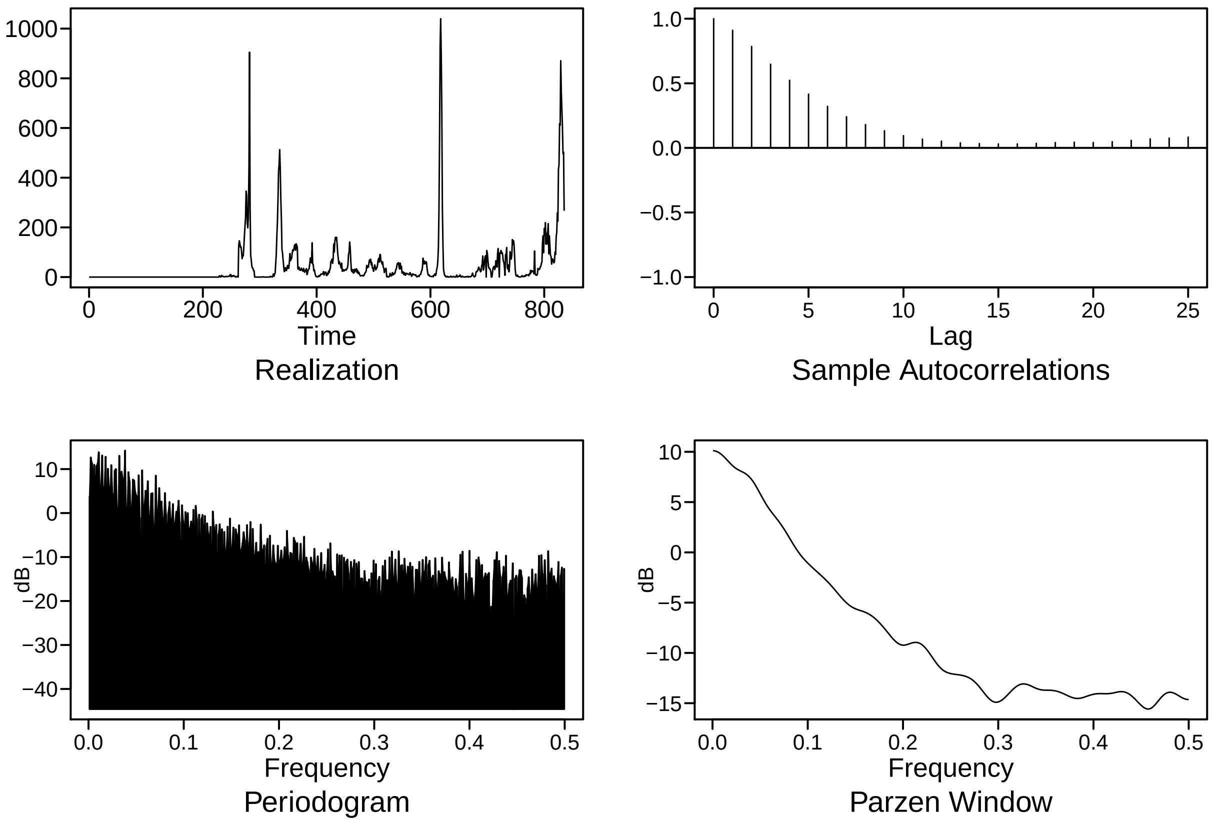



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 1.0975 -0.0898 -0.1326
##
## Factor                 Roots                Abs Recip    System Freq
## 1-1.3770B+0.4746B^2    1.4507+-0.0515i      0.6889       0.0056
## 1+0.2795B             -3.5784               0.2795       0.5000
##
##
```

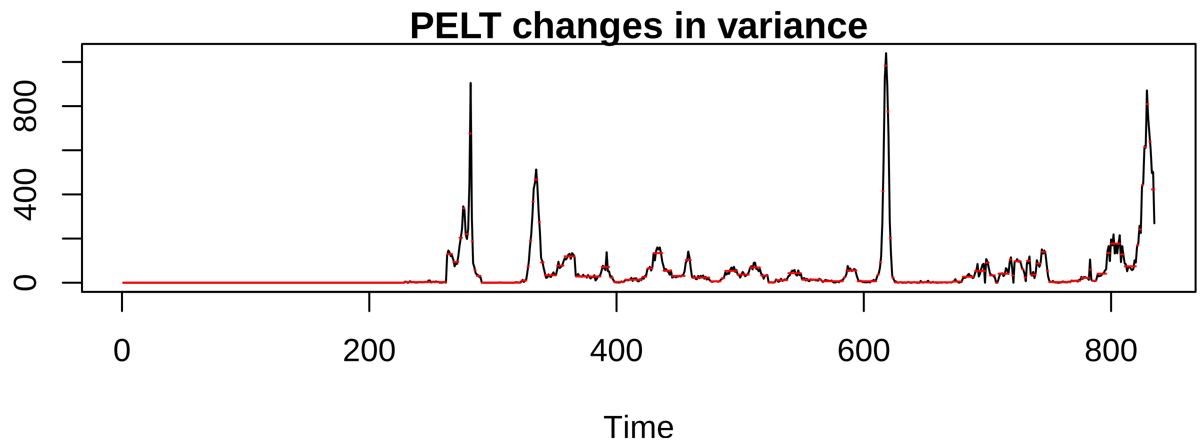

1000
800
600
400
200
0
−200
0
200
400
600
800
Time

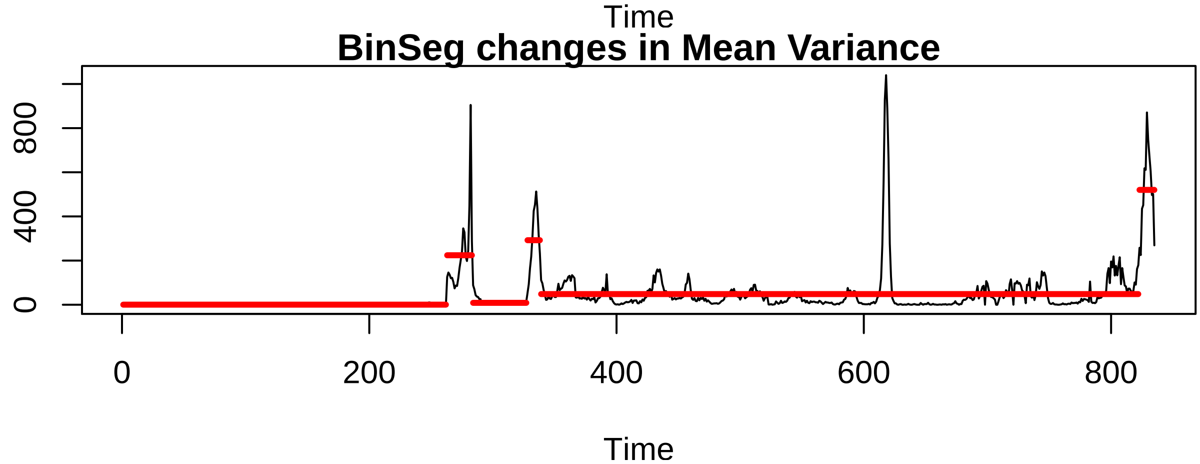



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

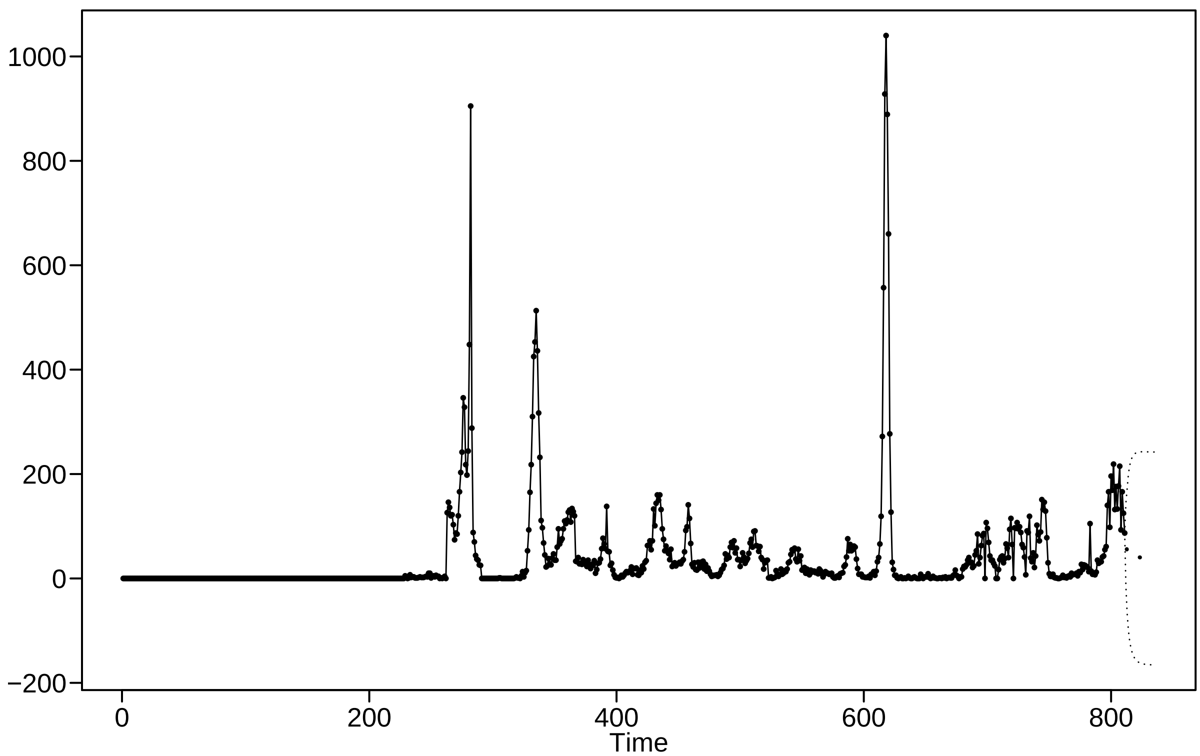


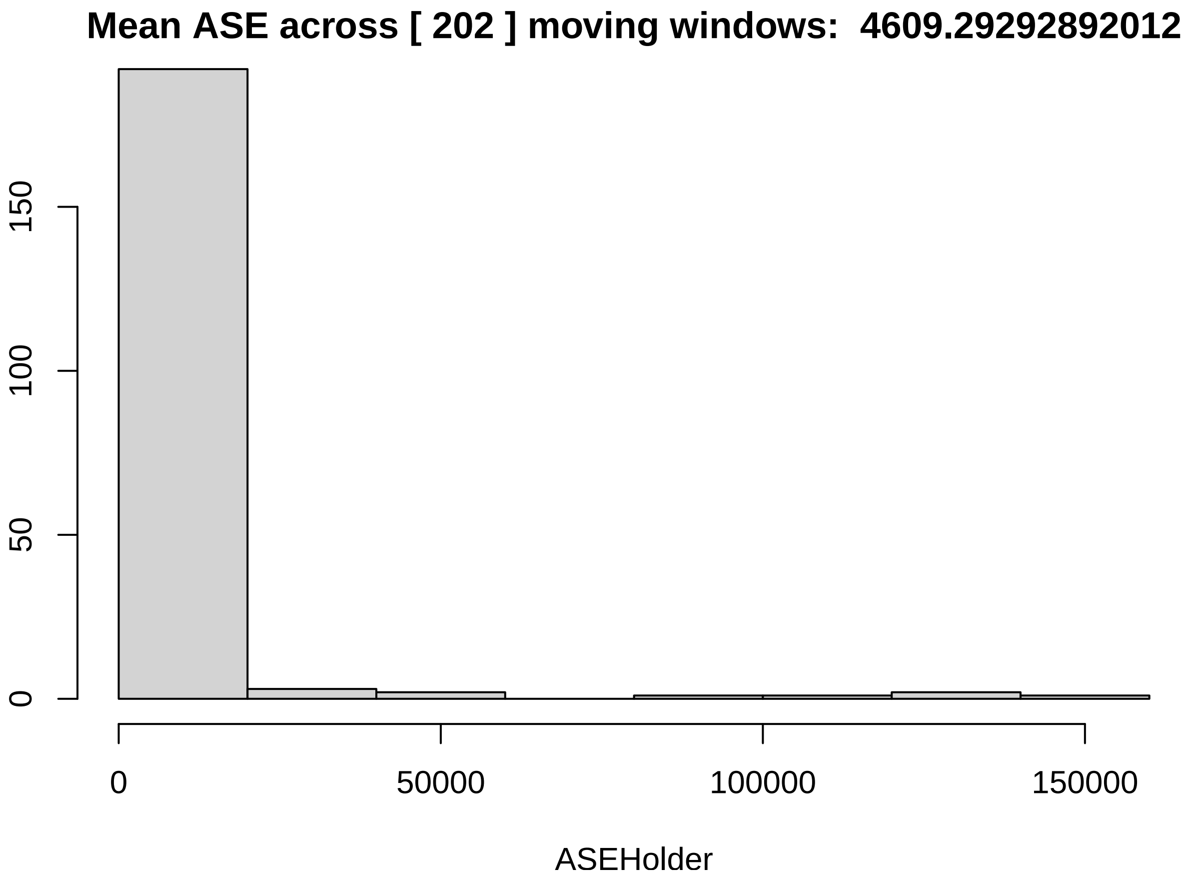

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



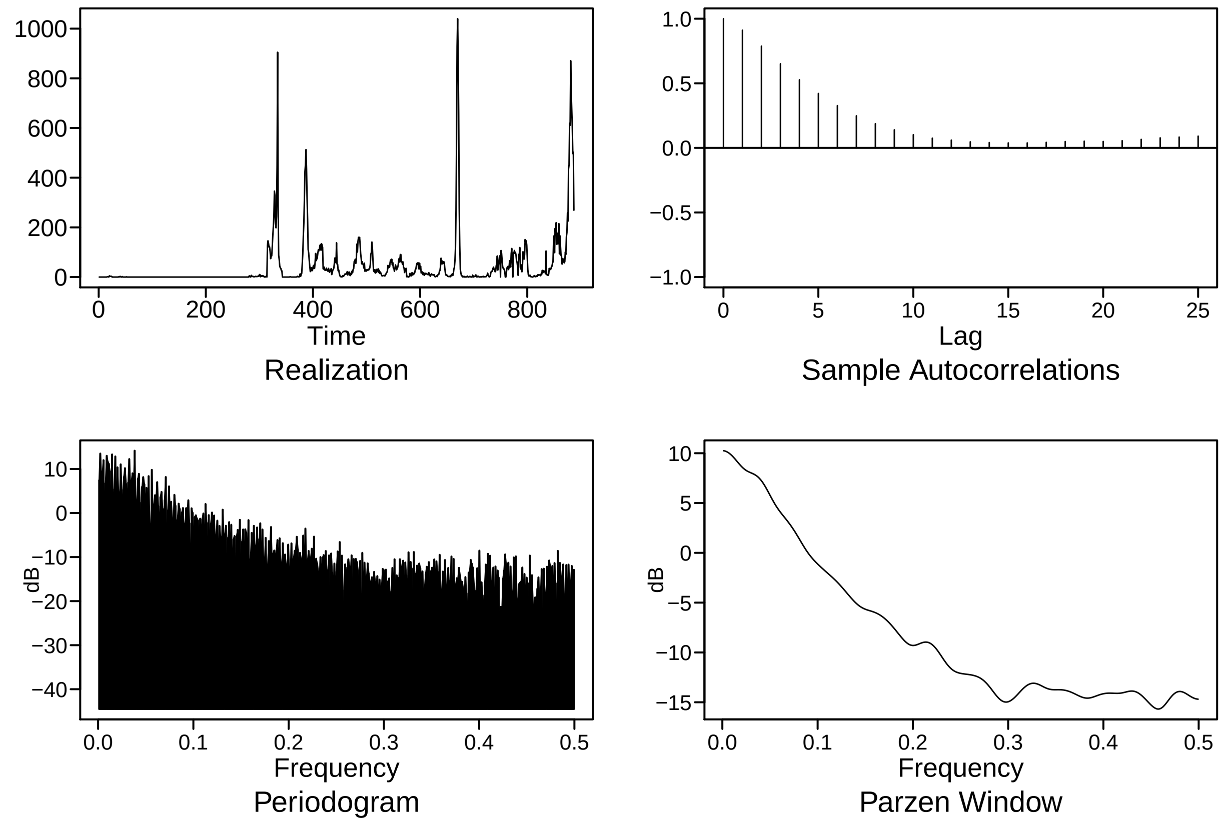



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8386
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8386B              1.1924               0.8386       0.0000
##
##
```

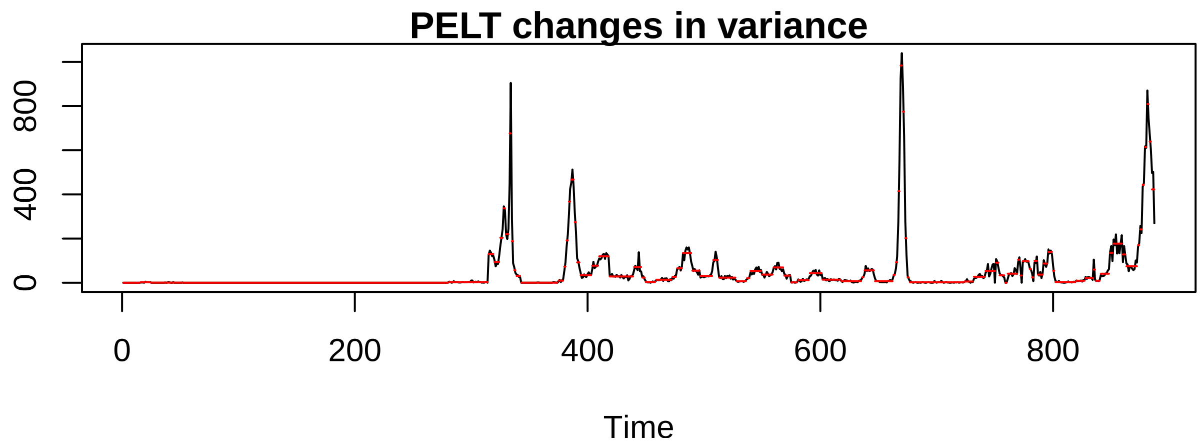

1000
800
600
400
200
0
−200
0
200
400
600
800
Time

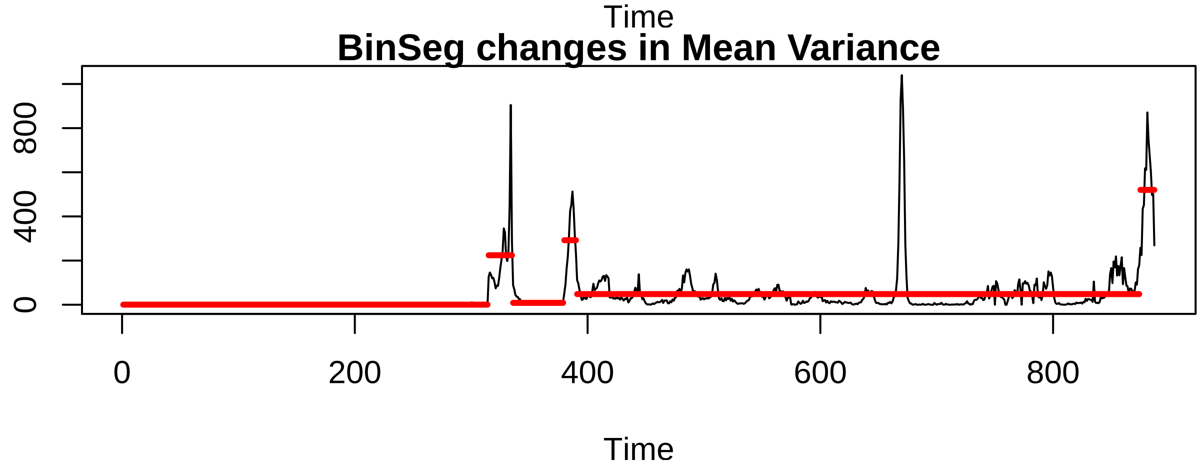



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

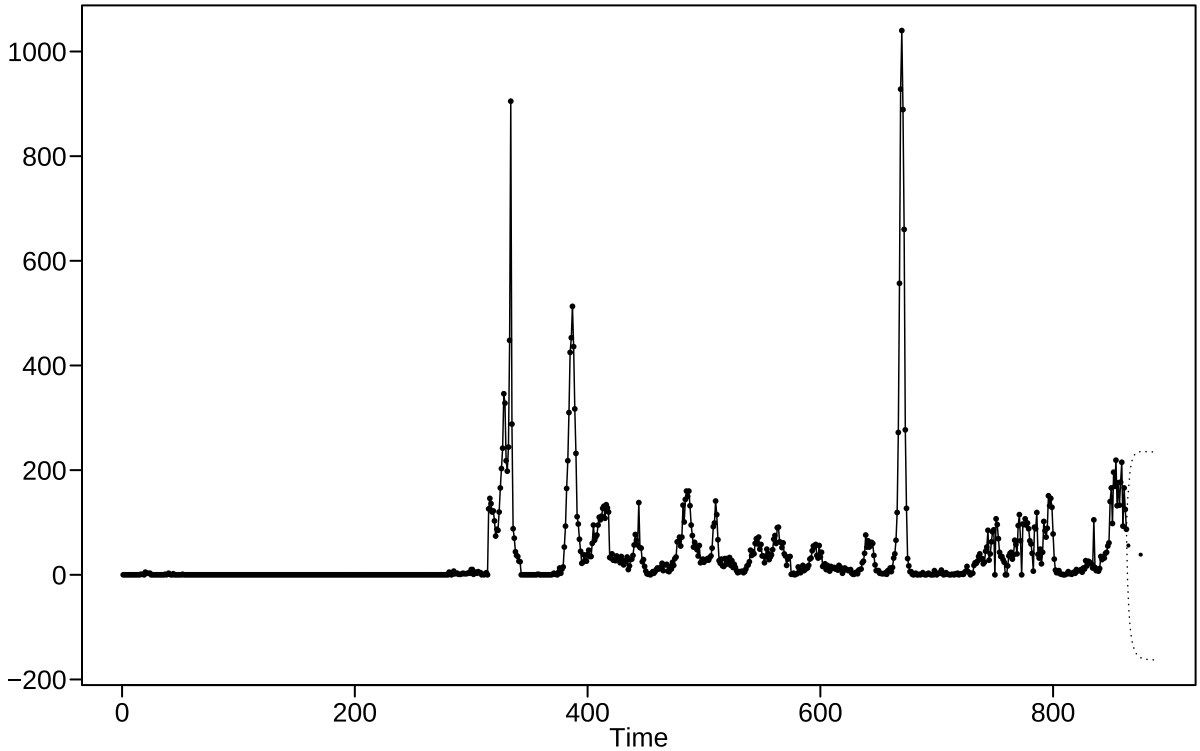


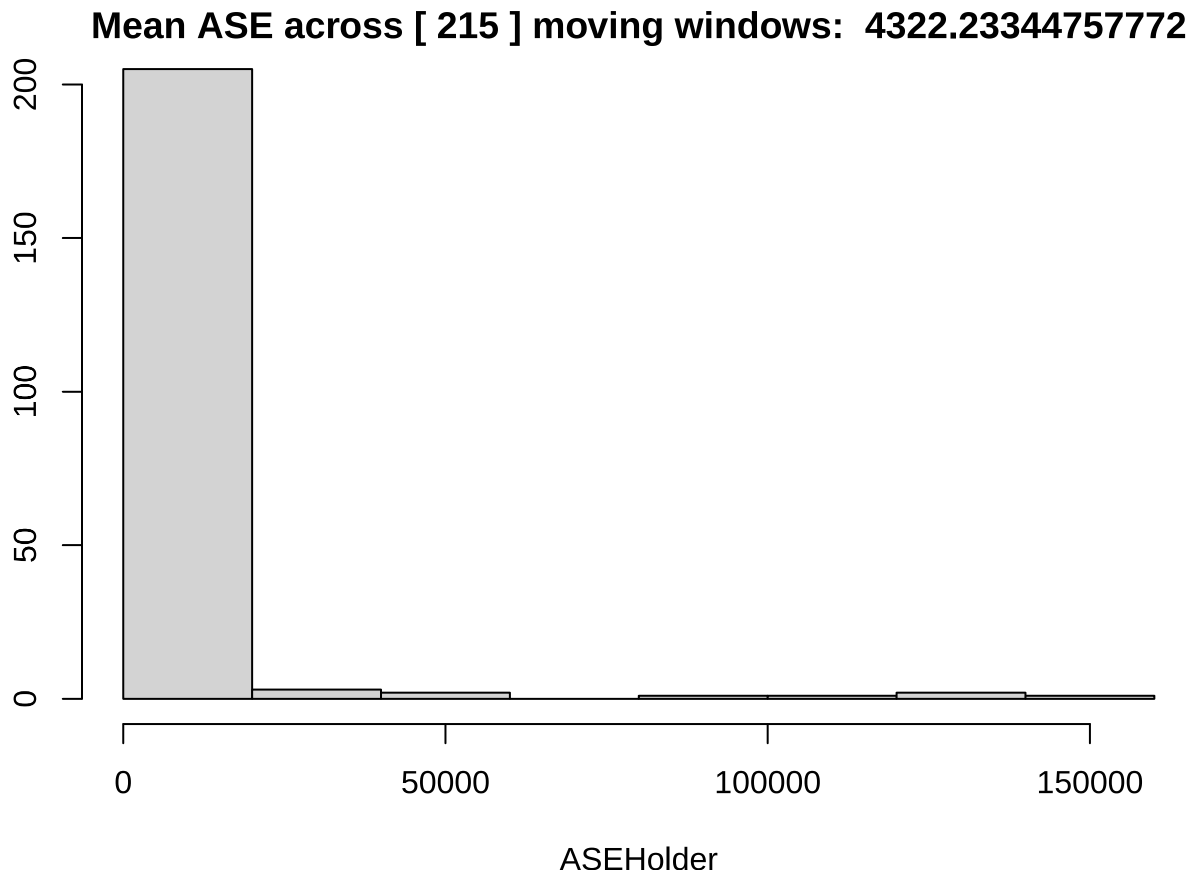

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



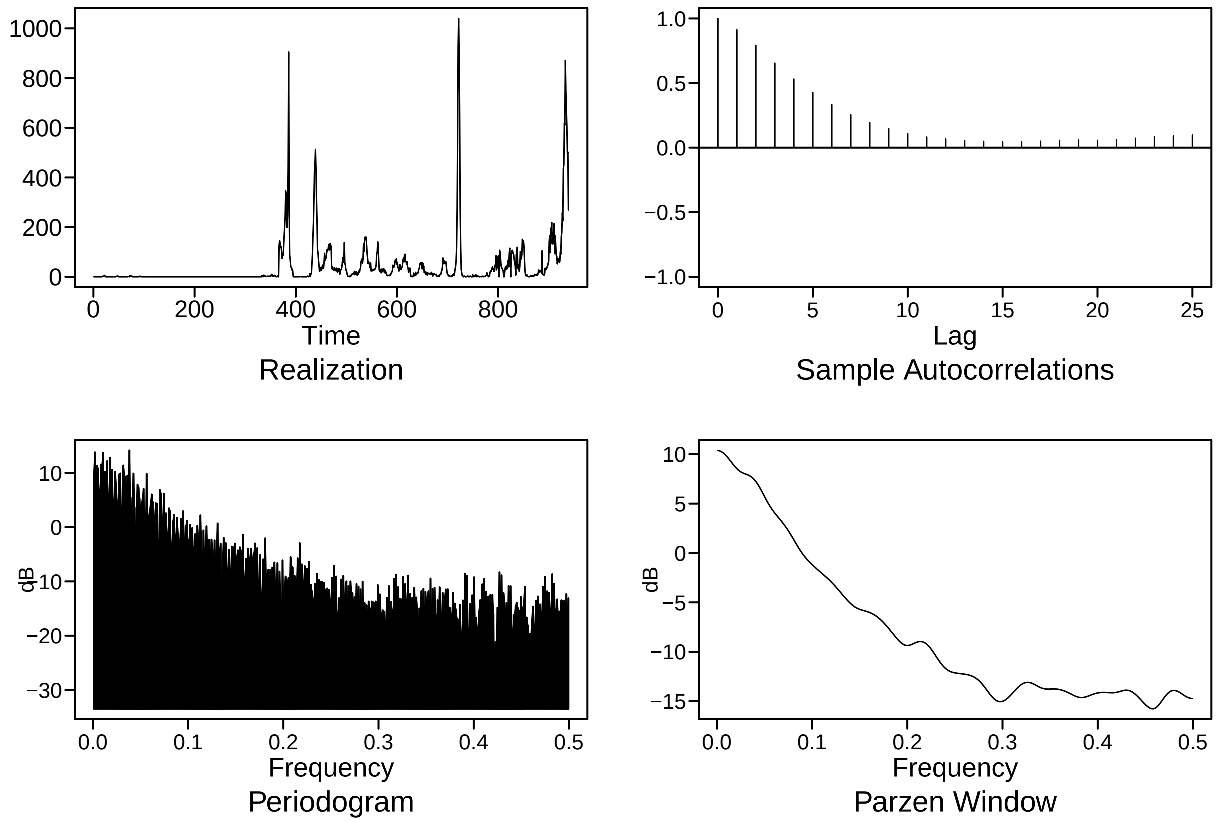



```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8404
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8404B              1.1899               0.8404       0.0000
##
##
```

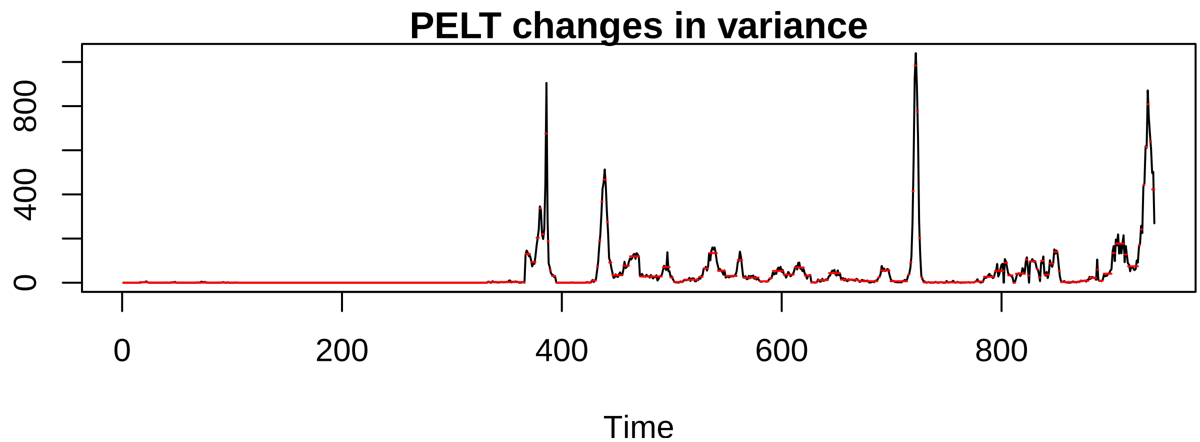

1000
800
600
400
200
0
−200
0
200
400
600
800
Time

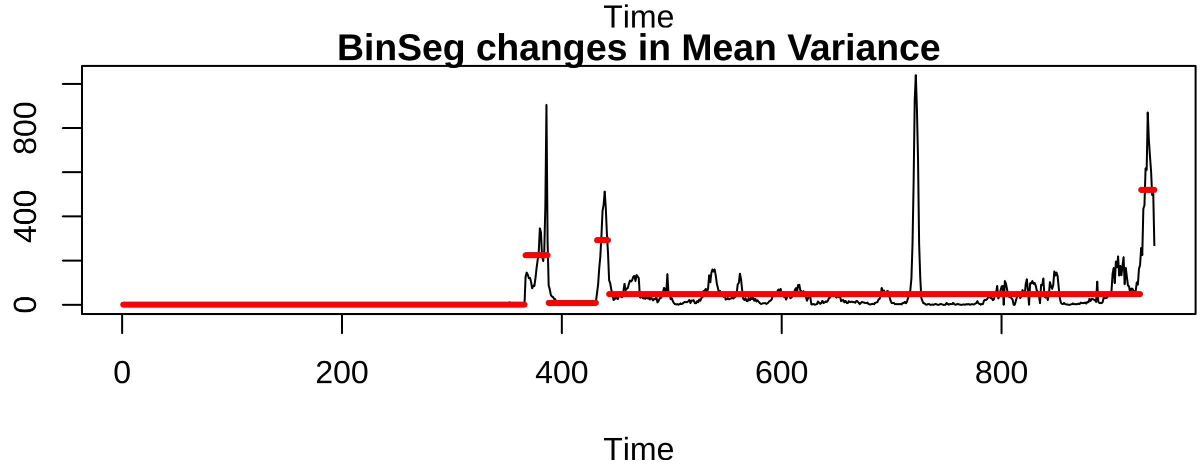



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

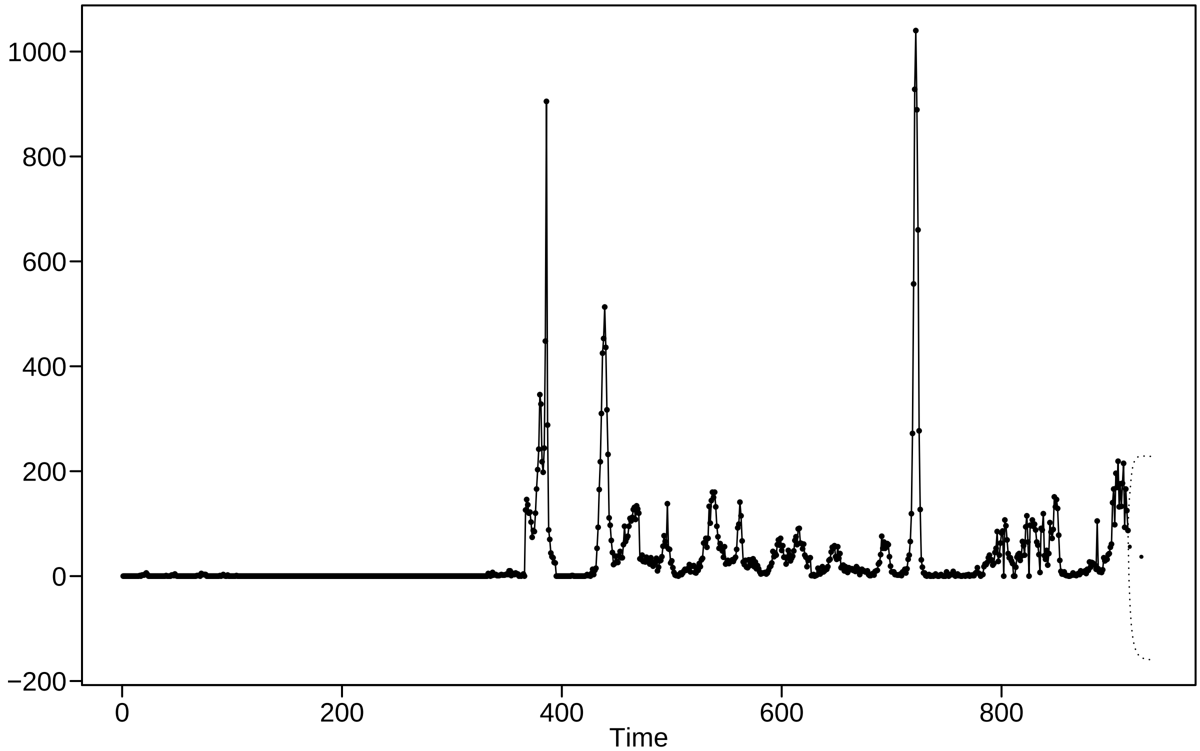



```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

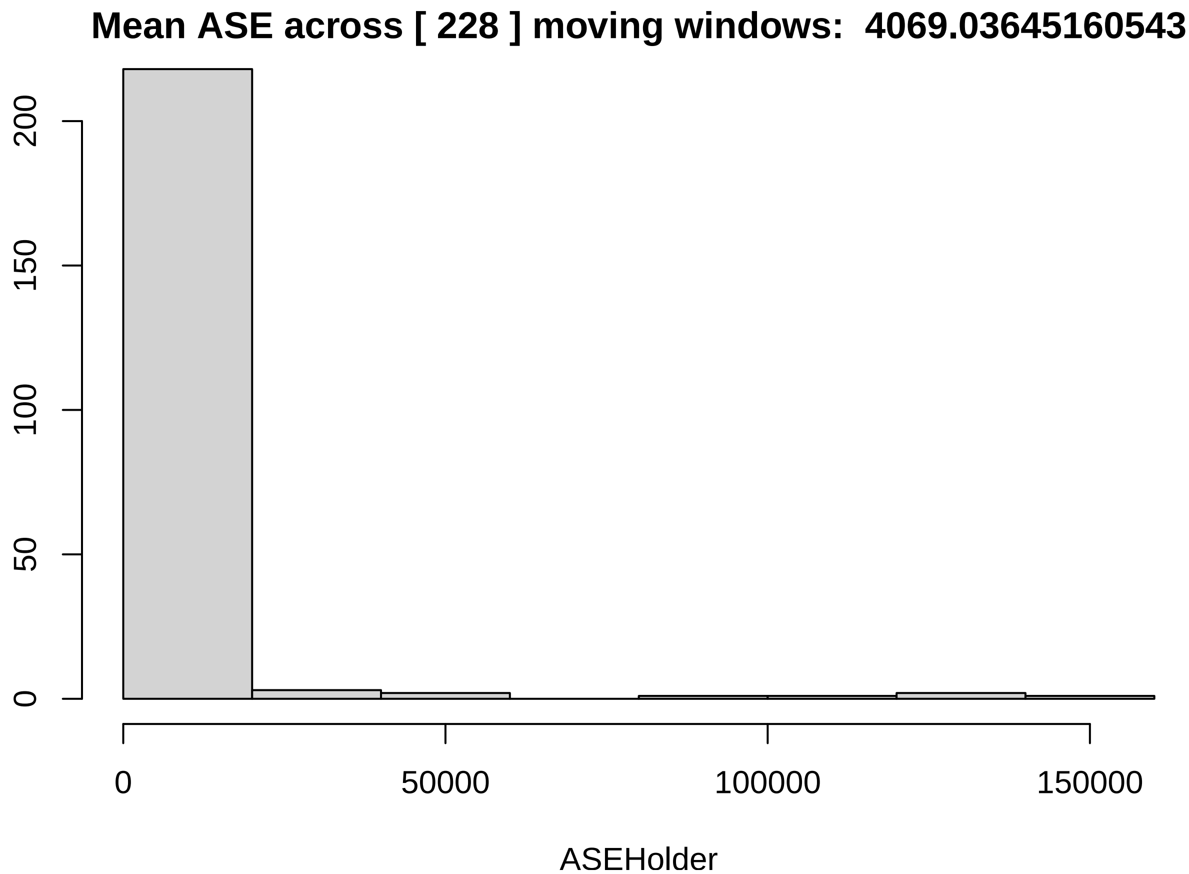




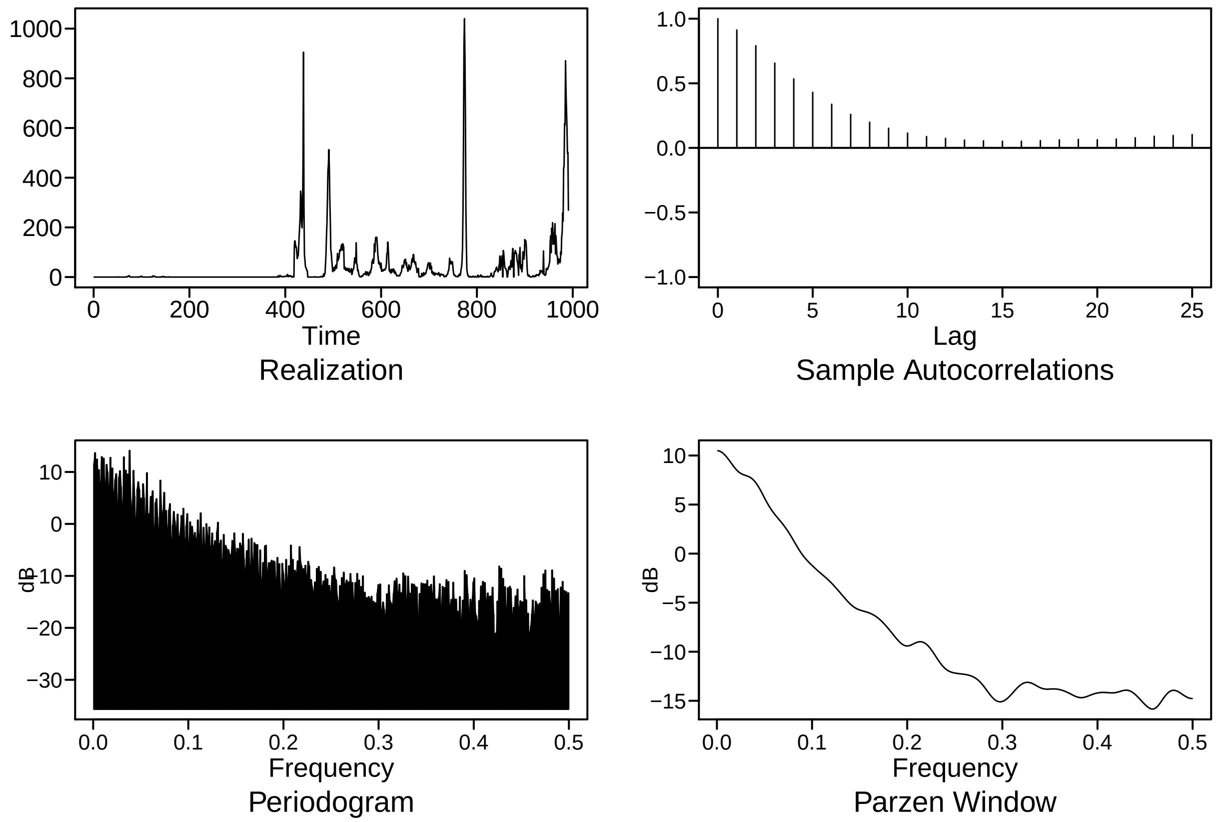

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8419
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8419B              1.1878               0.8419       0.0000
##
##
```



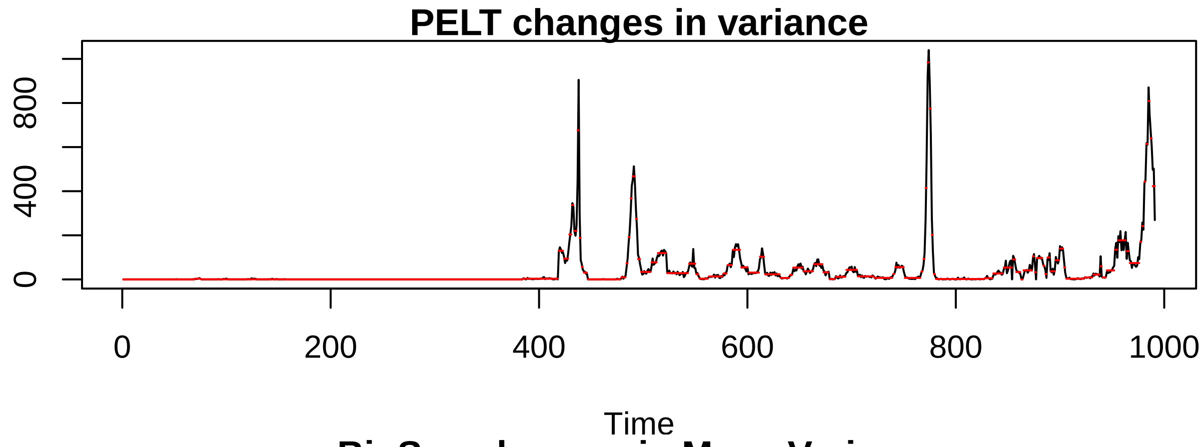
1000
800
600
400
200
0
−200
0
200
400
600
800
Time

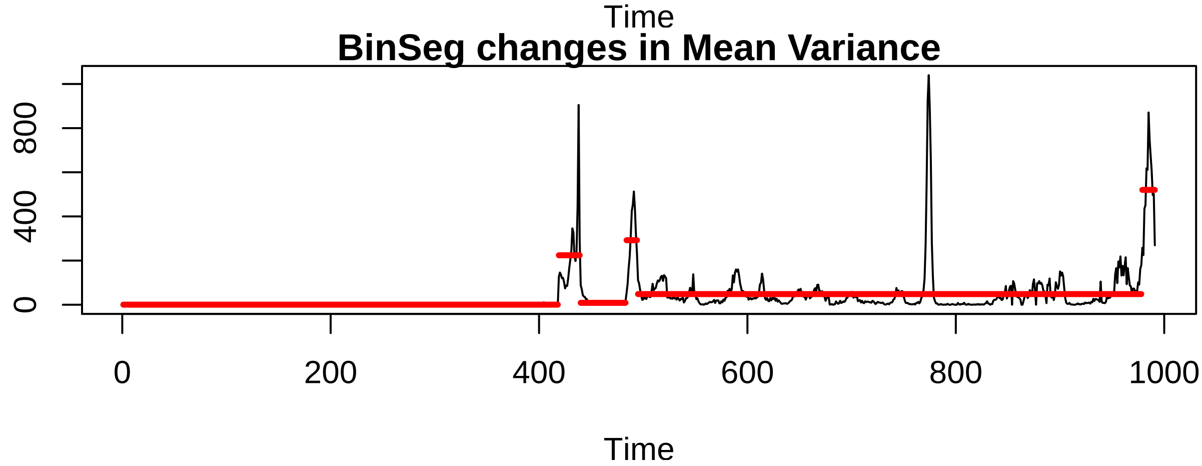



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

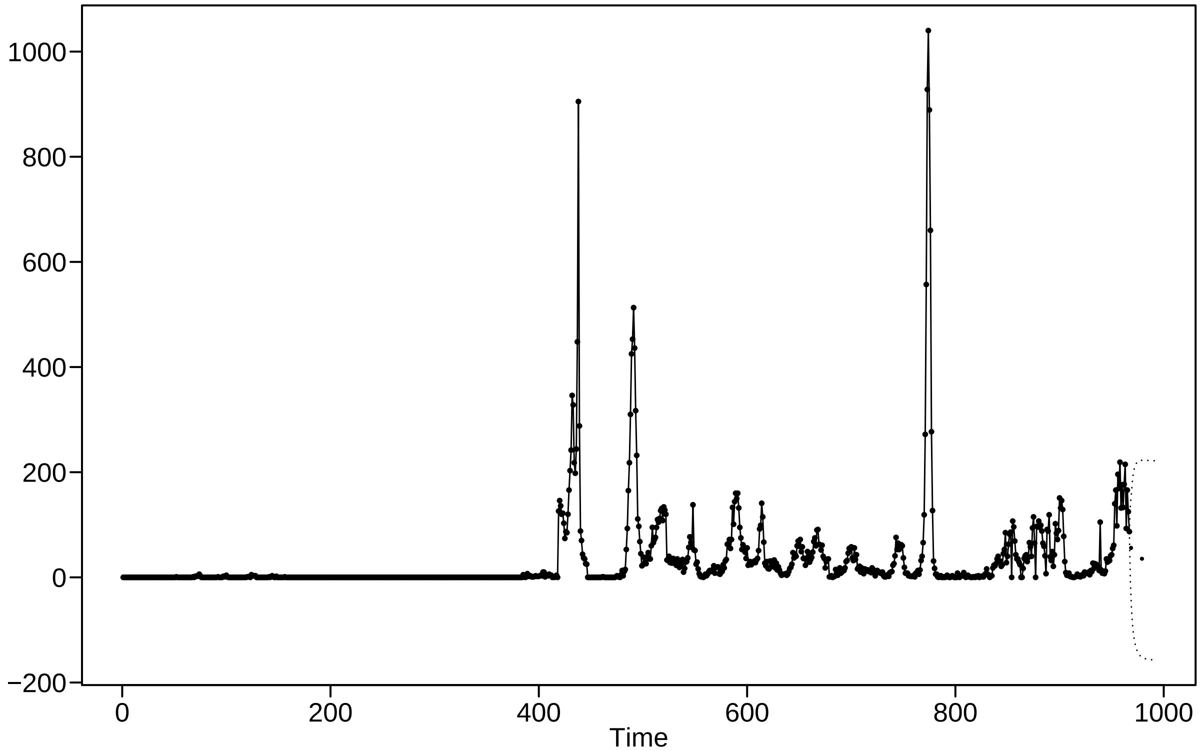



```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

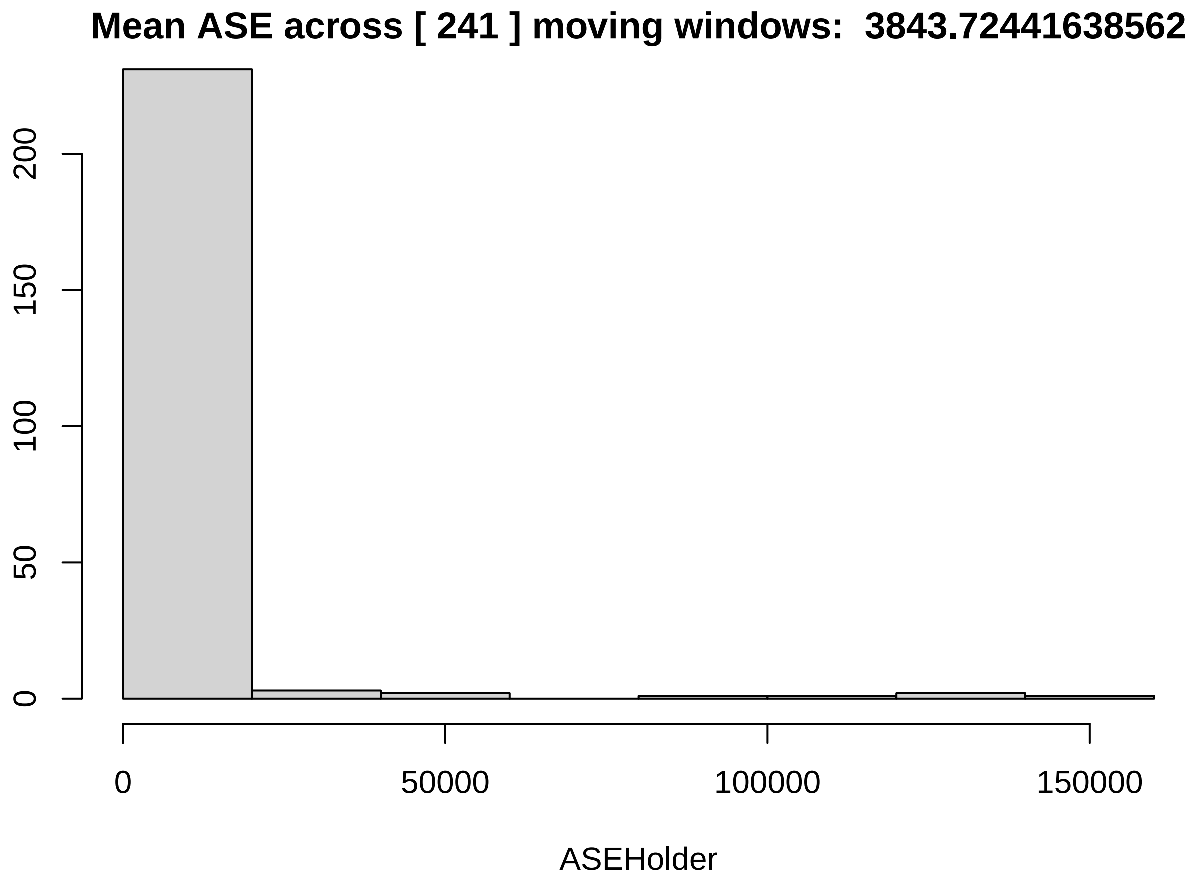




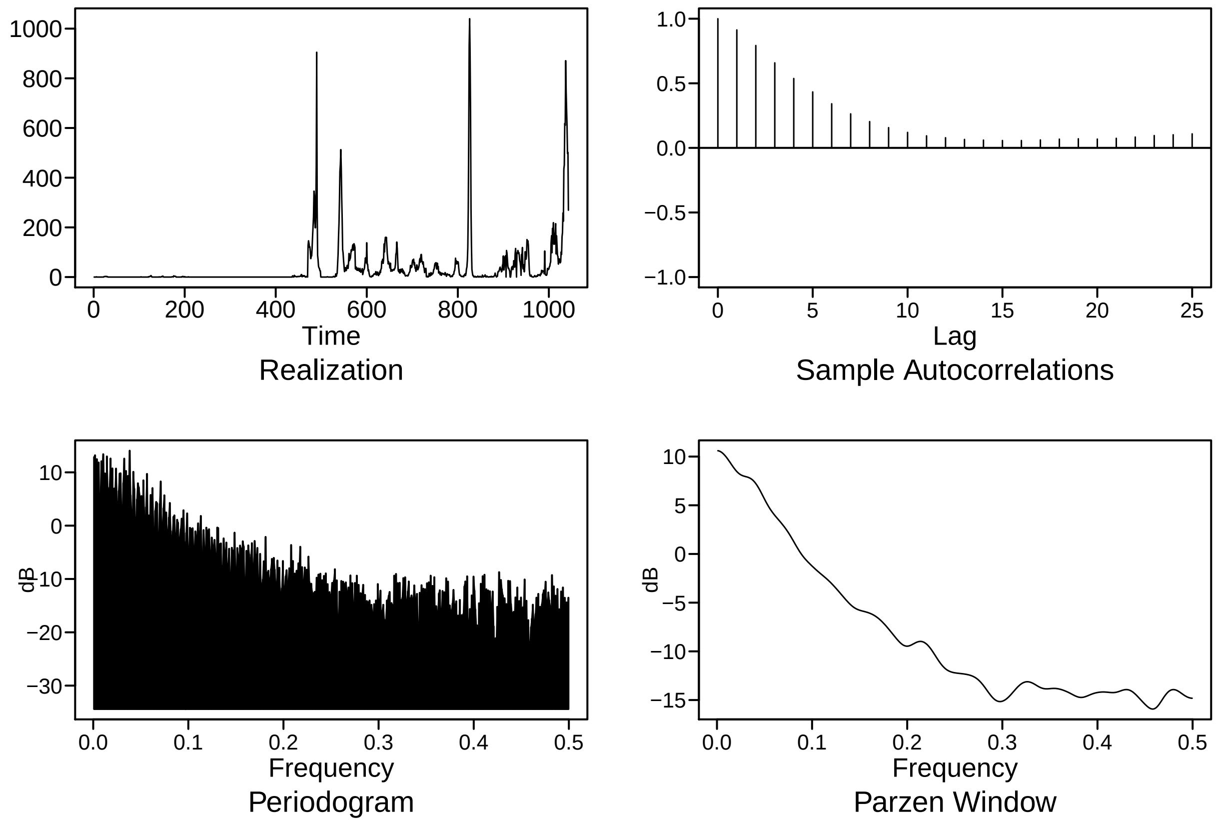

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8433
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8433B              1.1858               0.8433       0.0000
##
##
```



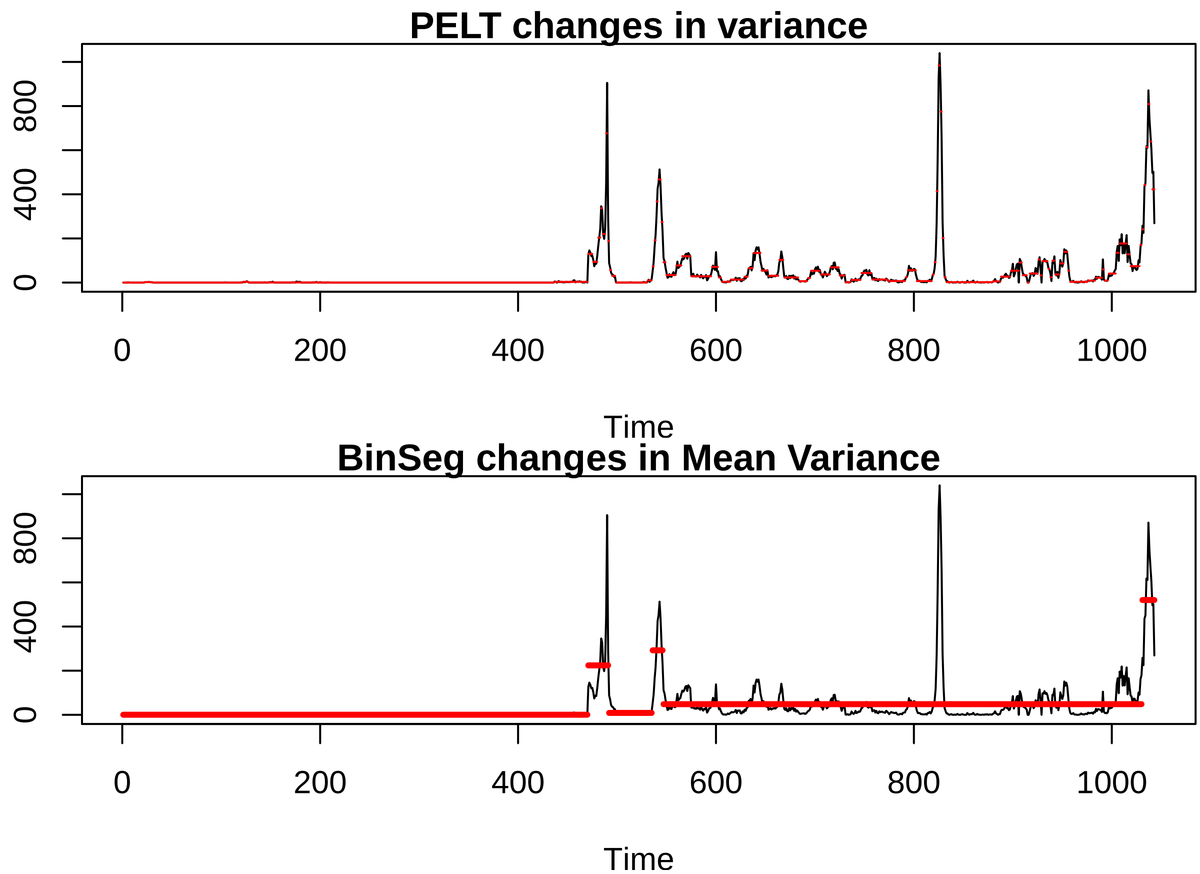
1000
800
600
400
200
0
−200
0
200
400
600
800
1000
Time



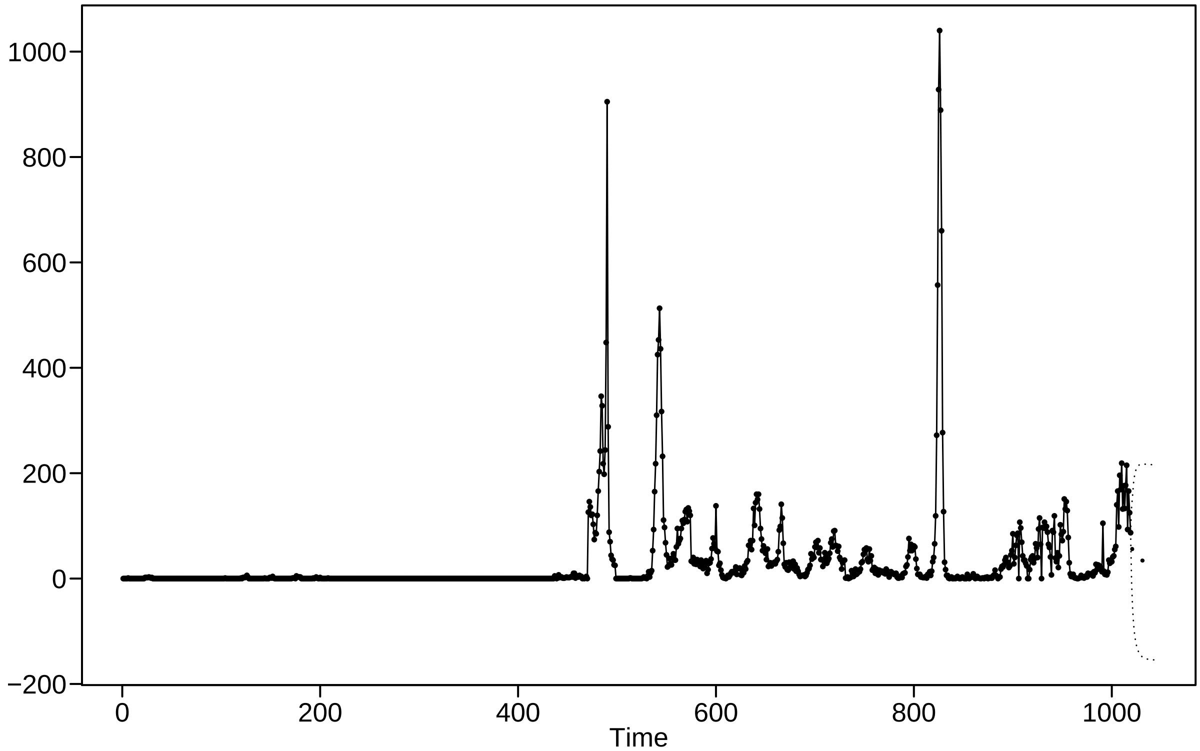

```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```



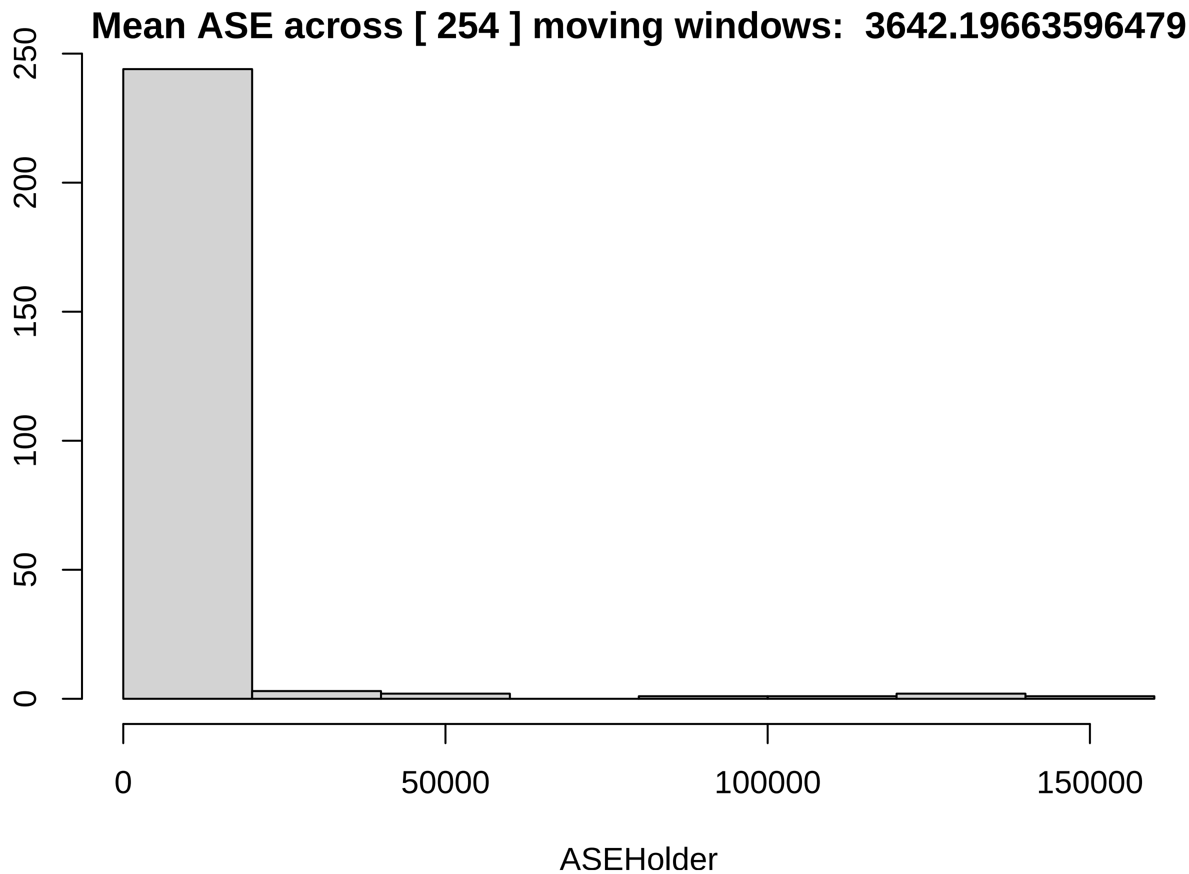

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```



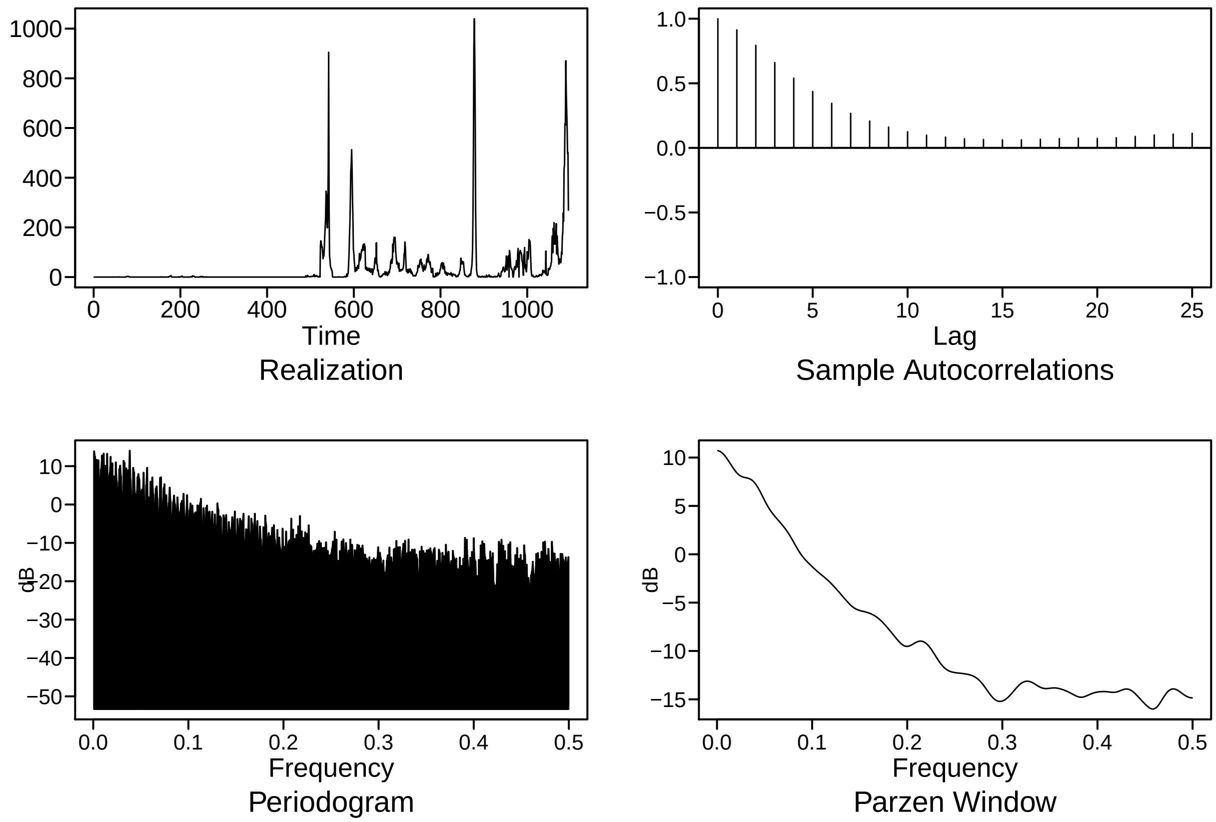

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8445
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8445B              1.1841               0.8445       0.0000
##
##
```



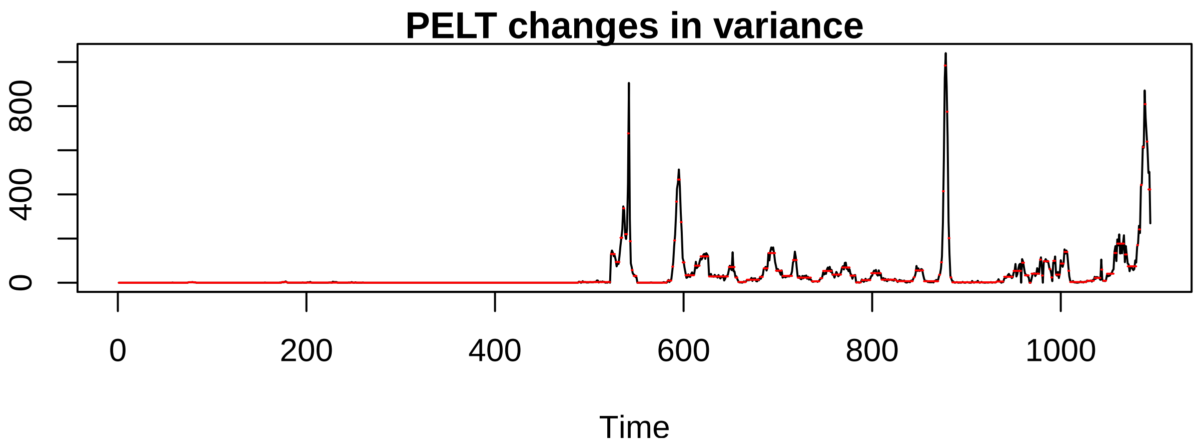
1000
800
600
400
200
0
−200
0
200
400
600
800
1000
Time

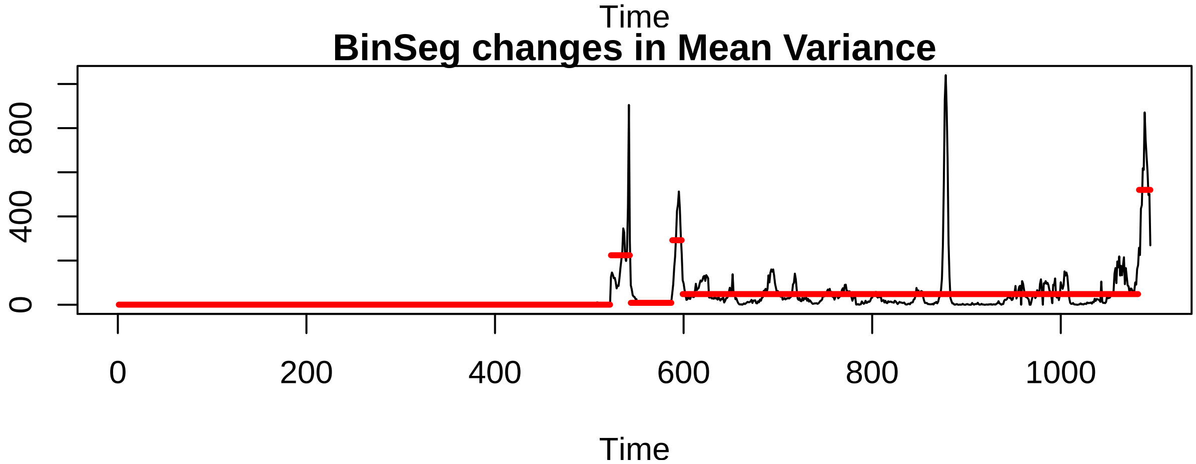



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

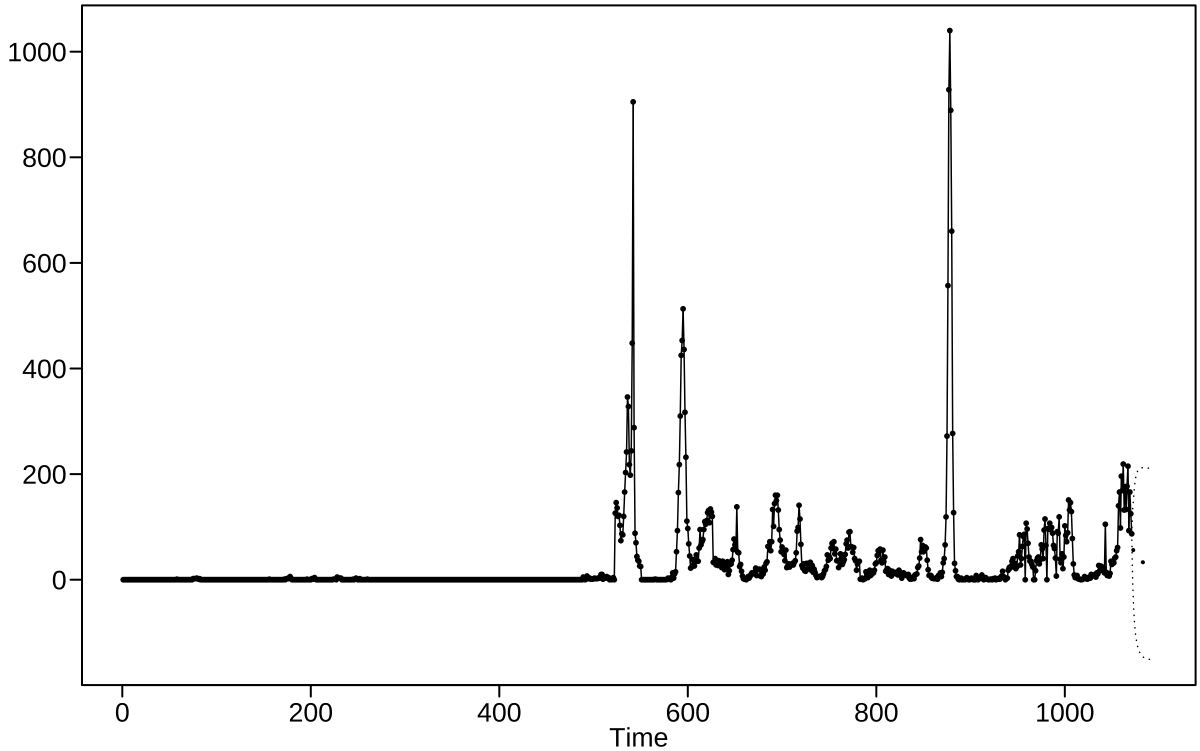



```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

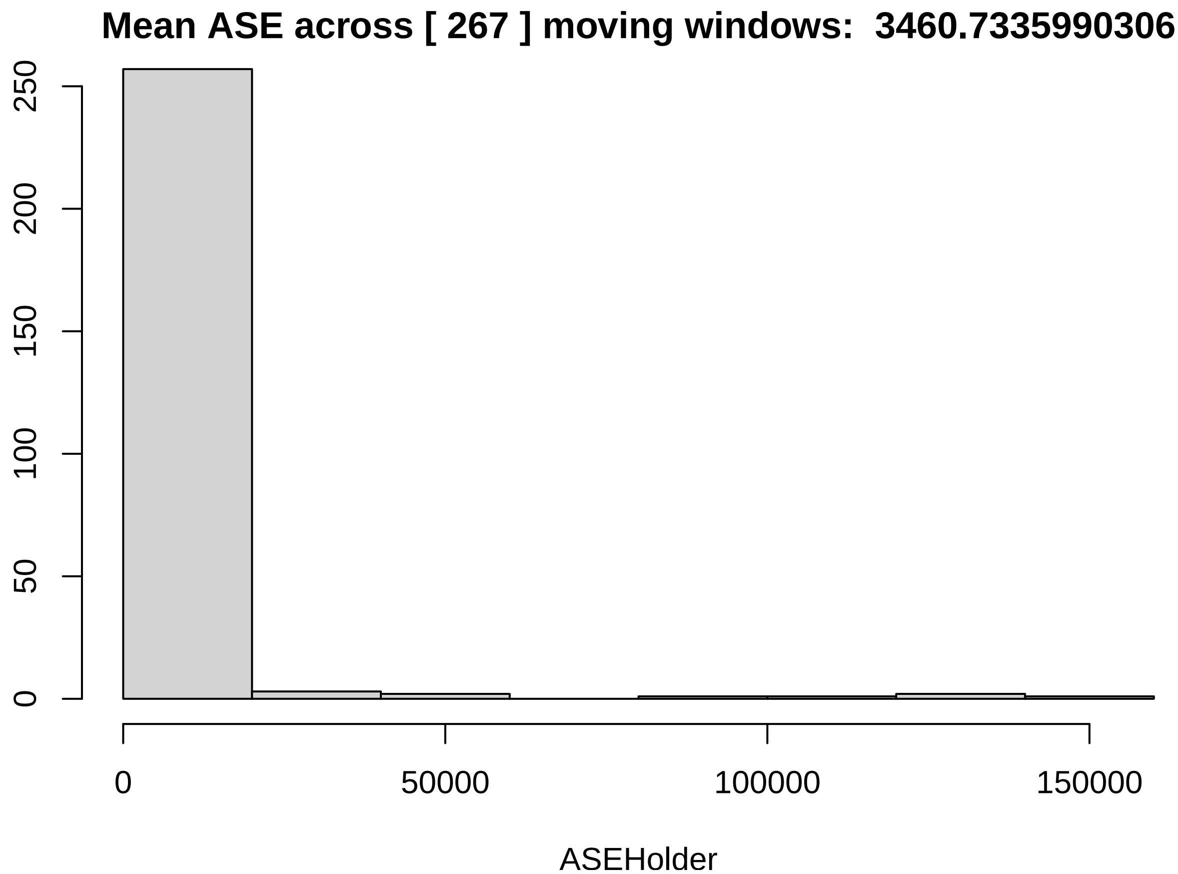




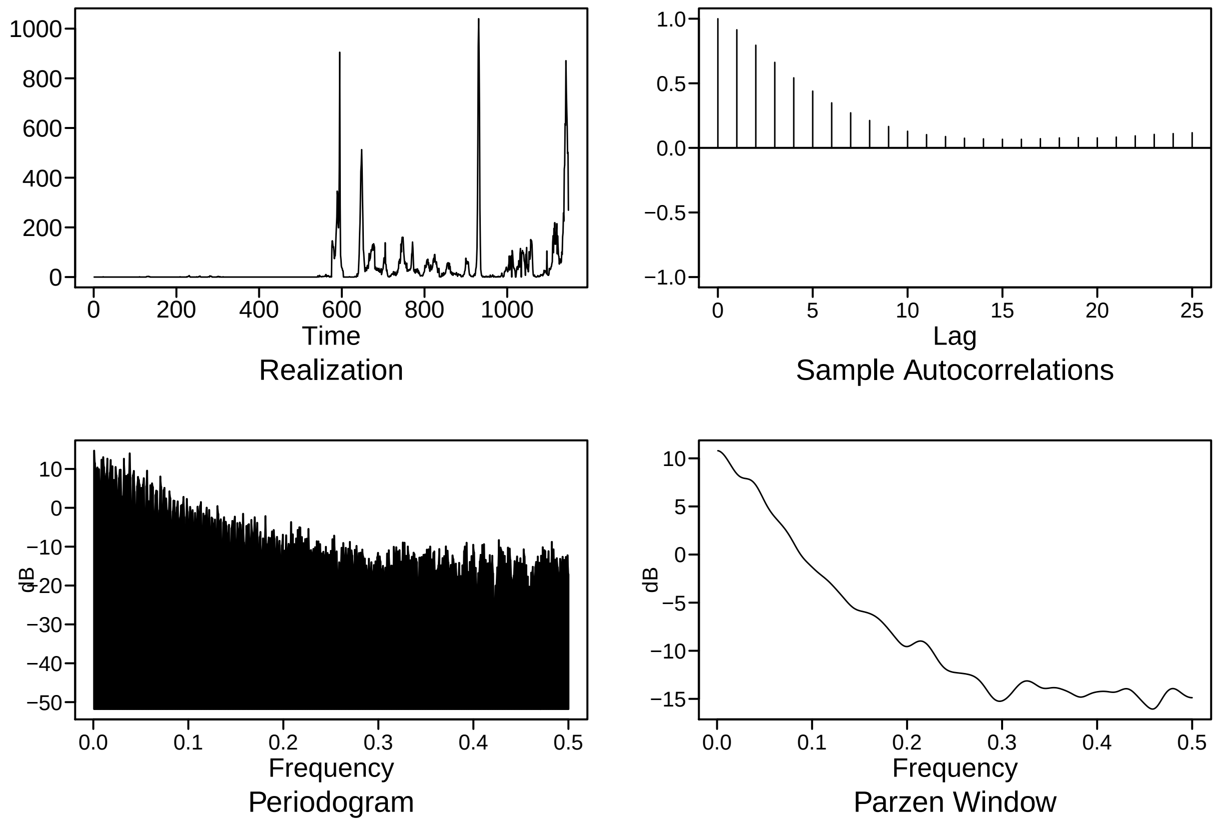

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8456
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8456B              1.1825               0.8456       0.0000
##
##
```



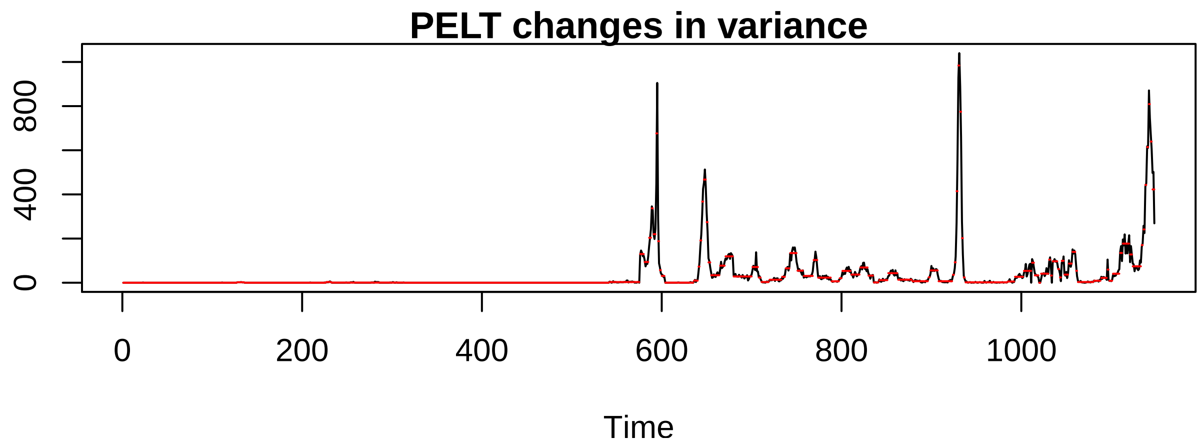
1000
800
600
400
200
0
0
200
400
600
800
1000
Time

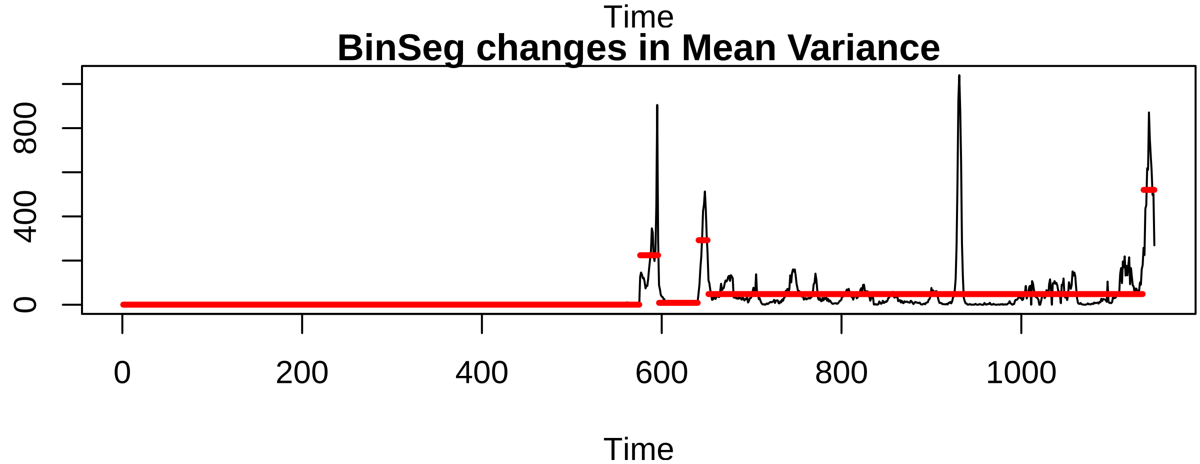



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

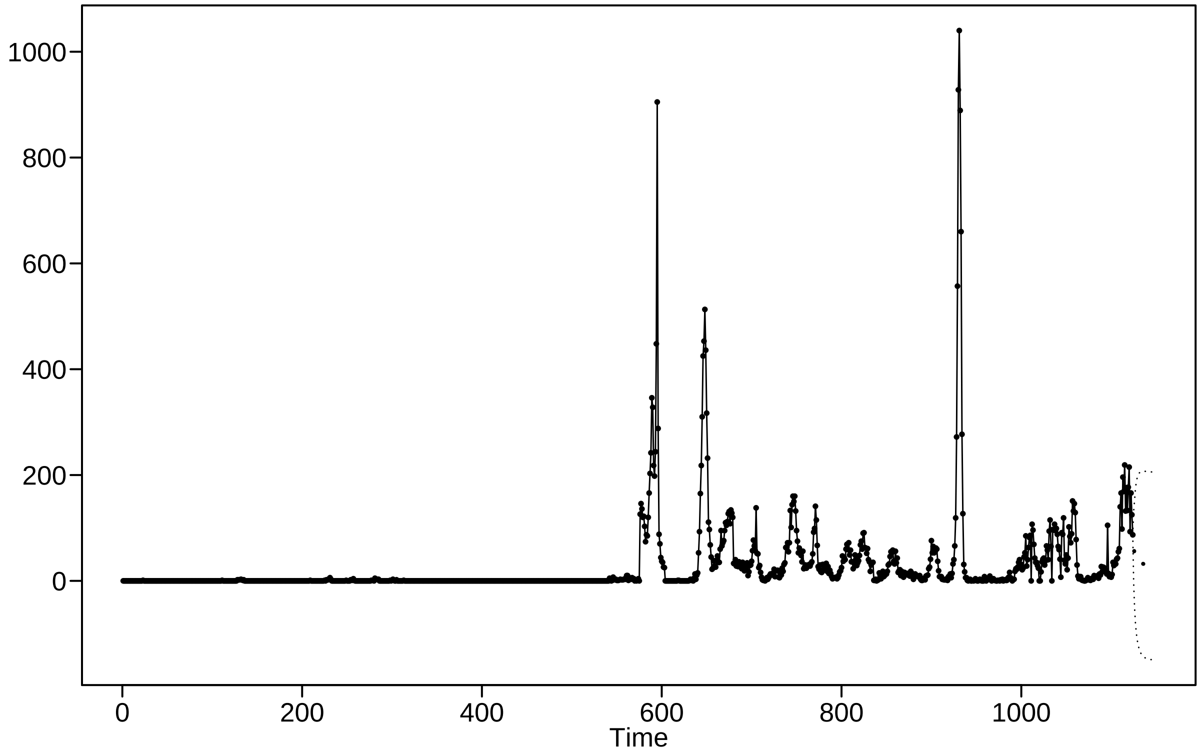



```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

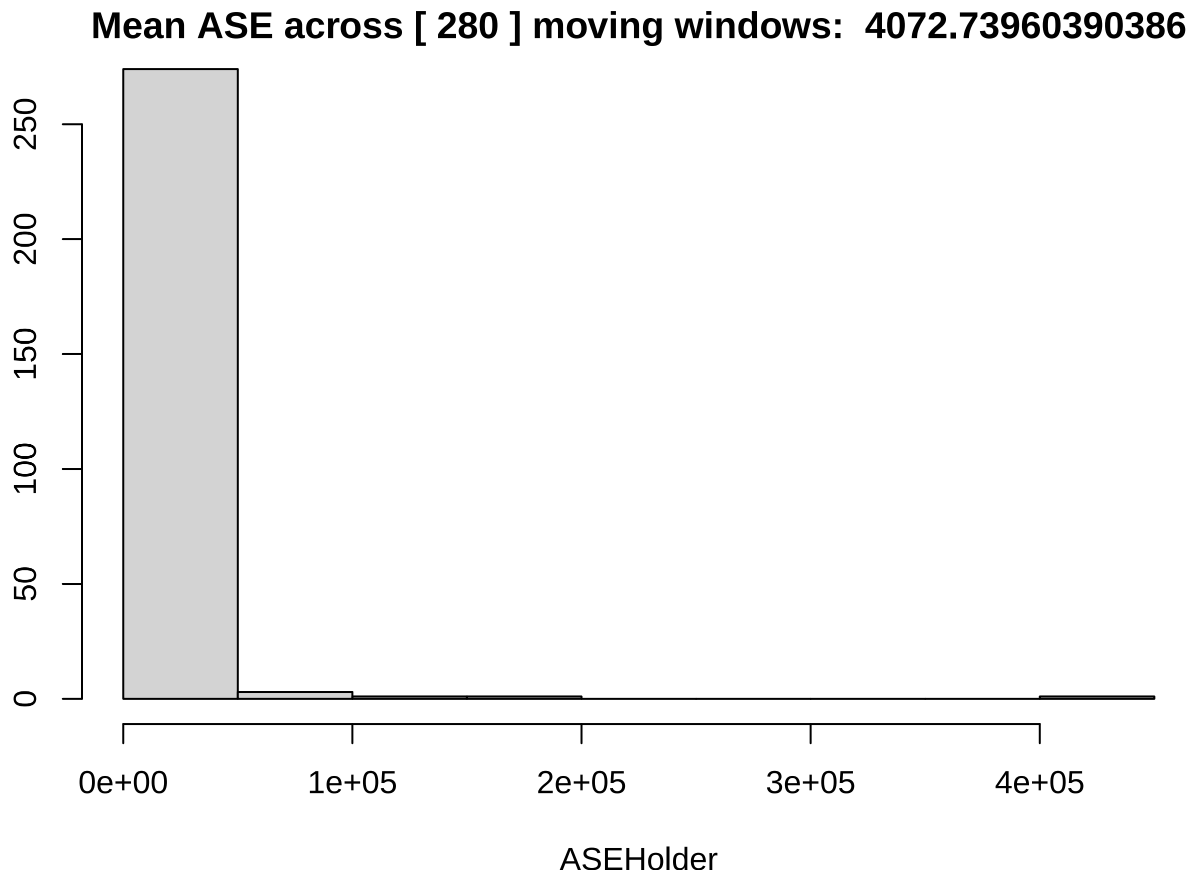




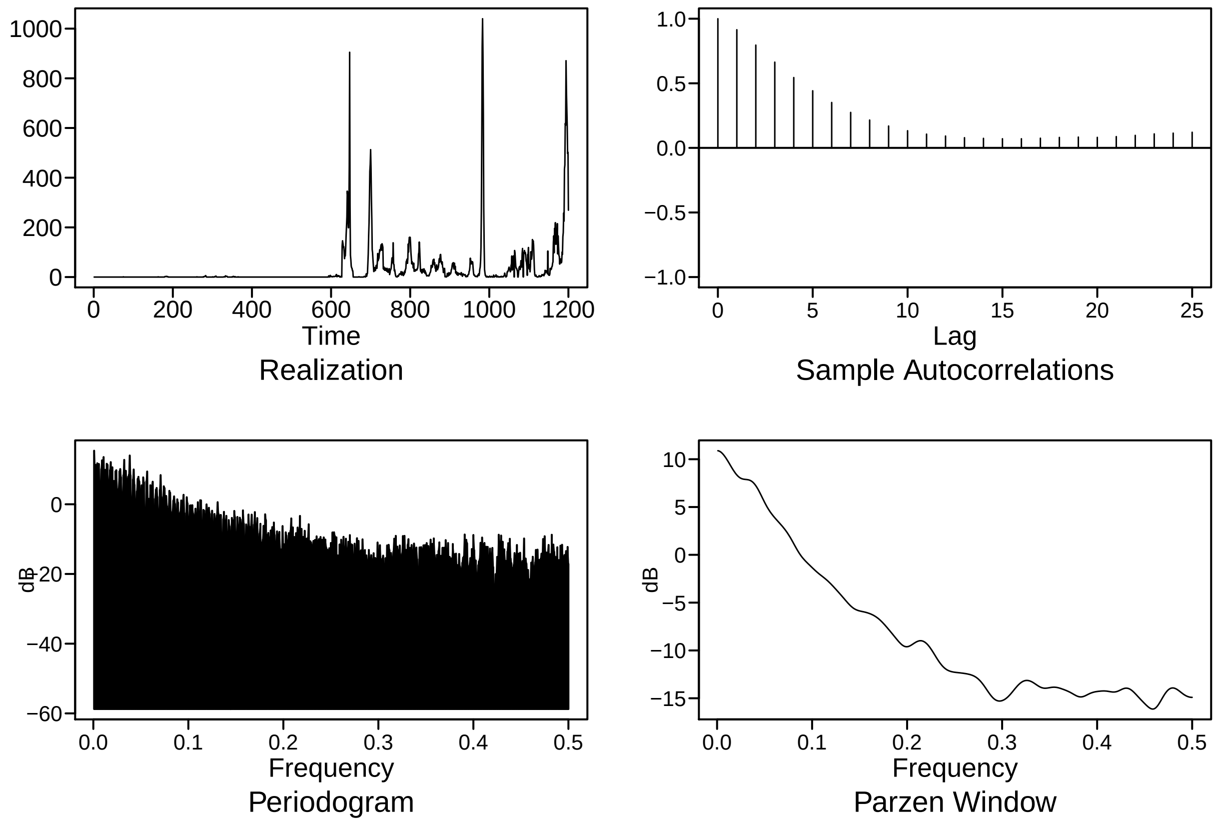

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8466
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8466B              1.1812               0.8466       0.0000
##
##
```



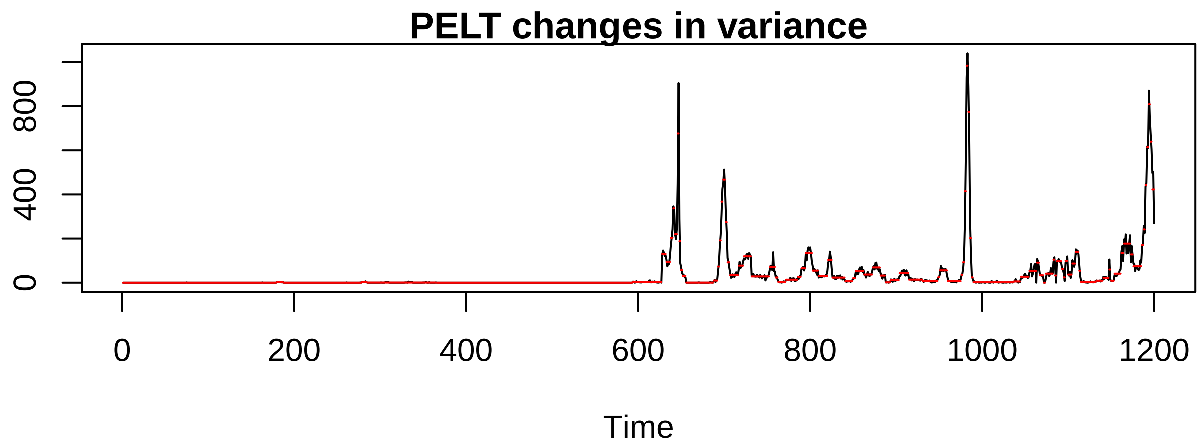
1000
800
600
400
200
0
0
200
400
600
800
1000
Time

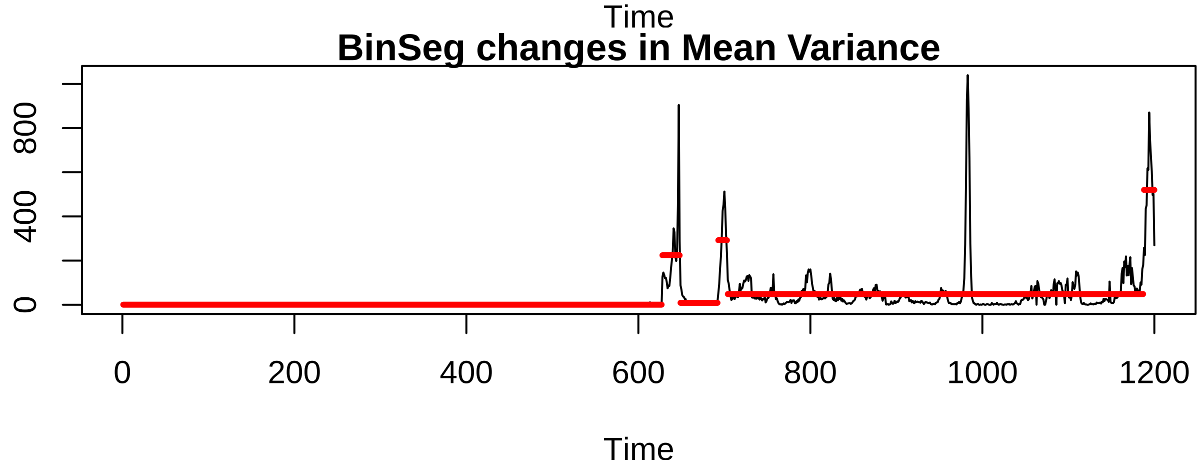



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

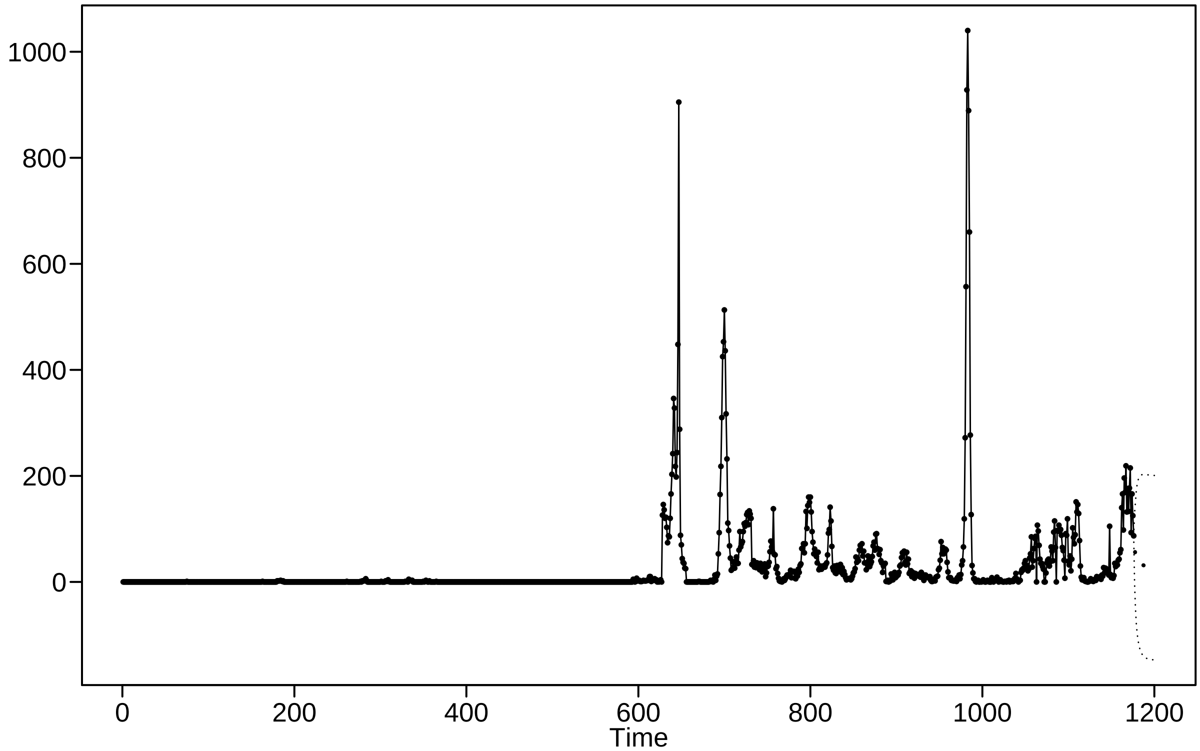



```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

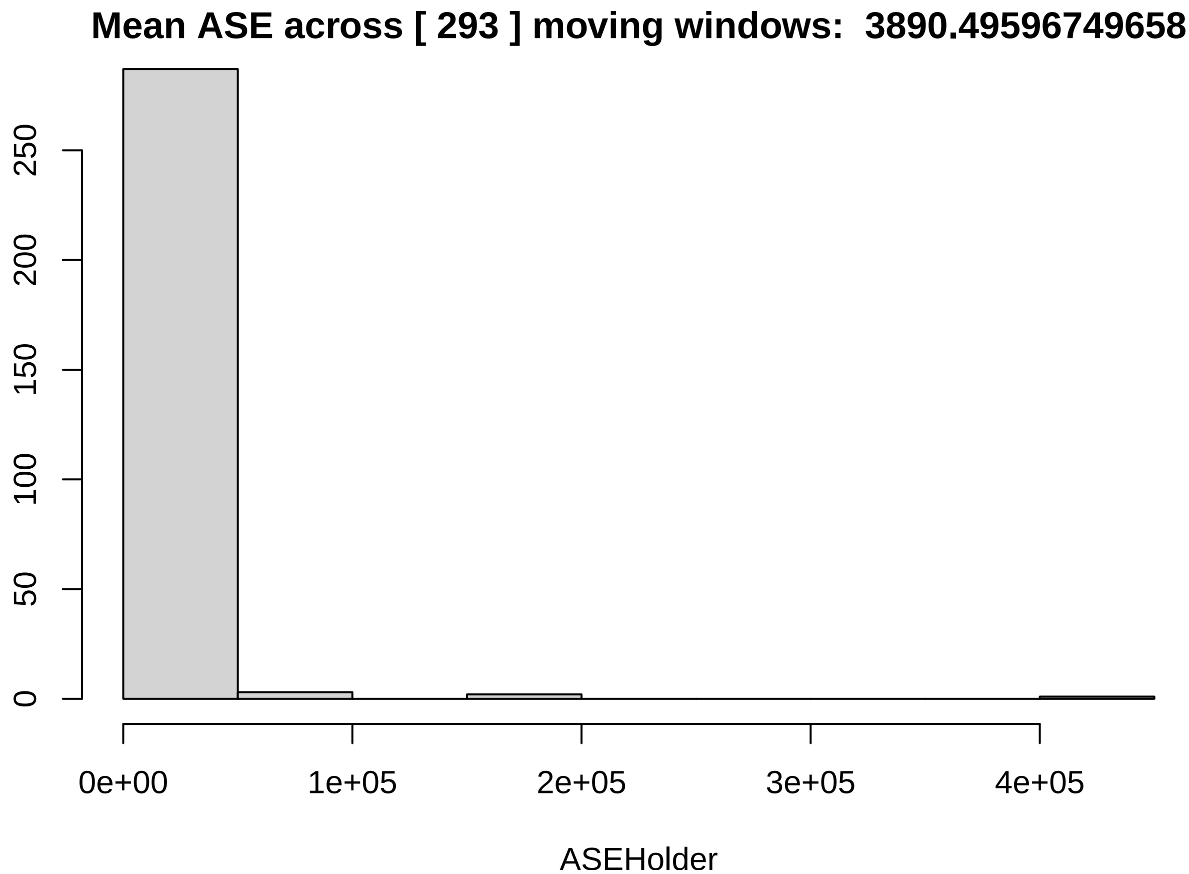




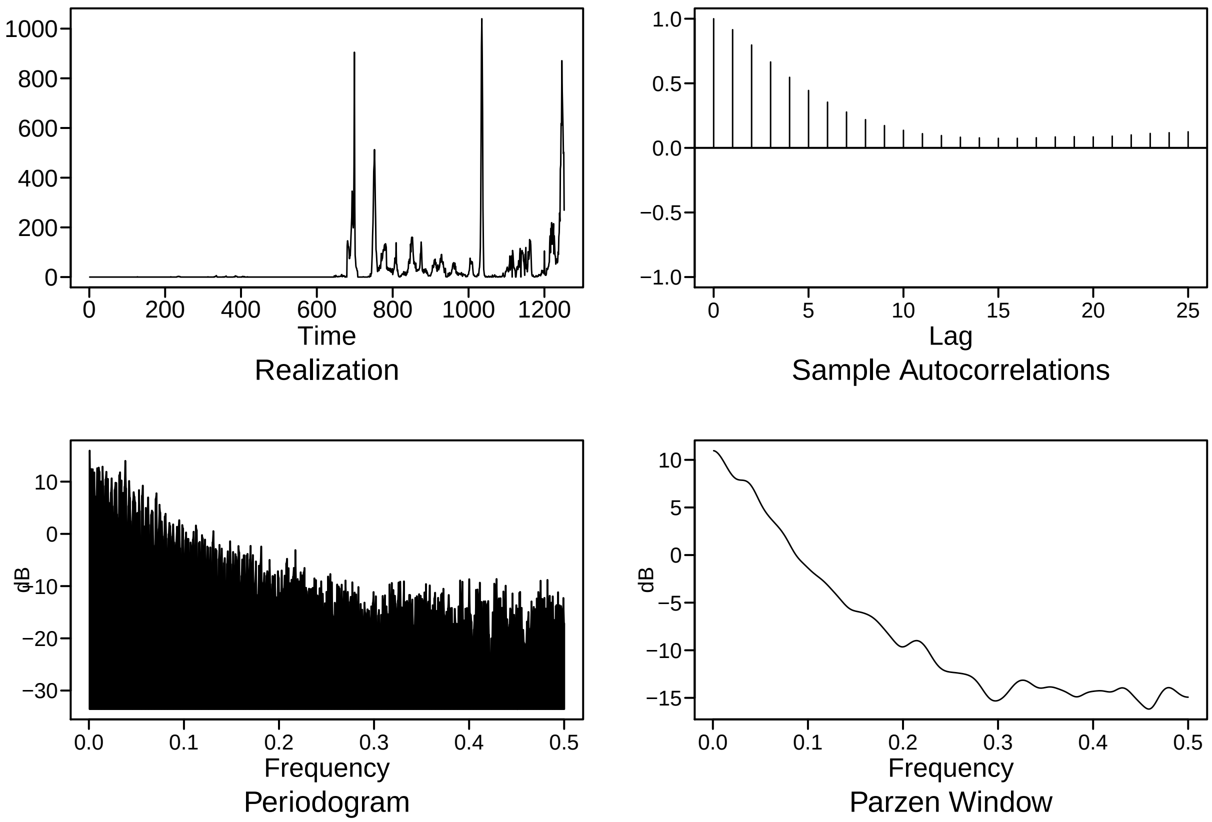

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8475
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8475B              1.1799               0.8475       0.0000
##
##
```



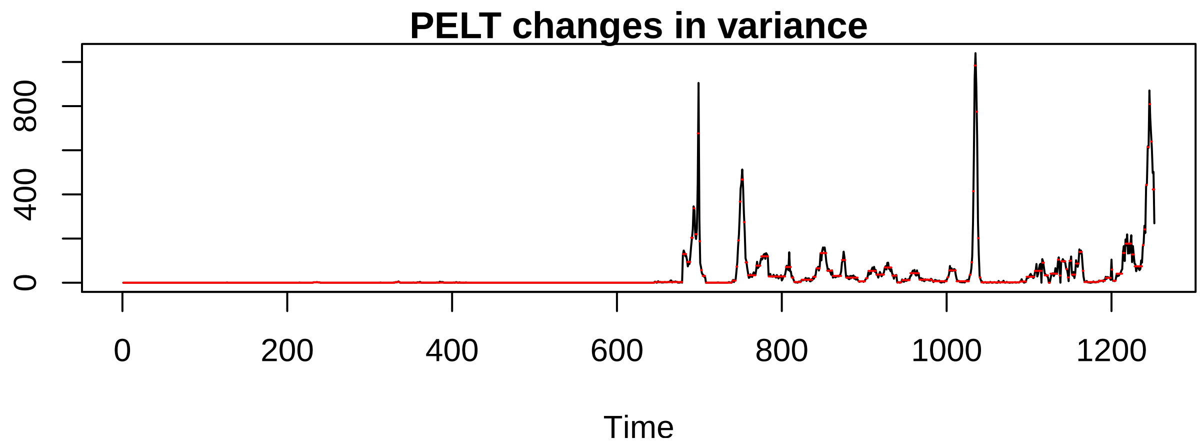
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
Time

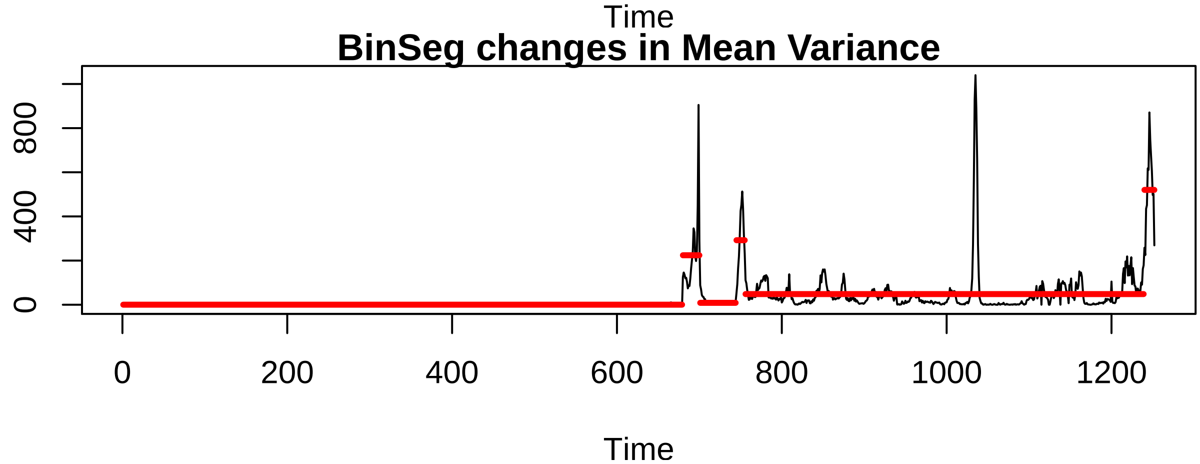



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

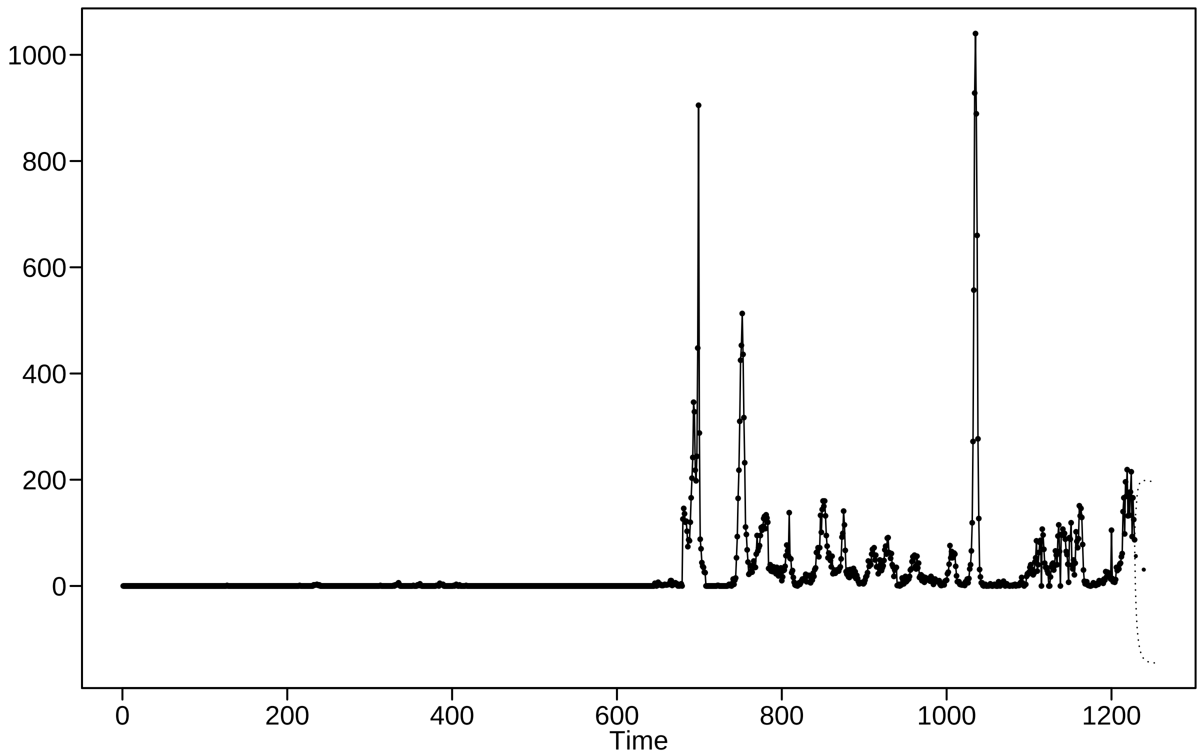



```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

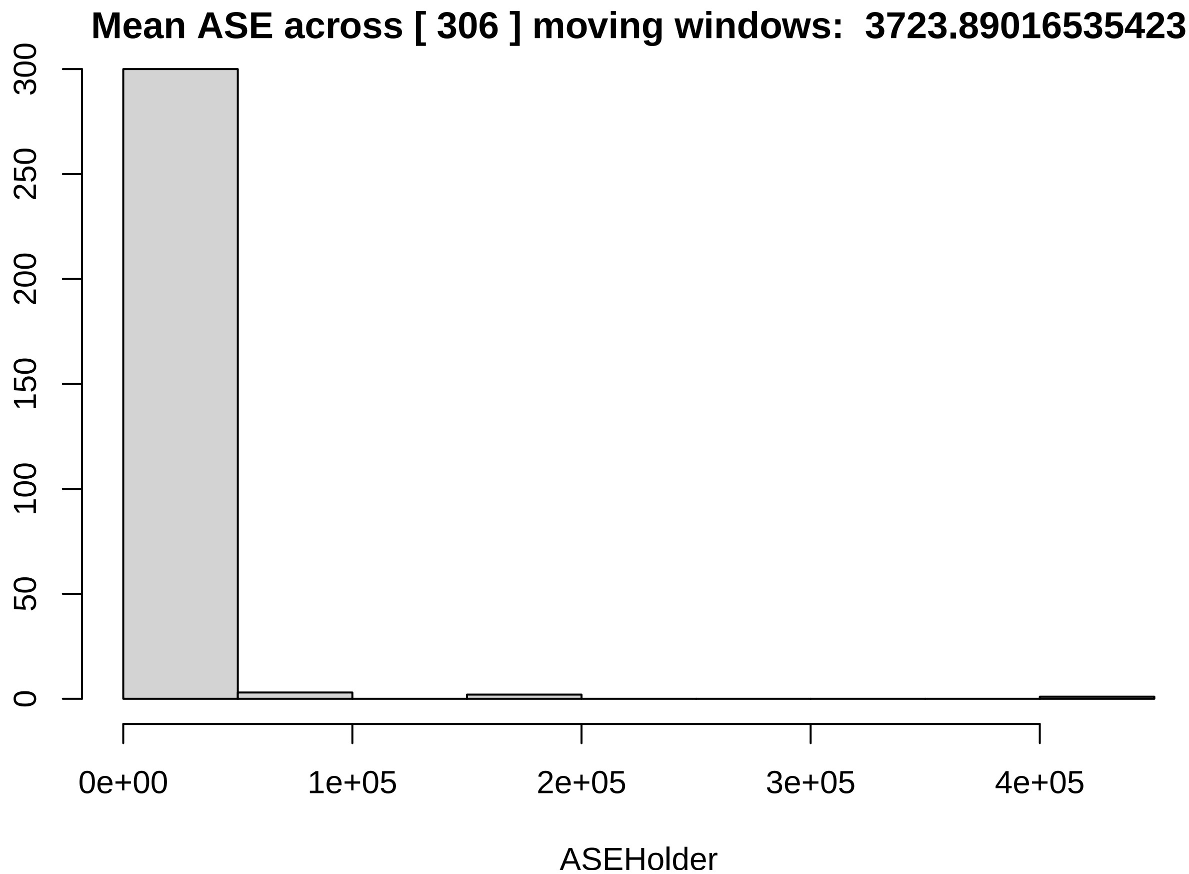




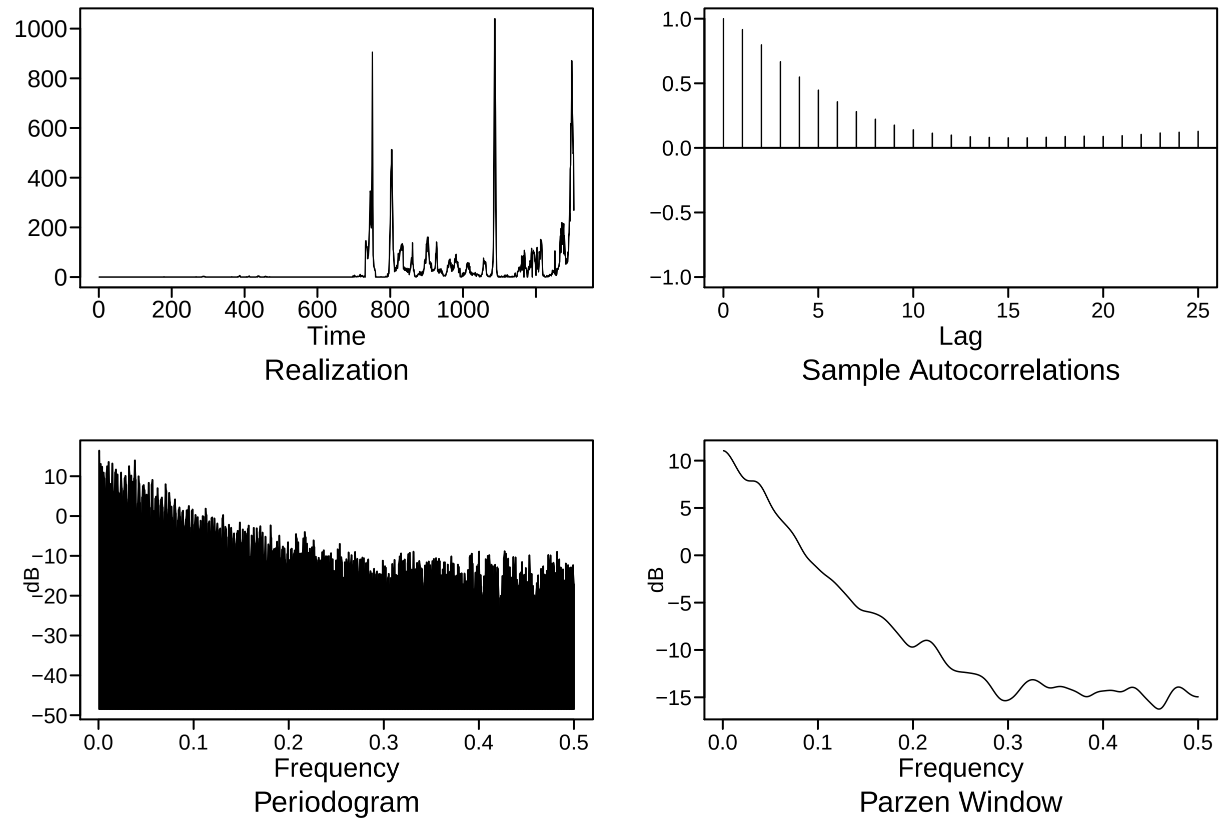

```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8483
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8483B              1.1788               0.8483       0.0000
##
##
```



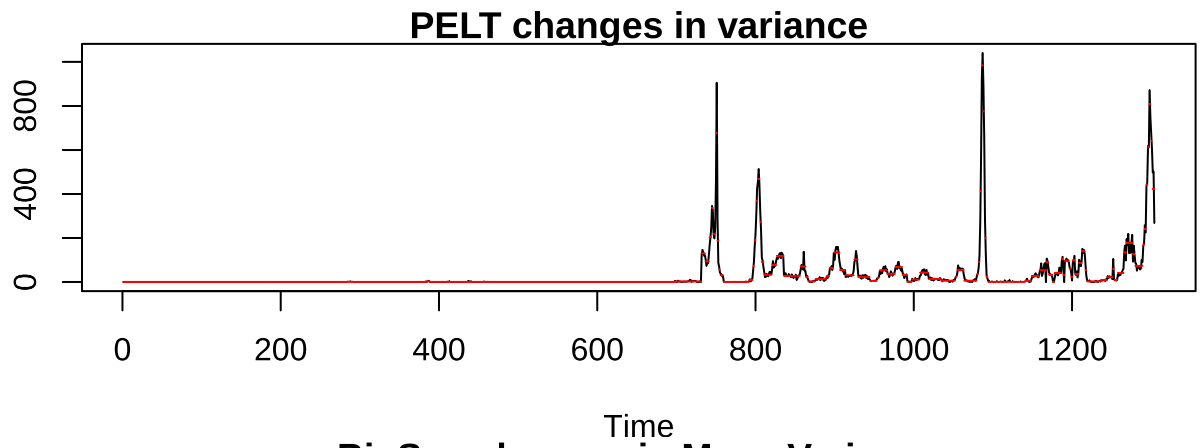
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
Time

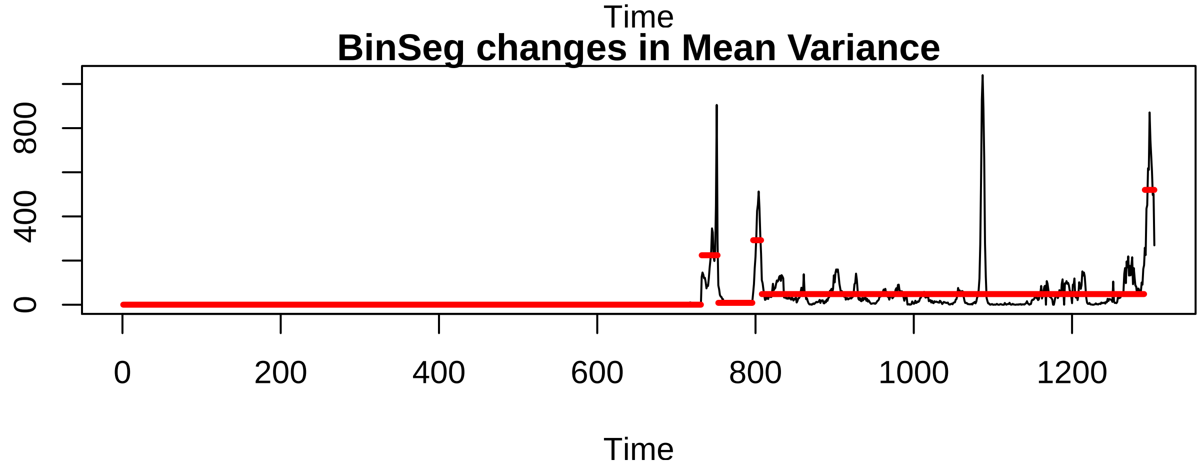



```
## Warning in adf.test(tdata): p-value smaller than printed p-value
```

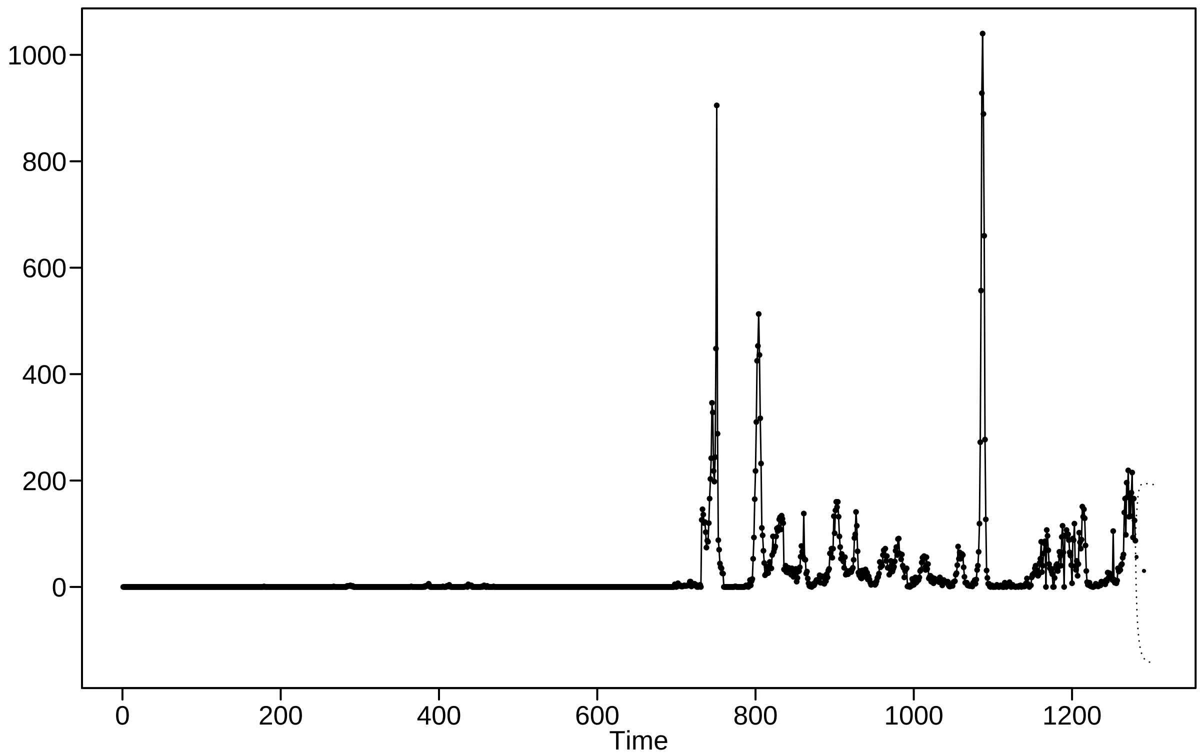


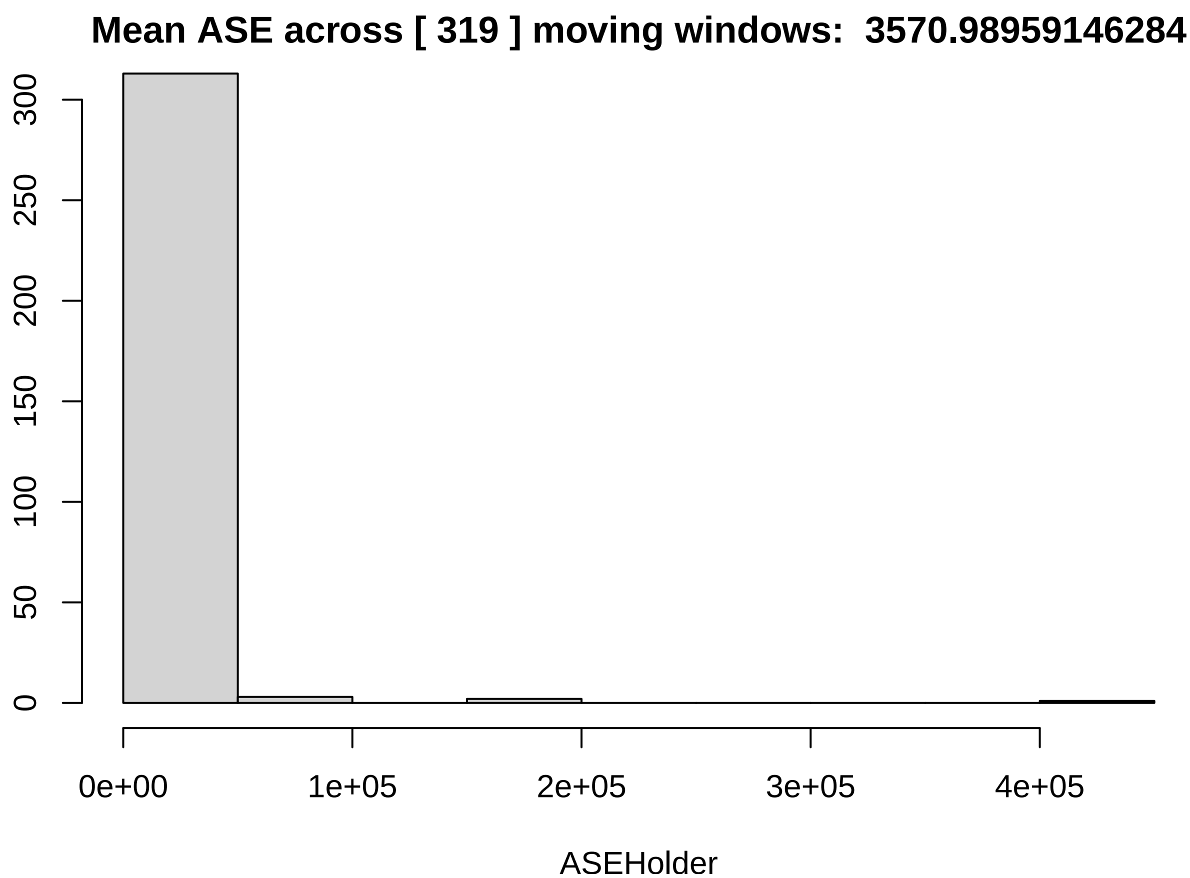

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```






```
## ---------WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of  aic
##
## Coefficients of Original polynomial:
## 0.8491
##
## Factor                 Roots                Abs Recip    System Freq
## 1-0.8491B              1.1778               0.8491       0.0000
##
##
```


1000
800
600
400
200
0
0
200
400
600
800
1000
1200
Time



hmm ggplots don't come through on the loop

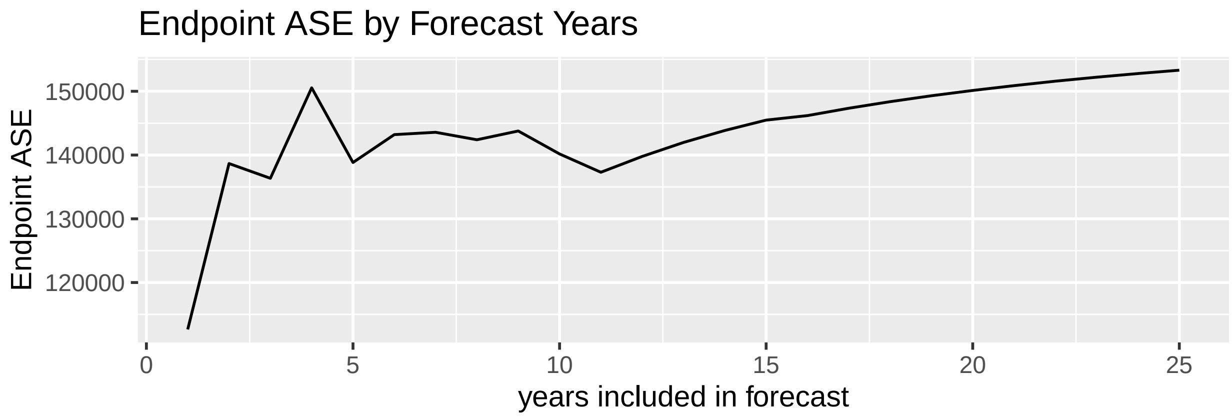
```
require(gridExtra)
```

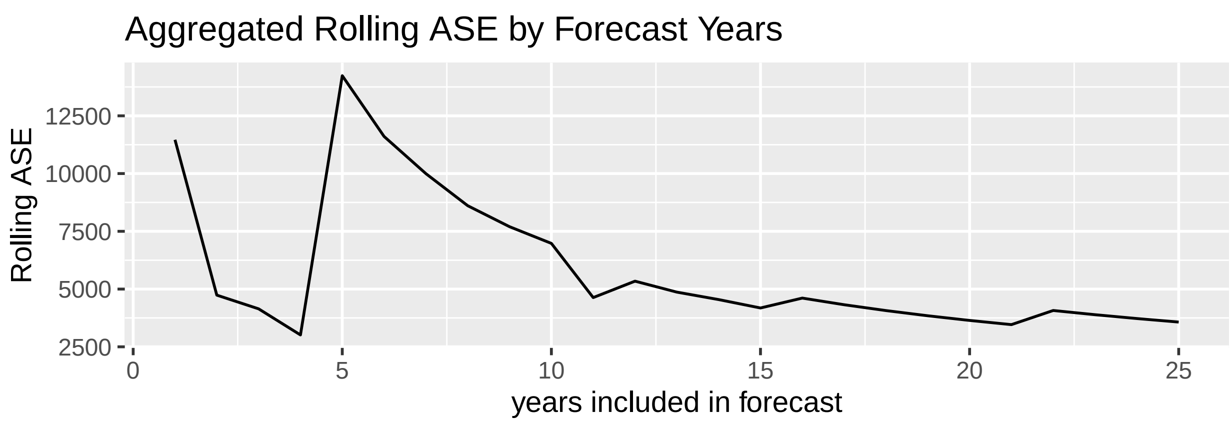
```
## Loading required package: gridExtra
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
g5 <- ggplot() + geom_line(aes(x = 1:length(meta_ep_ase), y = meta_ep_ase)) +
  ggtitle('Endpoint ASE by Forecast Years') +
  xlab('years included in forecast') + ylab('Endpoint ASE')

g6 <- ggplot() + geom_line(aes(x = 1:length(meta_roll_ase), y = meta_roll_ase)) +
  ggtitle('Aggregated Rolling ASE by Forecast Years') +
  xlab('years included in forecast') + ylab('Rolling ASE')

grid.arrange(g5, g6, nrow = 2)
```





Model evaluation metrics chage depending on criteria. Endpoint ASE performs the best at 1, 3, and 5 years of data used, getting worse the more data is included. This is likely due to the model's inability to adequately predict the large outbreaks approximately every 5 years. Rolling ASE performs the best at 4 years, while adding more years of data is likely due to a large number of NA's or sparse data.

#adjustable rolling window ase framework storage sliderInput(inputId = "roll_num", label = 'Training Data Length (Weeks)', min = 1, max = 52, value = 24), numericInput('horizonsize', label = 'Forecast Size (weeks)', value = 4), numericInput('stepsize', label = 'Step Size (weeks)', value = 8),

trainingSize = input$roll_{n}umhorizon = input$horizonsize
step_size = input$stepsize

```
n_windows = round((length(tdata)-(trainingSize + horizon))/step_size) #number of whole windows of trair
ASEHolder = numeric()
n_years = abs(input$num[1] - input$num[2])

fcastHolder = matrix(nrow = n_windows, ncol = horizon) #create matrix that has the number of rows and co
phis = m1$phi
thetas = m1$theta
s = 0
d = 0
for( i in 1:n_windows) #how many "windows" can fit into the total length of time, rounding down
{
  t_start = 1+(step_size*(i-1)) #starting point for each window's training data
  t_end = trainingSize + (step_size*(i-1)) #endpoint
  forecasts = fore.aruma.wge(tdata[t_start:t_end],phi = phis, theta = thetas, s = s, d = d, n.ahead = ho
  ASE = mean((tdata[(t_end + 1):(t_end+horizon)] - forecasts$f)^2)
  ASEHolder[i] = ASE
```

```
  fcastHolder[i, ] <- forecasts$f
}
```