PSTAT174 Final Project

Aaron Lee (3410388)

2023-05-30

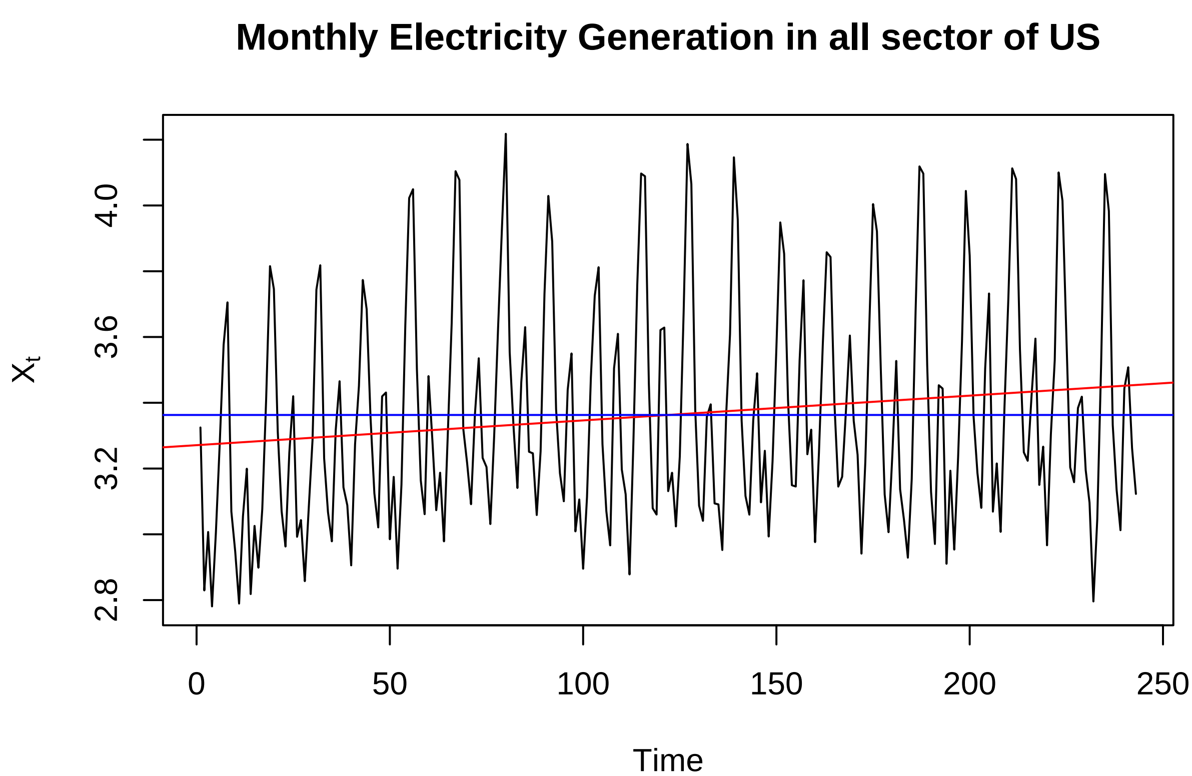
Energy Generation Data

```
electricity.csv <- read.table("electricity_data.csv", sep = ",", header = FALSE, skip = 1, nrows = 255)
electricity <- ts(electricity.csv[, 2], start = c(2001, 1, 1), frequency=12)

electricity1 = electricity[c(1: 243)]/100000
electricity1_test = electricity[c(244: 255)]/100000
```

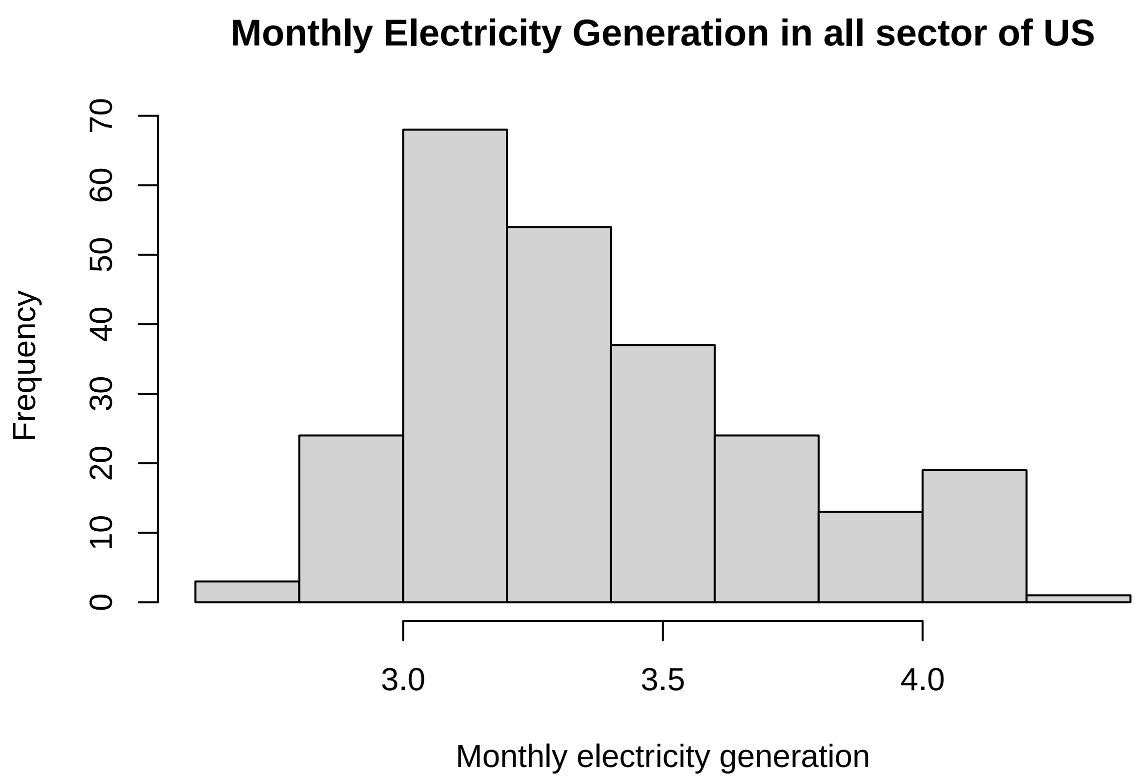
Plotting the original data

```
ts.plot(electricity1, main="Monthly Electricity Generation in all sector of US", ylab=expression(X[t]))
ele_fit <- lm(electricity1 ~ as.numeric(1:length(electricity1))); abline(ele_fit, col="red")
abline(h=mean(electricity1), col="blue")
```

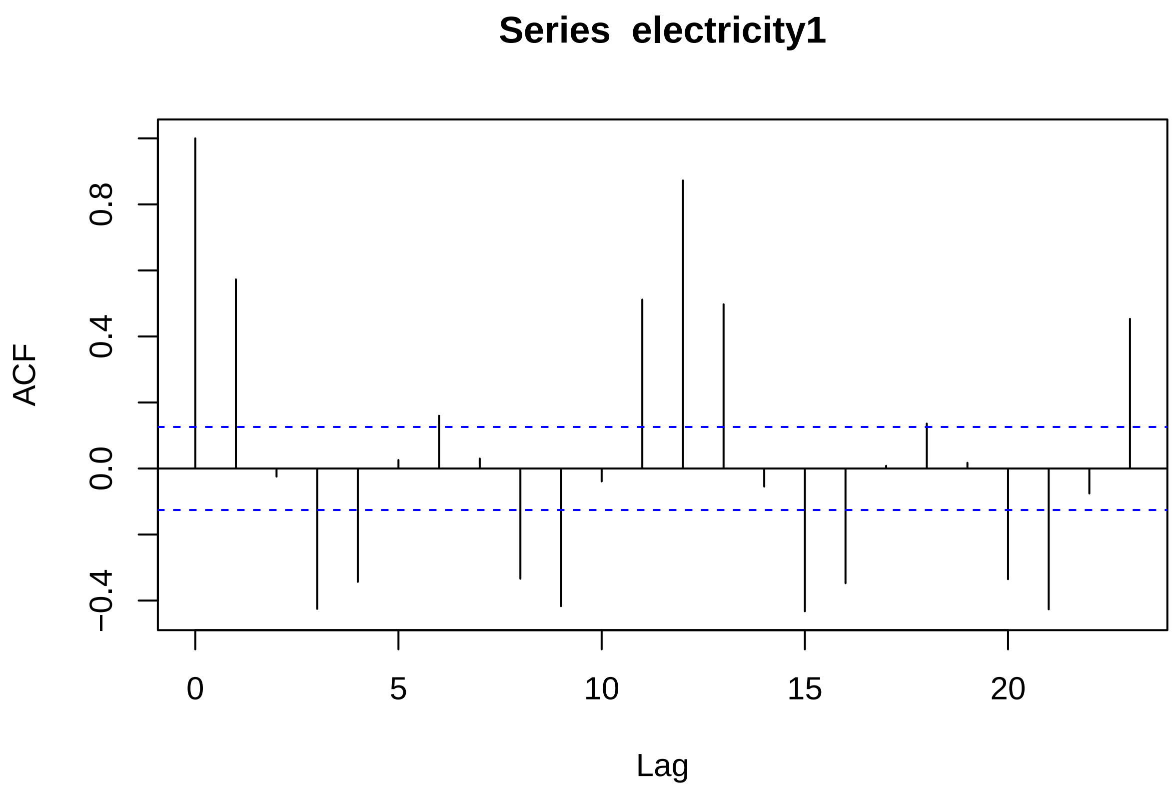


- Not stationary; we can see that there is an upper trend; seasonality; No constant variance.

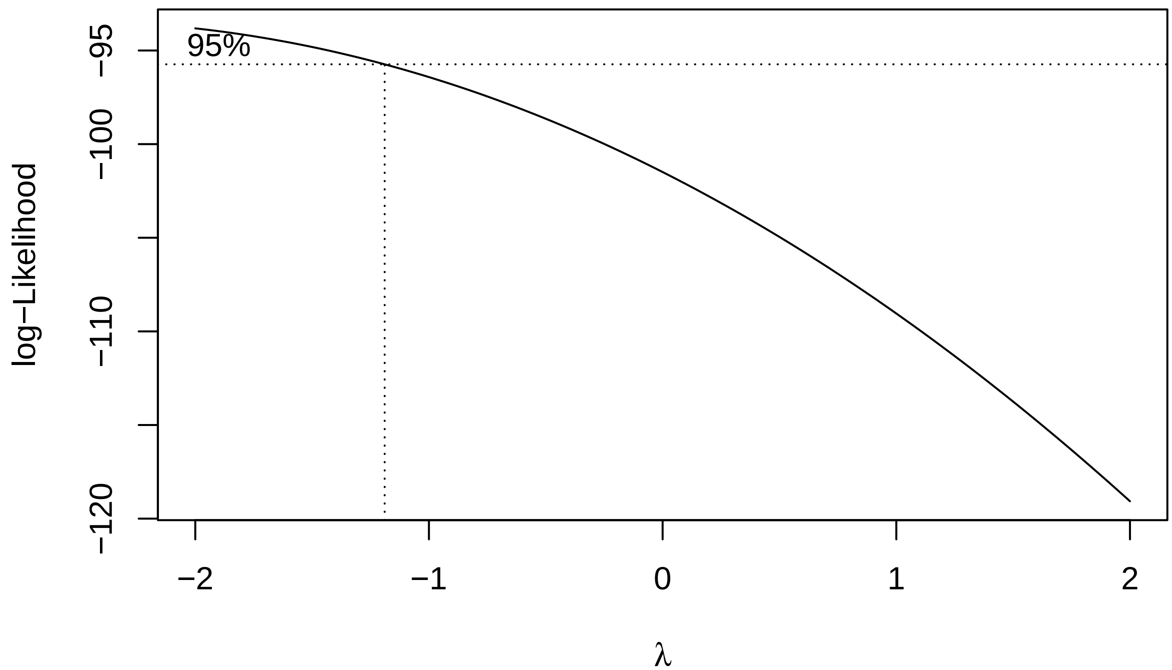
```
hist(electricity1, main="Monthly Electricity Generation in all sector of US", xlab="Monthly electricity
```




```
acf(electricity1)
```




```
library(MASS)
t <- 1:length(electricity1)
bcTransform <- boxcox(electricity1 ~ t, plotit=TRUE)
```

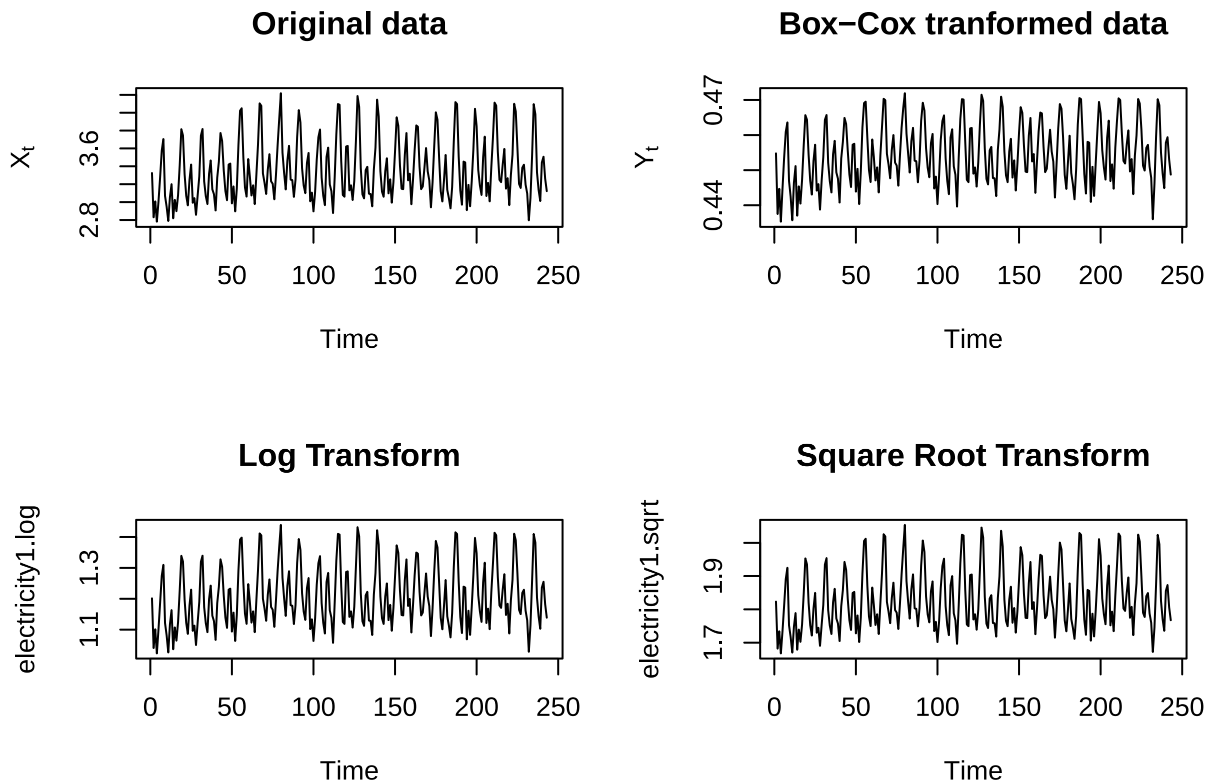



```
bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
```

```
## [1] -2
```

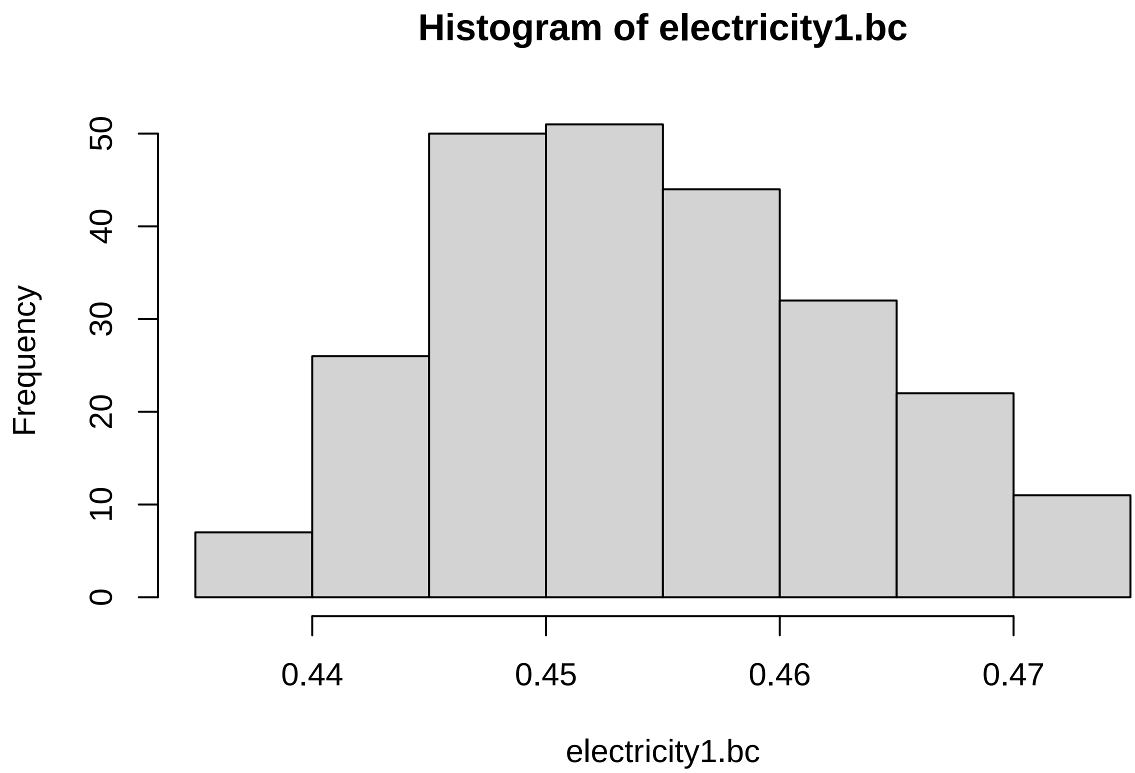
```
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
electricity1.bc = (1/lambda)*(electricity1^lambda-1)
electricity1.log = log(electricity1)
electricity1.sqrt = sqrt(electricity1)

op <- par(mfrow = c(2,2))
ts.plot(electricity1,main = "Original data",ylab = expression(X[t]))
ts.plot(electricity1.bc,main = "Box-Cox tranformed data", ylab = expression(Y[t]))
ts.plot(electricity1.log, main = "Log Transform")
ts.plot(electricity1.sqrt, main = "Square Root Transform")
```




```
par(op)
```

```
hist(electricity1.bc)
```

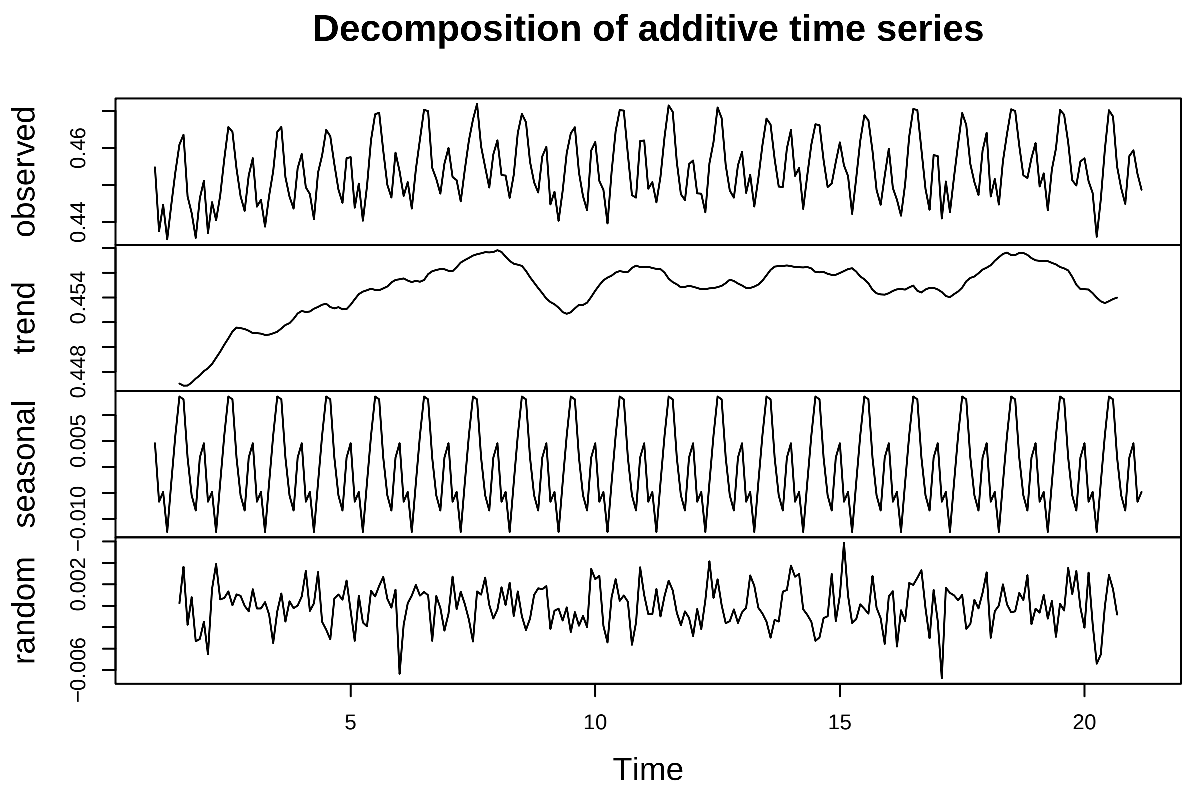


- Through the histogram, we choose box-cox transformation.

Produce decomposition of Box-Cox U_t

```
library(ggplot2)
#install.packages('ggfortify')
library(ggfortify)

y <- ts(as.ts(electricity1.bc), frequency = 12)
decomp <- decompose(y)
plot(decomp)
```




```
# Calculate the sample variance and plot the acf/pacf
var(electricity1)
```

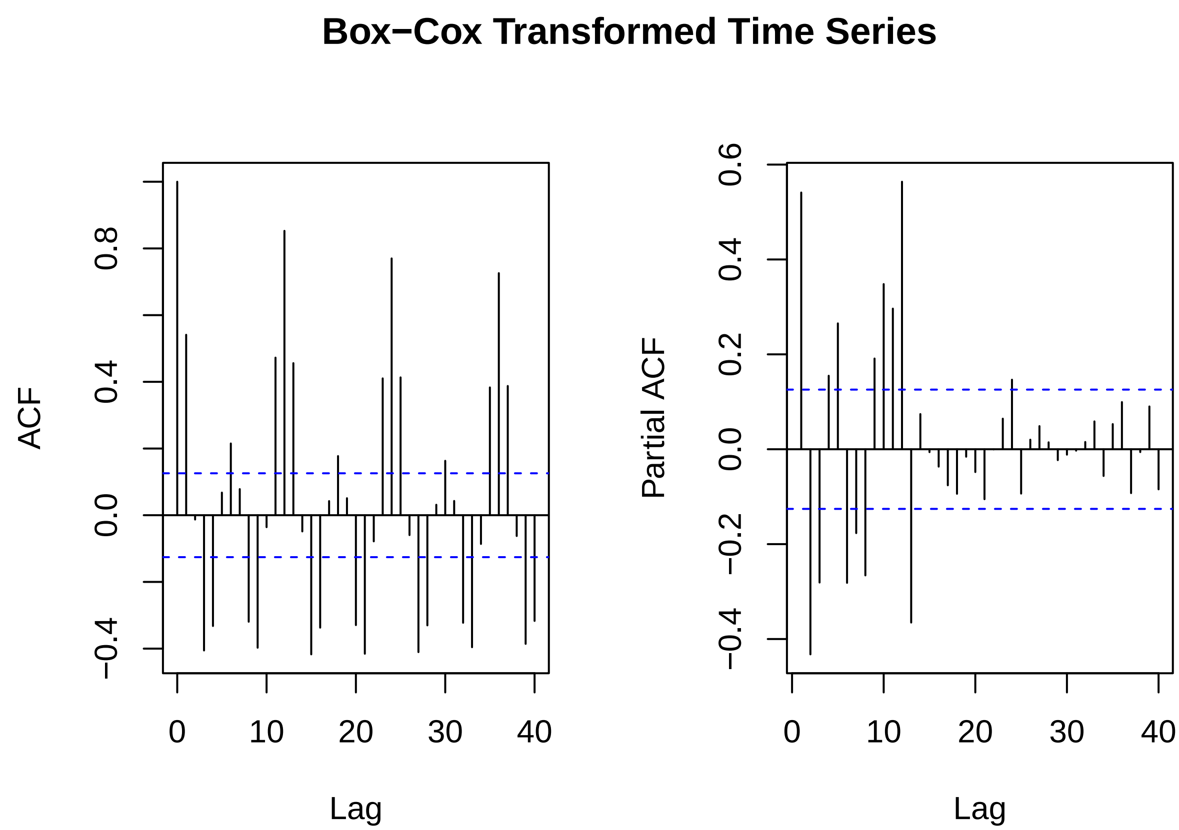
```
## [1] 0.116328
```

```
var(electricity1.bc)
```

```
## [1] 7.342597e-05
```

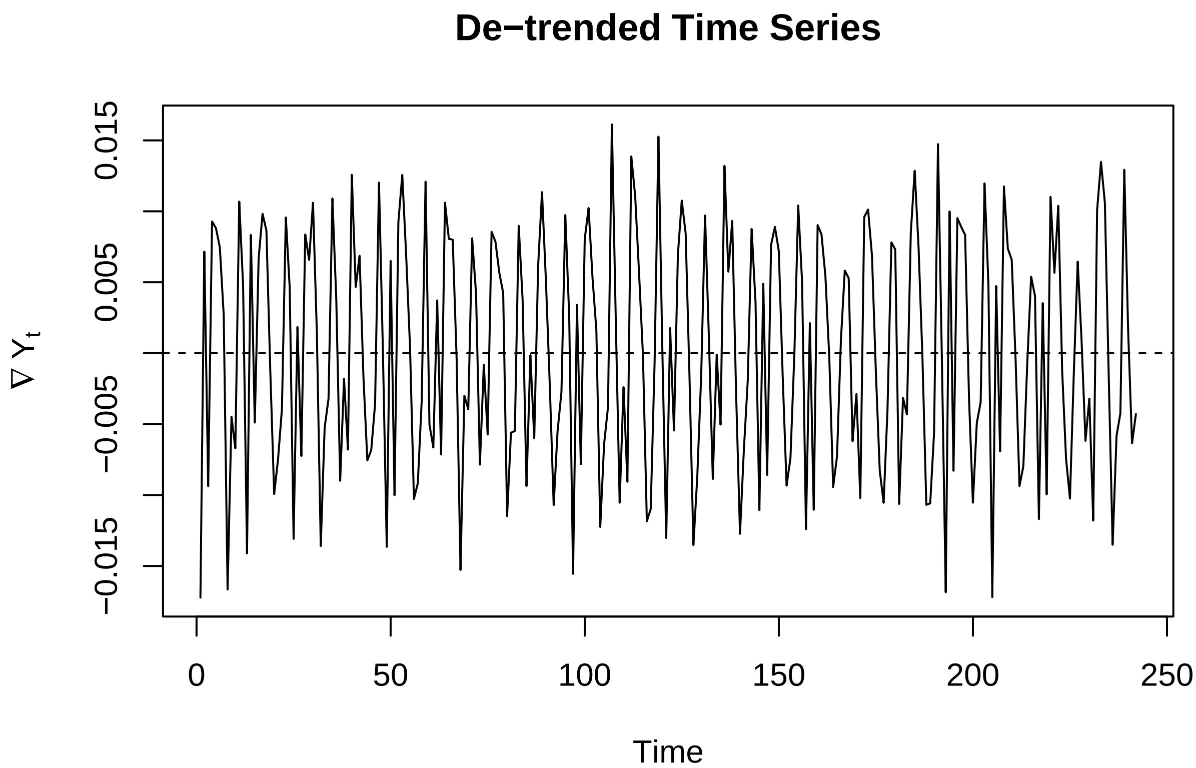
The variance increases after the transformation.

```
op = par(mfrow = c(1,2))
acf(electricity1.bc,lag.max = 40,main = "")
pacf(electricity1.bc,lag.max = 40,main = "")
title("Box-Cox Transformed Time Series", line = -1, outer=TRUE)
```




```r
par(op)
```

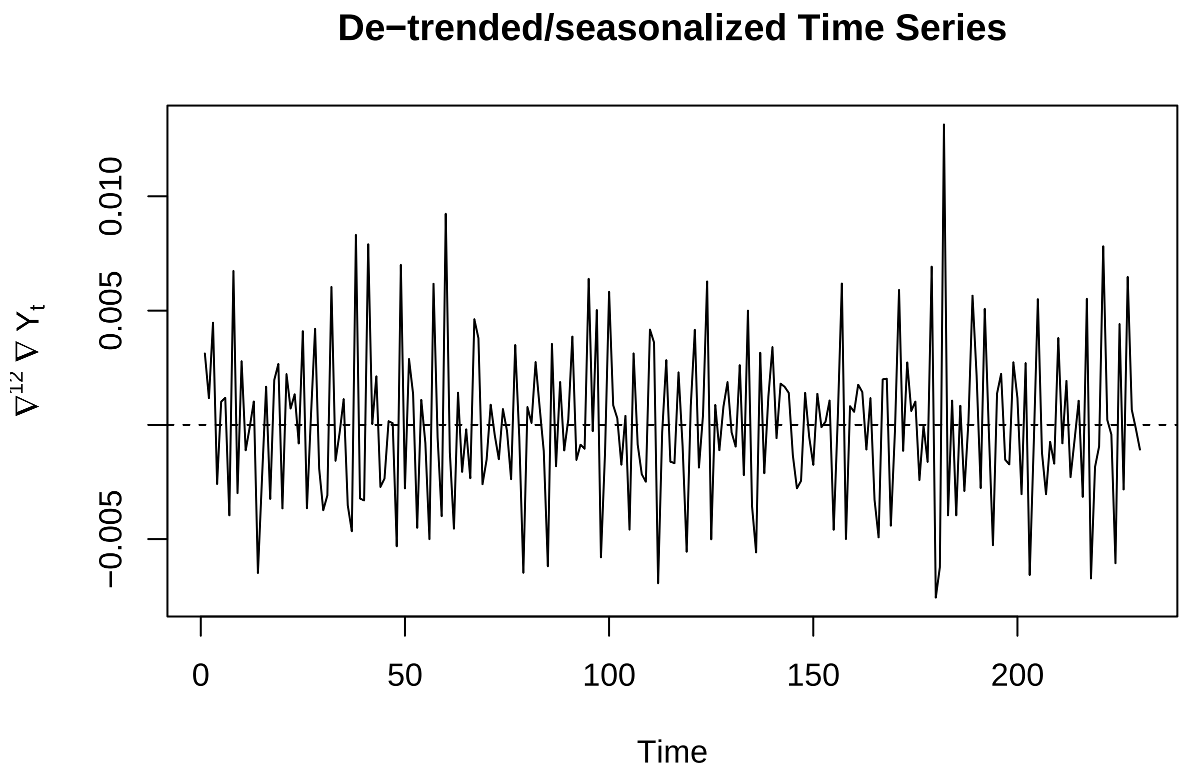
```r
# Difference at lag  = 1 to remove trend component
y1 = diff(electricity1.bc, 1)
plot.ts(y1,main = "De-trended Time Series",ylab = expression(nabla~Y[t]))
abline(h = 0,lty = 2)
```




```
var(y1) # smaller that 7.342619e-5
```

```
## [1] 6.7531e-05
```

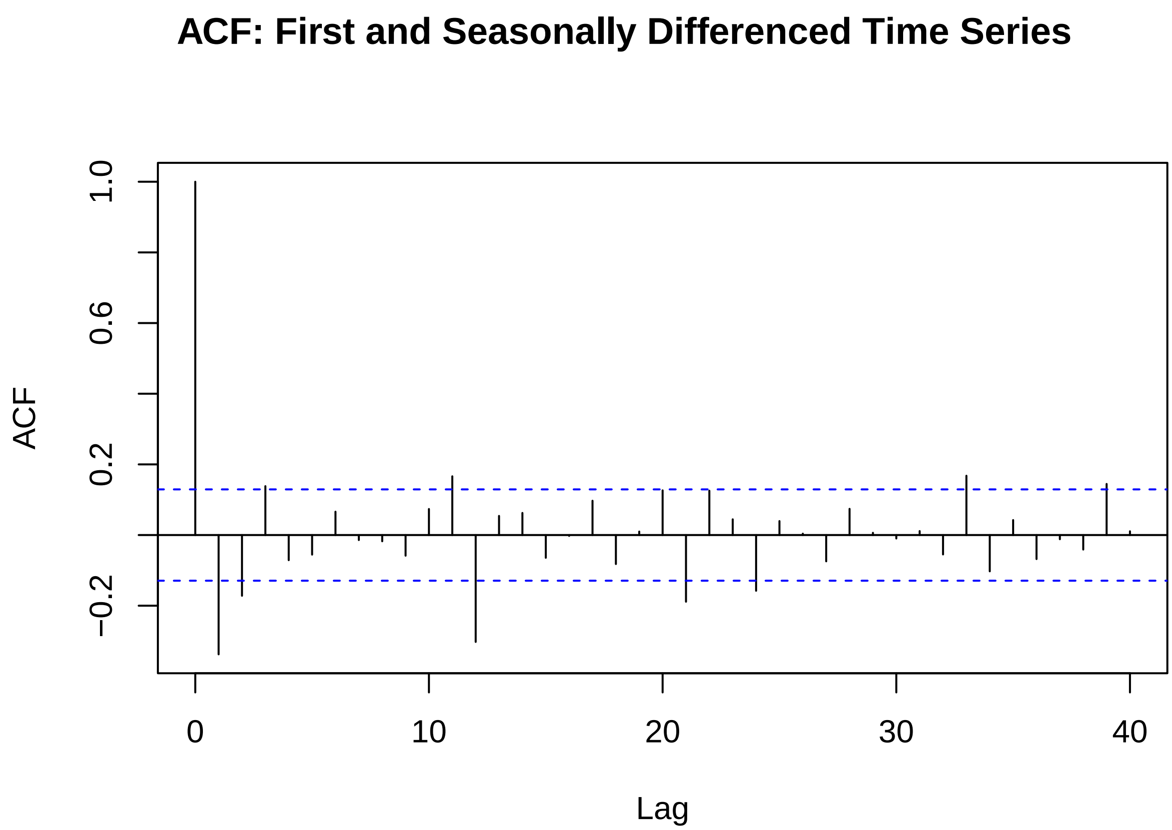
```
# Difference at lag  = 12 (cycle determined by the ACF) to remove seasonal component
y12 = diff(y1, 12)
plot.ts(y12, main = "De-trended/seasonalized Time Series", ylab = expression(nabla^{12}~nabla~Y[t]))
abline(h = 0,lty = 2)
```



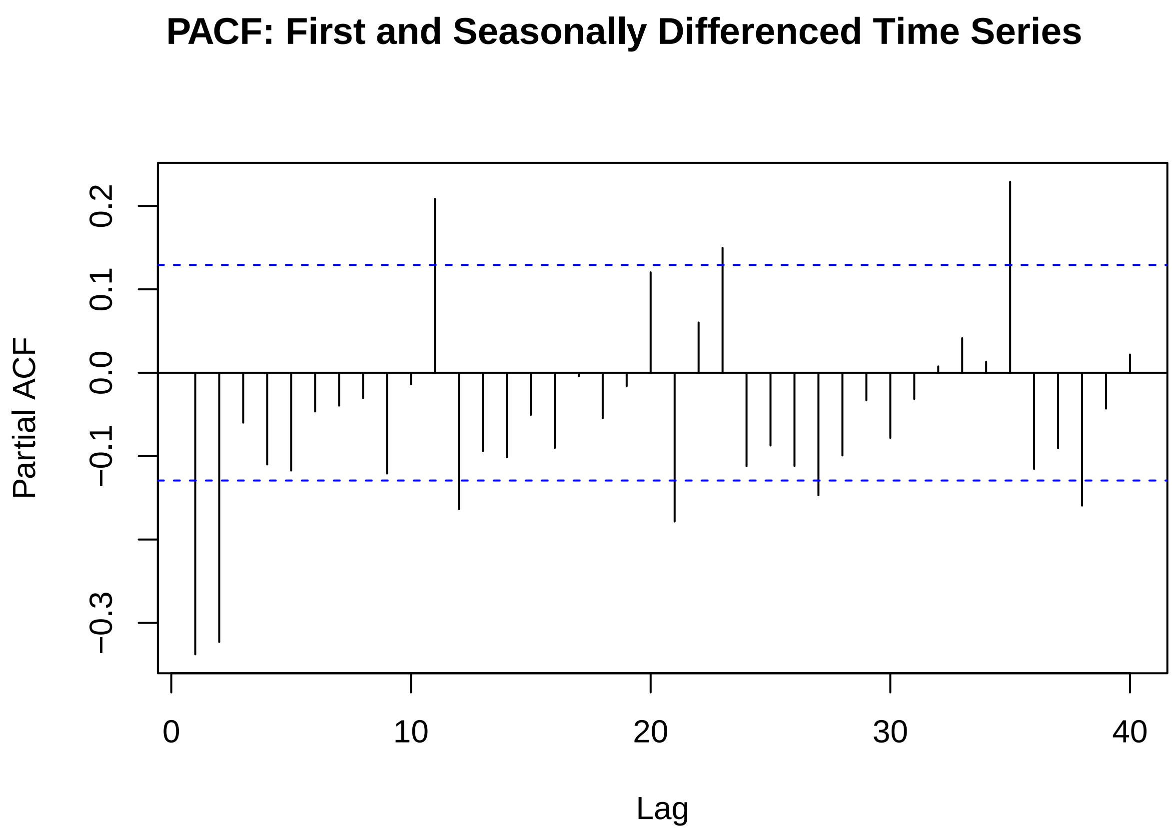

```
var(y12) # smaller than 7.342619e-5 and 6.753134e-5
```

```
## [1] 1.163955e-05
```

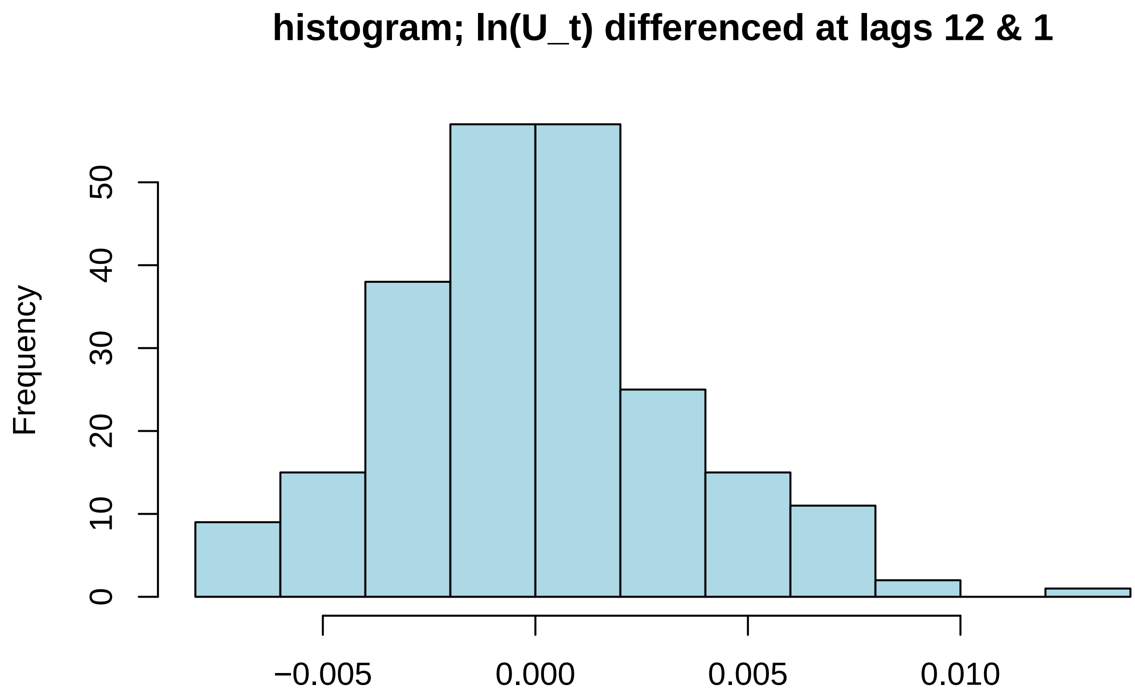
```
acf(y12,lag.max = 40,main = "")
title("ACF: First and Seasonally Differenced Time Series", line = -1, outer = TRUE)
```



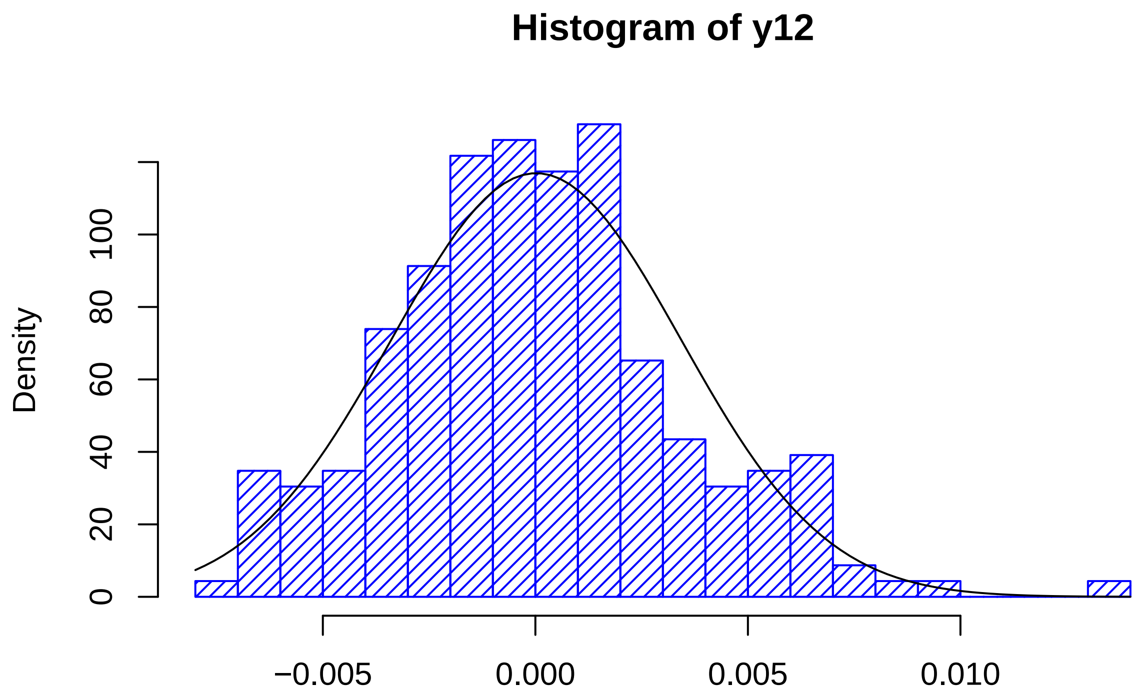

```r
pacf(y12,lag.max = 40,main = "")
title("PACF: First and Seasonally Differenced Time Series", line = -1, outer = TRUE)
```




```
hist(y12, col="light blue", xlab="", main="histogram; ln(U_t) differenced at lags 12 & 1")
```




```r
# Compare histograms of Box-Cox (Ut) to the normal curve, really similar.
hist(y12, density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m<-mean(y12)
std<- sqrt(var(y12))
curve( dnorm(x,m,std), add=TRUE )
```



Modeling the seasonal part (P, D, Q): For this part, focus on the seasonal lags h = 1s, 2s, etc.

- We applied one seasonal differencing so D = 1 at lag s = 12.
- The ACF shows a strong peak at h = 1s and smaller peaks appearing at h = 2s. A good choice for the MA part could be Q = 1 or Q = 2.
- The PACF shows there is a peak at h = 1s. A good choice for the AR part could be P = 1.

Modeling the non-seasonal part (p , d, q): In this case focus on the within season lags, h = 1,. . . ,11.

- We applied one differencing to remove the trend: d = 1.
- A good choice for the MA part could be q = 0 or q = 1 respectively.
- A good choice for the AR part could be p = 2

Also, the model might be MA(33); SARIMA(2,1,0)(1,1,1)[12]; SARIMA(2,1,1)(1,1,1)[12]; SARIMA(2,1,0)(1,1,2)[12]; SARIMA(2,1,1)(1,1,2)[12]

Trying Models:

```
library(astsa)
library(MuMIn)
arima(electricity1.bc, order = c(0,1,1), seasonal = list(order = c(0,1,2), period = 12), method="ML")
```

SMA models tried: Q=1, 2, q=0,1. Model producing the lowest AICc:

```
##
```

```
## Call:
## arima(x = electricity1.bc, order = c(0, 1, 1), seasonal = list(order = c(0,
##     1, 2), period = 12), method = "ML")
##
## Coefficients:
##           ma1     sma1     sma2
##       -0.6406  -0.7834  -0.2164
## s.e.   0.0674   0.1087   0.0773
##
## sigma^2 estimated as 5.337e-06:  log likelihood = 1053.52,  aic = -2099.04
```

```
# Calculating AICc
AICc(arima(electricity1.bc, order = c(0,1,1), seasonal = list(order = c(0,1,2), period = 12), method="MI
```

```
## [1] -2098.865
```

```
arima(electricity1.bc, order = c(0,1,0), seasonal = list(order = c(0,1,2), period = 12), method="ML")
```

```
##
## Call:
## arima(x = electricity1.bc, order = c(0, 1, 0), seasonal = list(order = c(0,
##     1, 2), period = 12), method = "ML")
##
## Coefficients:
##          sma1     sma2
##       -0.7795  -0.2205
## s.e.   0.1666   0.0788
##
## sigma^2 estimated as 6.906e-06:  log likelihood = 1024.18,  aic = -2042.36
```

```
AICc(arima(electricity1.bc, order = c(0,1,0), seasonal = list(order = c(0,1,2), period = 12), method="MI
```

```
## [1] -2042.249
```

```
arima(electricity1.bc, order = c(0,1,1), seasonal = list(order = c(0,1,1), period = 12), method="ML")
```

```
##
## Call:
## arima(x = electricity1.bc, order = c(0, 1, 1), seasonal = list(order = c(0,
##     1, 1), period = 12), method = "ML")
##
## Coefficients:
##           ma1     sma1
##       -0.6530  -0.9816
## s.e.   0.0689   0.2676
##
## sigma^2 estimated as 5.498e-06:  log likelihood = 1050.11,  aic = -2094.22
```

```
AICc(arima(electricity1.bc, order = c(0,1,1), seasonal = list(order = c(0,1,1), period = 12), method="MI
```

```
## [1] -2094.118
```

```
arima(electricity1.bc, order = c(0,1,0), seasonal = list(order = c(0,1,1), period = 12), method="ML")
```

```
##
## Call:
## arima(x = electricity1.bc, order = c(0, 1, 0), seasonal = list(order = c(0,
##     1, 1), period = 12), method = "ML")
```

```
##
## Coefficients:
##          sma1
##       -0.9121
## s.e.   0.0668
##
## sigma^2 estimated as 7.481e-06:  log likelihood = 1020.46,  aic = -2036.92
```

```
AICc(arima(electricity1.bc, order = c(0,1,0), seasonal = list(order = c(0,1,2), period = 12), method="MI
```

```
## [1] -2042.249
```

SAR

```
arima(electricity1.bc, order = c(2,1,0), seasonal = list(order = c(1,1,0), period = 12), method="ML")
```

```
##
## Call:
## arima(x = electricity1.bc, order = c(2, 1, 0), seasonal = list(order = c(1,
##     1, 0), period = 12), method = "ML")
##
## Coefficients:
##           ar1      ar2     sar1
##       -0.4371  -0.3102  -0.3058
## s.e.   0.0629   0.0627   0.0656
##
## sigma^2 estimated as 8.353e-06:  log likelihood = 1017.57,  aic = -2027.14
```

```
AICc(arima(electricity1.bc, order = c(2,1,0), seasonal = list(order = c(1,1,0), period = 12), method="MI
```

```
## [1] -2026.967
```

SARIMA(2,1,1)(1,1,2)_s=12

```
arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,2), period = 12), method="ML")
```

```
##
## Call:
## arima(x = electricity1.bc, order = c(2, 1, 1), seasonal = list(order = c(1,
##     1, 2), period = 12), method = "ML")
##
## Coefficients:
##          ar1     ar2      ma1     sar1     sma1     sma2
##       0.3109  0.0653  -0.8828  -0.2344  -0.5707  -0.4286
## s.e.  0.0922  0.0836   0.0636   0.2006   0.2238   0.1865
##
## sigma^2 estimated as 5.103e-06:  log likelihood = 1057.82,  aic = -2101.64
```

```
AICc(arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,2), period = 12), method="MI
```

```
## [1] -2101.137
```

Best fit model

```
arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,2), period = 12), fixed = c(NA,0
```

```
## Warning in arima(electricity1.bc, order = c(2, 1, 1), seasonal = list(order =
## c(1, : some AR parameters were fixed: setting transform.pars = FALSE

##
## Call:
## arima(x = electricity1.bc, order = c(2, 1, 1), seasonal = list(order = c(1,
##     1, 2), period = 12), fixed = c(NA, 0, NA, NA, NA, NA), method = "ML")
##
## Coefficients:
##          ar1  ar2      ma1     sar1     sma1     sma2
##       0.2889    0  -0.8478  -0.2373  -0.5696  -0.4305
## s.e.  0.0995    0   0.0644   0.2002   0.2199   0.1861
##
## sigma^2 estimated as 5.117e-06:  log likelihood = 1057.53,  aic = -2103.06
```

```
AICc(arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,2), period = 12), fixed = c
```

```
## Warning in arima(electricity1.bc, order = c(2, 1, 1), seasonal = list(order =
## c(1, : some AR parameters were fixed: setting transform.pars = FALSE

## [1] -2102.679
```

MA(33) AICc is not smaller than -2102.679

```
arima(electricity1.bc, order = c(0,0,33), seasonal = list(order = c(0,0,0), period = 12),method="ML")
```

```
## Warning in arima(electricity1.bc, order = c(0, 0, 33), seasonal = list(order =
## c(0, : possible convergence problem: optim gave code = 1

##
## Call:
## arima(x = electricity1.bc, order = c(0, 0, 33), seasonal = list(order = c(0,
##     0, 0), period = 12), method = "ML")
##
## Coefficients:
##          ma1     ma2     ma3      ma4     ma5     ma6     ma7      ma8      ma9
##       0.6037  0.2032  0.0764  -0.1432  0.0813  0.2086  0.1872  -0.0921  -0.3215
## s.e.  0.3901  0.1977  0.3881   0.6862  0.6604  0.1700  0.5087   0.7919   0.4348
##          ma10    ma11    ma12    ma13    ma14     ma15     ma16    ma17    ma18
##       -0.0305  0.1992  1.2492  0.8014  0.2644  -0.0138  -0.2634  0.1866  0.1651
## s.e.   0.2887  0.6221  0.5426  0.2538  0.5357   0.2372   0.3902  0.8067  0.5426
##          ma19     ma20     ma21     ma22    ma23    ma24    ma25    ma26
##       -0.0392  -0.1944  -0.5625  -0.0995  0.0577  0.7052  0.4498  0.0647
## s.e.   0.1613   0.5941   0.7192   0.2907  0.2164  0.5724  0.3857  0.1041
##          ma27     ma28    ma29    ma30     ma31     ma32     ma33  intercept
##       -0.0688  -0.1214  0.1341  0.0441  -0.1045  -0.0982  -0.0926     0.4542
## s.e.   0.1676   0.1766  0.1161  0.1038   0.1355   0.2082   0.1057     0.0009
##
## sigma^2 estimated as 9.972e-06:  log likelihood = 1031.8,  aic = -1993.61
```

```
AICc(arima(electricity1.bc, order = c(0,0,33), seasonal = list(order = c(0,0,0), period = 12),method="MI
```

```
## Warning in arima(electricity1.bc, order = c(0, 0, 33), seasonal = list(order =
## c(0, : possible convergence problem: optim gave code = 1

## [1] -1981.432
```

```
arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,1), period = 12),method="ML")
```

```
##
## Call:
## arima(x = electricity1.bc, order = c(2, 1, 1), seasonal = list(order = c(1,
##     1, 1), period = 12), method = "ML")
##
## Coefficients:
##          ar1     ar2      ma1     sar1      sma1
##       0.3157  0.0656  -0.8819  0.1519  -0.9999
## s.e.  0.0940  0.0845   0.0660  0.0735   0.1372
##
## sigma^2 estimated as 5.197e-06:  log likelihood = 1055.99,  aic = -2099.97
```

```
AICc(arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,1), period = 12),method="ML
```

```
## [1] -2099.598
```

second less AICc

```
arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,1), period = 12), fixed = c(NA,0
```

```
## Warning in arima(electricity1.bc, order = c(2, 1, 1), seasonal = list(order =
## c(1, : some AR parameters were fixed: setting transform.pars = FALSE
```

```
##
## Call:
## arima(x = electricity1.bc, order = c(2, 1, 1), seasonal = list(order = c(1,
##     1, 1), period = 12), fixed = c(NA, 0, NA, NA, NA), method = "ML")
##
## Coefficients:
##          ar1  ar2      ma1    sar1     sma1
##       0.2924    0  -0.8456  0.1500  -1.0001
## s.e.  0.1005    0   0.0653  0.0729   0.1313
##
## sigma^2 estimated as 5.214e-06:  log likelihood = 1055.7,  aic = -2101.39
```

```
AICc(arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,1), period = 12), fixed = c
```

```
## Warning in arima(electricity1.bc, order = c(2, 1, 1), seasonal = list(order =
## c(1, : some AR parameters were fixed: setting transform.pars = FALSE
```

```
## [1] -2101.127
```

- not invertible, because 1.0001 is bigger than 1

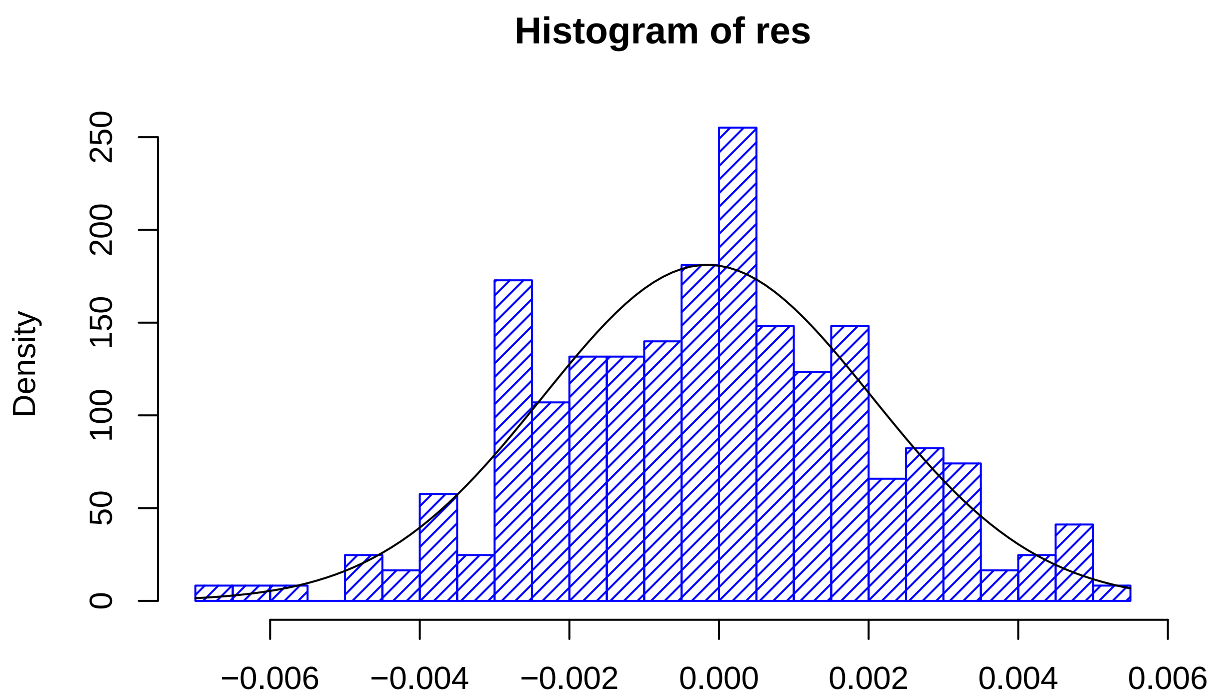
Diagnostic checking

```
#fit <- arima(electricity1.bc, order=c(2,1,1), seasonal = list(order = c(1,1,2), period = 12), method="
fit <- arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,2), period = 12), fixed =
```

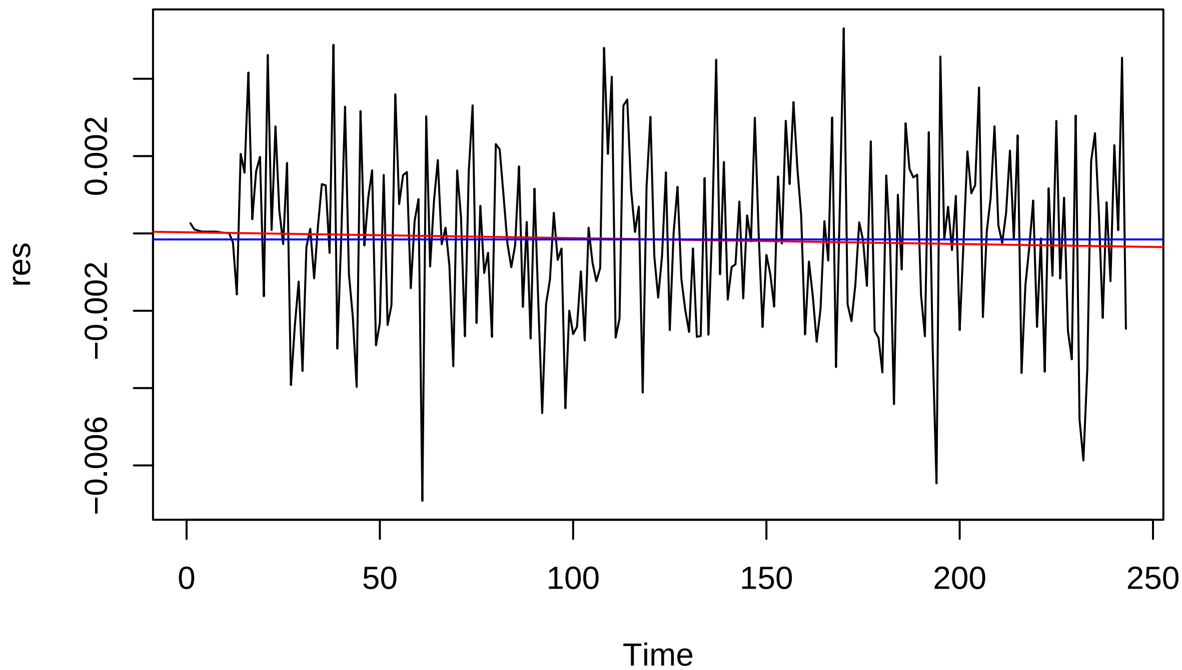
```
## Warning in arima(electricity1.bc, order = c(2, 1, 1), seasonal = list(order =
## c(1, : some AR parameters were fixed: setting transform.pars = FALSE
```

```
res <- residuals(fit)
hist(res,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(res)
```

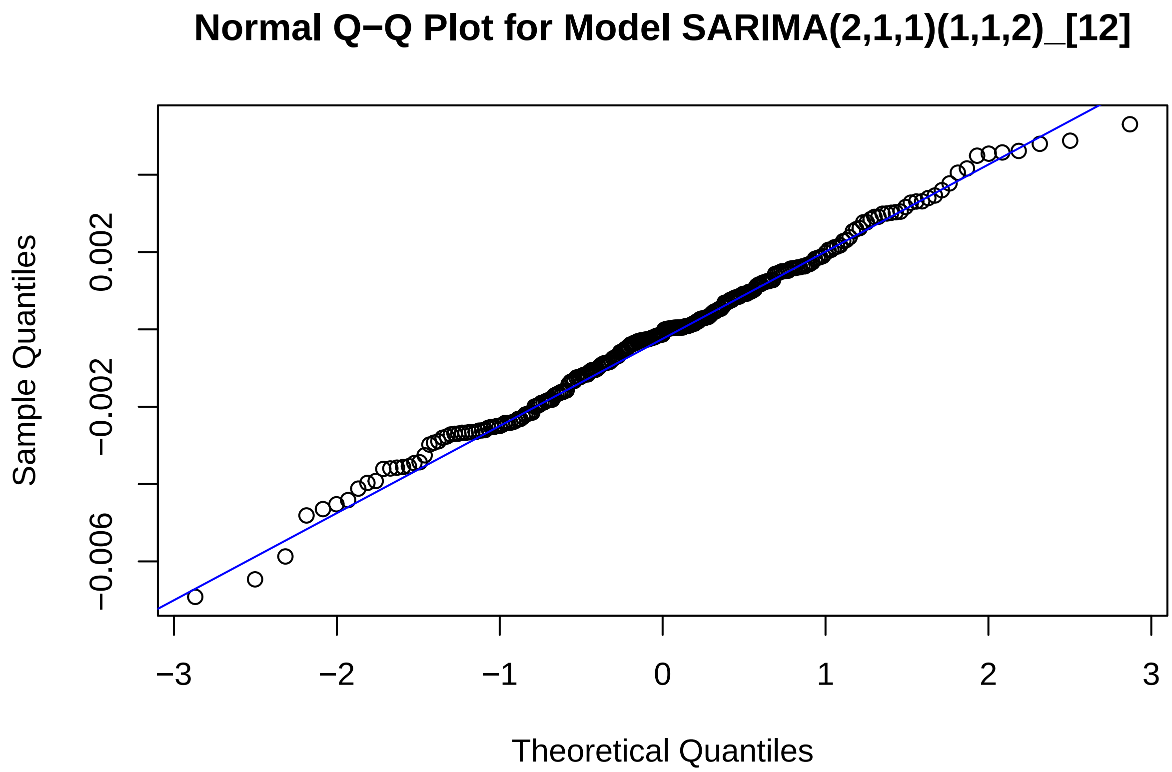
```
std <- sqrt(var(res))
curve( dnorm(x,m,std), add=TRUE )
```



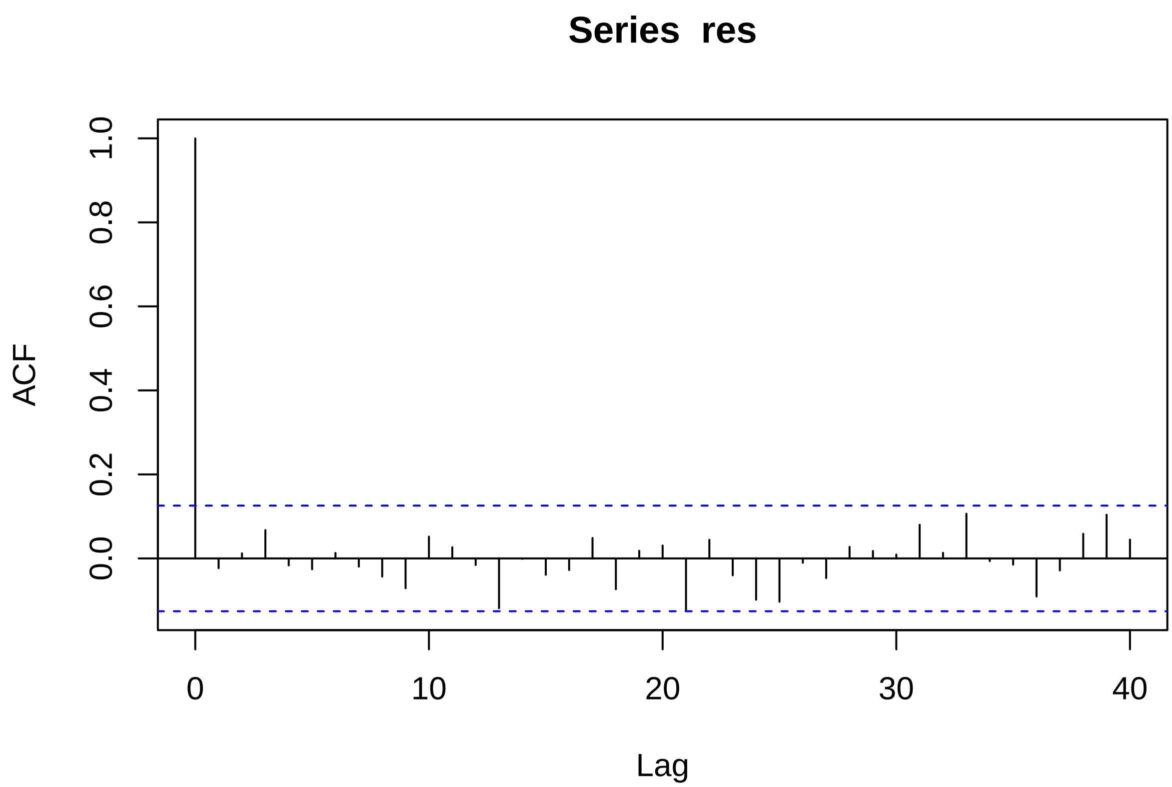

```
plot.ts(res)
fitt <- lm(res~as.numeric(1:length(res))); abline(fitt, col="red")
abline(h=mean(res), col="blue")
```



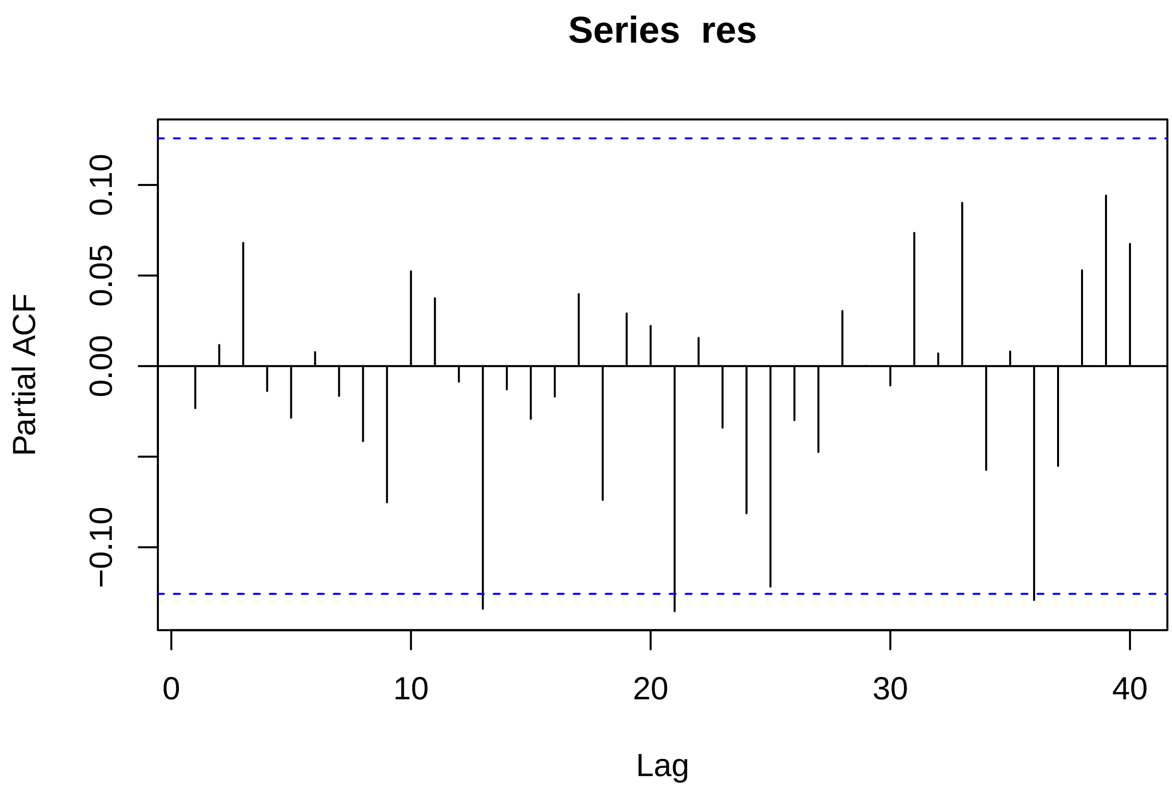

```
qqnorm(res,main= "Normal Q-Q Plot for Model SARIMA(2,1,1)(1,1,2)_[12]")
qqline(res,col="blue")
```




```
acf(res, lag.max=40)
```



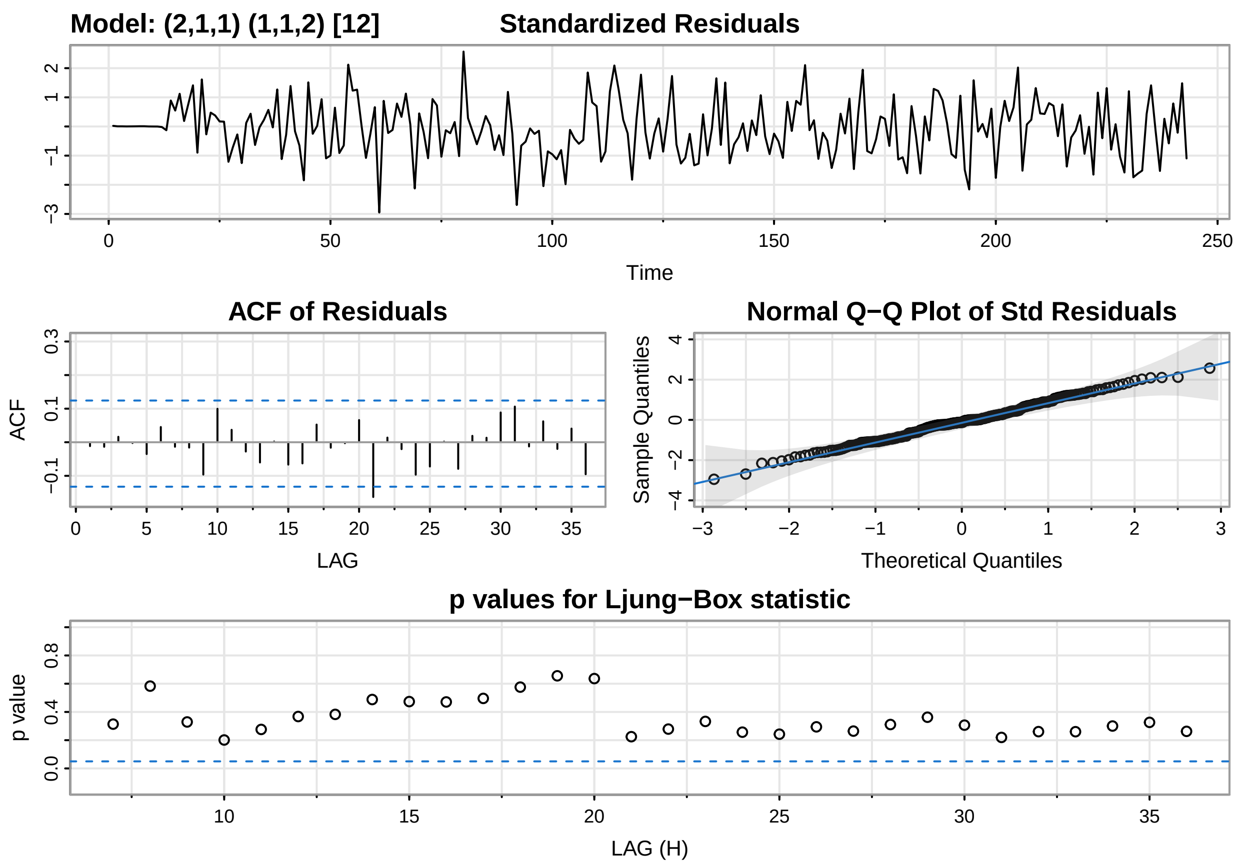

```
pacf(res, lag.max=40)
```




```
fit.i <- sarima(xdata=electricity1, p=2, d=1, q=1, P=1, D=1, Q=2, S=12)
```

```
## initial  value -2.038133
## iter   2 value -2.261471
## iter   3 value -2.295944
## iter   4 value -2.327468
## iter   5 value -2.341883
## iter   6 value -2.350212
## iter   7 value -2.356947
## iter   8 value -2.362372
## iter   9 value -2.376226
## iter  10 value -2.382971
## iter  11 value -2.388489
## iter  12 value -2.388997
## iter  13 value -2.391364
## iter  14 value -2.392533
## iter  15 value -2.394227
## iter  16 value -2.394397
## iter  17 value -2.394506
## iter  18 value -2.394547
## iter  19 value -2.394548
## iter  20 value -2.394548
## iter  21 value -2.394549
## iter  22 value -2.394550
## iter  23 value -2.394550
## iter  23 value -2.394550
```

```
## iter  23 value -2.394550
## final  value -2.394550
## converged
## initial  value -2.369136
## iter   2 value -2.372702
## iter   3 value -2.377056
## iter   4 value -2.378329
## iter   5 value -2.379099
## iter   6 value -2.379280
## iter   7 value -2.379354
## iter   8 value -2.379361
## iter   9 value -2.379369
## iter  10 value -2.379387
## iter  11 value -2.379409
## iter  12 value -2.379423
## iter  13 value -2.379426
## iter  14 value -2.379426
## iter  14 value -2.379426
## final  value -2.379426
## converged
```




```
# p-value should be bigger that 0.05
shapiro.test(res) # p-value should be bigger that 0.05
```

```
##
##  Shapiro-Wilk normality test
##
```

```
## data:  res
## W = 0.99464, p-value = 0.5489
```

```r
Box.test(res, lag = 16, type = c("Box-Pierce"), fitdf = 3)
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 8.1828, df = 13, p-value = 0.8315
```

```r
Box.test(res, lag = 16, type = c("Ljung-Box"), fitdf = 3)
```

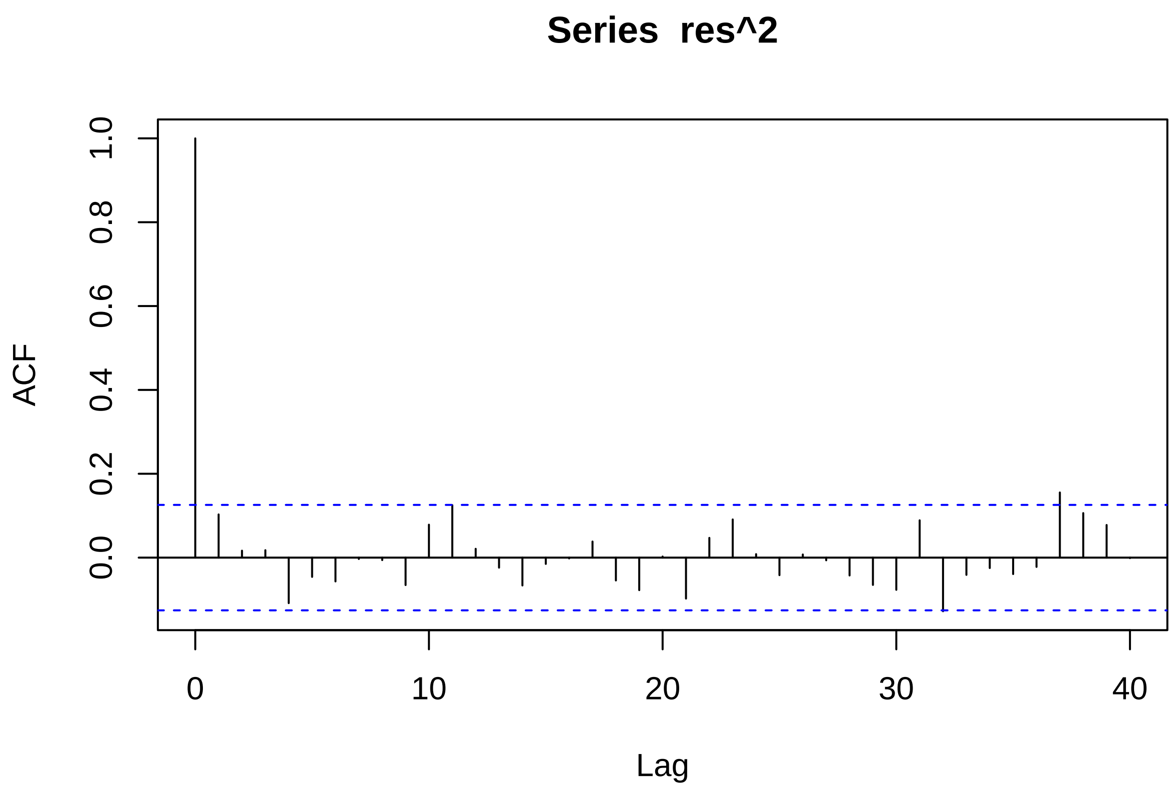
```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 8.6088, df = 13, p-value = 0.8018
```

```r
Box.test(res^2, lag = 16, type = c("Ljung-Box"), fitdf = 0)
```

```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 15.206, df = 16, p-value = 0.5096
```

All p-value is larger than 0.05.

```r
acf(res^2, lag.max=40) # do not need this
```




```r
ar(res, aic = TRUE,  order.max = NULL, method = c("yule-walker"))
```

```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  4.85e-06
```

Fitted residual to AR(0), White noise Pass Diagnostic checking. Ready to be used for forecasting.

```r
library(forecast)
```

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

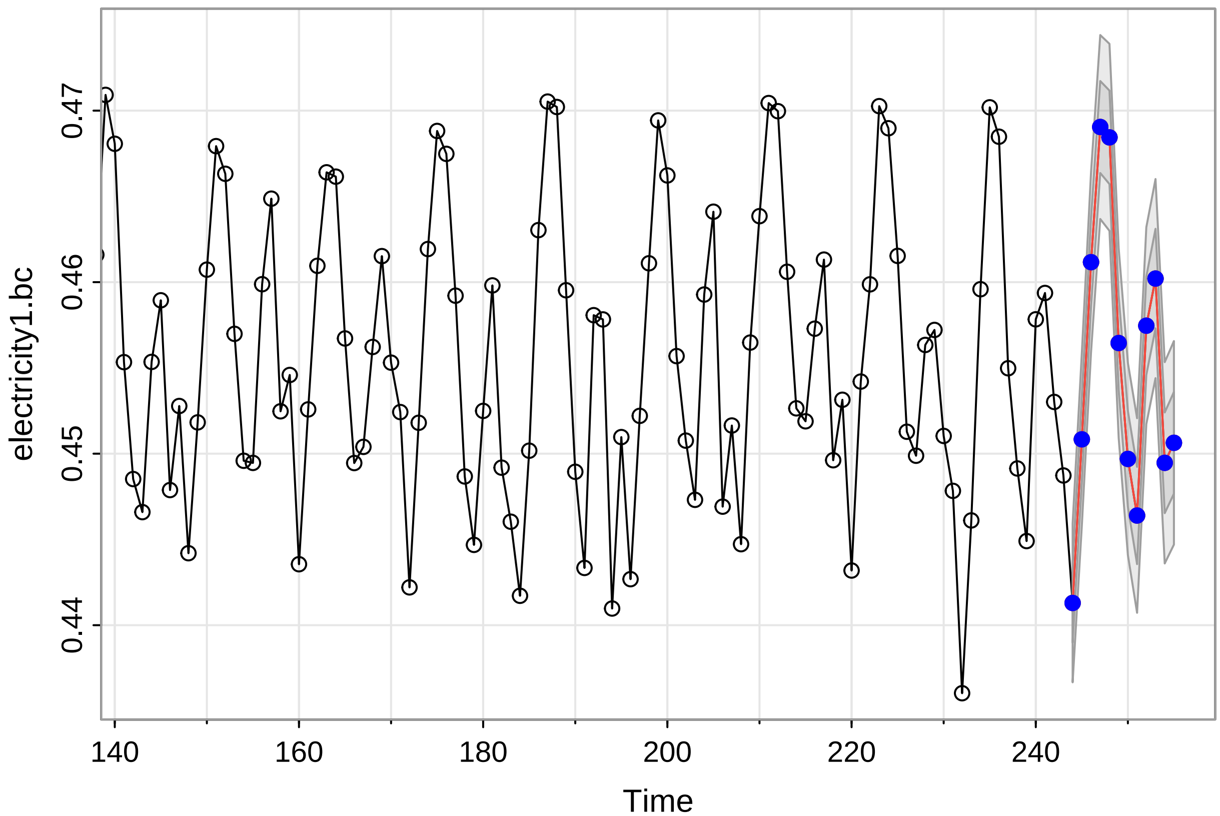
```
## Registered S3 methods overwritten by 'forecast':
##   method                 from
##   autoplot.Arima         ggfortify
##   autoplot.acf           ggfortify
##   autoplot.ar            ggfortify
##   autoplot.bats          ggfortify
##   autoplot.decomposed.ts ggfortify
##   autoplot.ets    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

```
##   fitted.ar              ggfortify
##   fortify.ts             ggfortify
##   residuals.ar           ggfortify
```

```
##
## Attaching package: 'forecast'
```

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

```
pred.tr <- sarima.for(electricity1.bc,n.ahead = 12,p=2,d=1,q=1,P=1,D=1,Q=2, S=12)
#sarima.for(electricity1.bc,n.ahead = 12,p=2,d=1,q=1,P=1,D=1,Q=2, S=12)
points(length(electricity1) + 1:length(electricity1_test),pred.tr$pred, col="blue",pch = 19)
```




```
#pred.orig <- InvBoxCox(pred.tr$pred, lambda)
#sarima.for(electricity1,n.ahead = 12,p=2,d=1,q=1,P=1,D=1,Q=2, S=12)
#ts.plot(electricity1, xlim=c(1,length(electricity1)+12))
#points(length(electricity1) + 1:length(electricity1_test),electricity1_test, col="blue",pch = 19)
```

```
# arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,2), period = 12), fixed = c(NA

# Forecasting using model A:
fit.A <- arima(electricity1.bc, order = c(2,1,1), seasonal = list(order = c(1,1,2), period = 12), fixed
```

```
## Warning in arima(electricity1.bc, order = c(2, 1, 1), seasonal = list(order =
## c(1, : some AR parameters were fixed: setting transform.pars = FALSE
```

```r
forecast(fit.A)
```

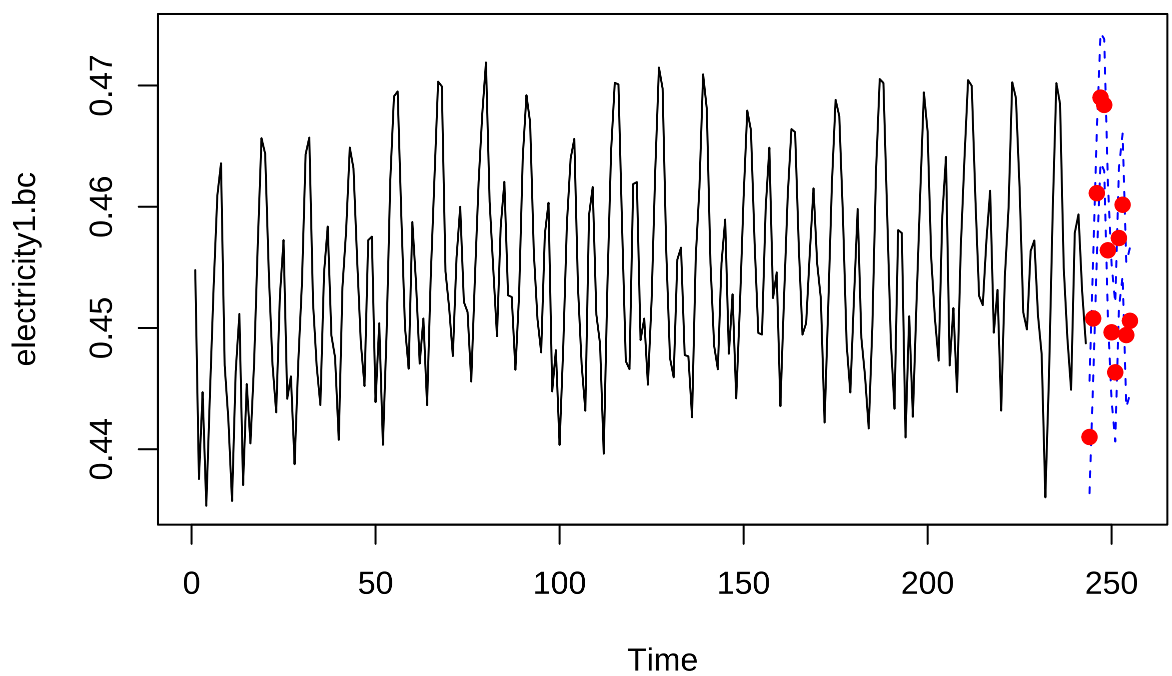
```
## Warning in predict.Arima(object, n.ahead = h): seasonal MA part of model is not
## invertible

##     Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
## 244      0.4410098 0.4380394 0.4439801 0.4364670 0.4455525
## 245      0.4507861 0.4475395 0.4540327 0.4458209 0.4557514
## 246      0.4611176 0.4577664 0.4644688 0.4559924 0.4662428
## 247      0.4689993 0.4655775 0.4724212 0.4637661 0.4742326
## 248      0.4683948 0.4649114 0.4718782 0.4630674 0.4737222
## 249      0.4564071 0.4528654 0.4599489 0.4509905 0.4618237
## 250      0.4496435 0.4460449 0.4532420 0.4441399 0.4551470
## 251      0.4463421 0.4426877 0.4499966 0.4407532 0.4519311
## 252      0.4574193 0.4537097 0.4611288 0.4517460 0.4630925
## 253      0.4601636 0.4564002 0.4639269 0.4544080 0.4659191
## 254      0.4494165 0.4456000 0.4532330 0.4435797 0.4552534
## 255      0.4505900 0.4467211 0.4544589 0.4446730 0.4565070
## 256      0.4430399 0.4389421 0.4471377 0.4367729 0.4493070
## 257      0.4526111 0.4484045 0.4568177 0.4461776 0.4590445
## 258      0.4615669 0.4572783 0.4658555 0.4550080 0.4681257
## 259      0.4690589 0.4646961 0.4734218 0.4623866 0.4757313
## 260      0.4686045 0.4641704 0.4730385 0.4618232 0.4753858
## 261      0.4574720 0.4529684 0.4619757 0.4505843 0.4643598
## 262      0.4501620 0.4455899 0.4547342 0.4431695 0.4571545
## 263      0.4472239 0.4425842 0.4518636 0.4401281 0.4543197
## 264      0.4575117 0.4528051 0.4622184 0.4503136 0.4647099
## 265      0.4603917 0.4556190 0.4651643 0.4530925 0.4676908
## 266      0.4489297 0.4440928 0.4537666 0.4415323 0.4563271
## 267      0.4509546 0.4460544 0.4558548 0.4434603 0.4584488
```

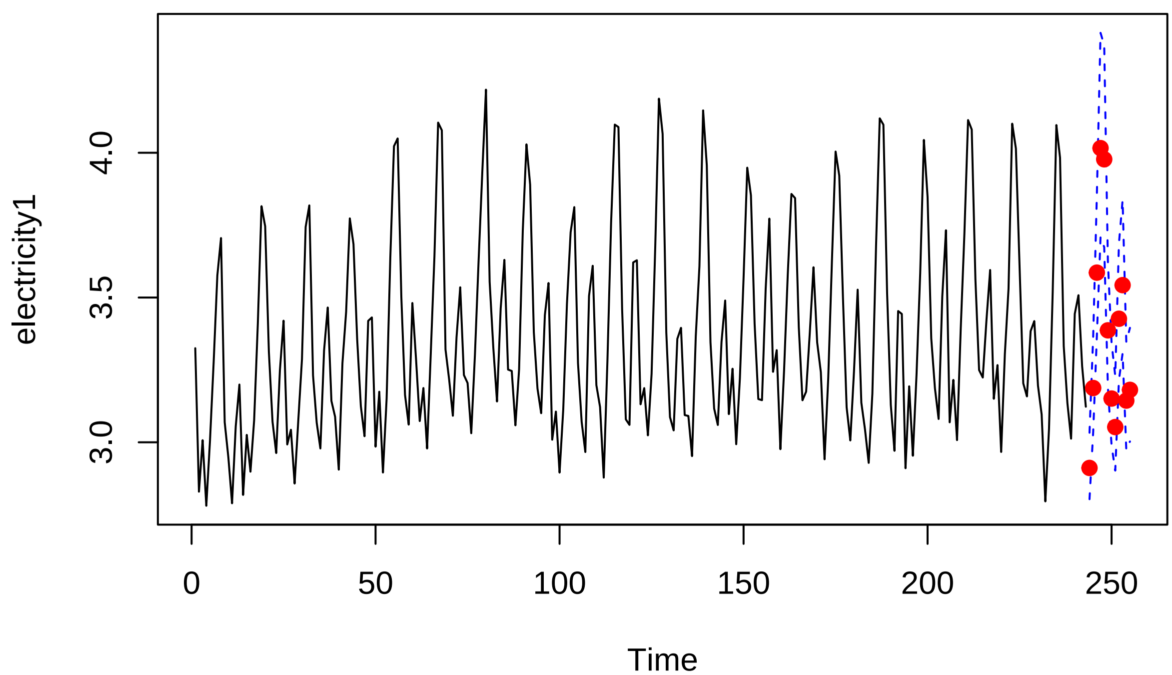
```r
# To produce graph with 12 forecasts on transformed data:
pred.tr1 <- predict(fit.A, n.ahead = 12)
```

```
## Warning in predict.Arima(fit.A, n.ahead = 12): seasonal MA part of model is not
## invertible
```

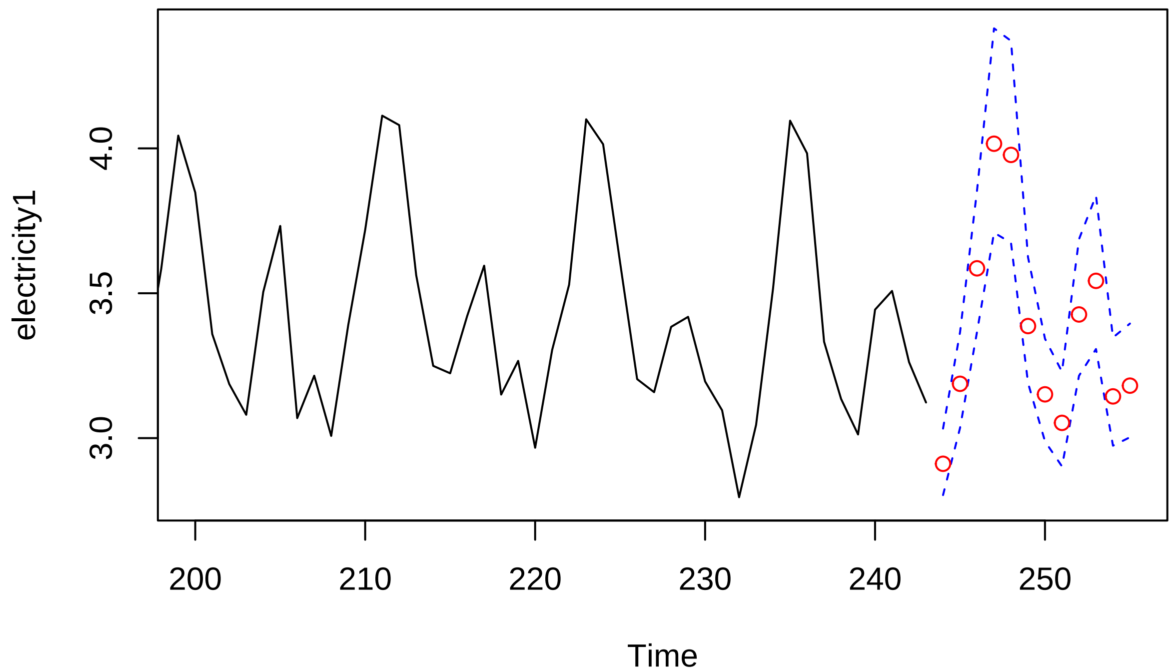
```r
U.tr = pred.tr1$pred + 2*pred.tr1$se # upper bound of the prediction interval
L.tr = pred.tr1$pred - 2*pred.tr1$se # lower bound
plot.ts(electricity1.bc, xlim=c(1,length(electricity1.bc)+12), ylim = c(min(electricity1.bc), max(U.tr)
lines(U.tr, col="blue",lty="dashed")
lines(L.tr, col="blue",lty="dashed")
points((length(electricity1.bc)+1):(length(electricity1.bc)+12), pred.tr1$pred, col="red",pch = 19)
```



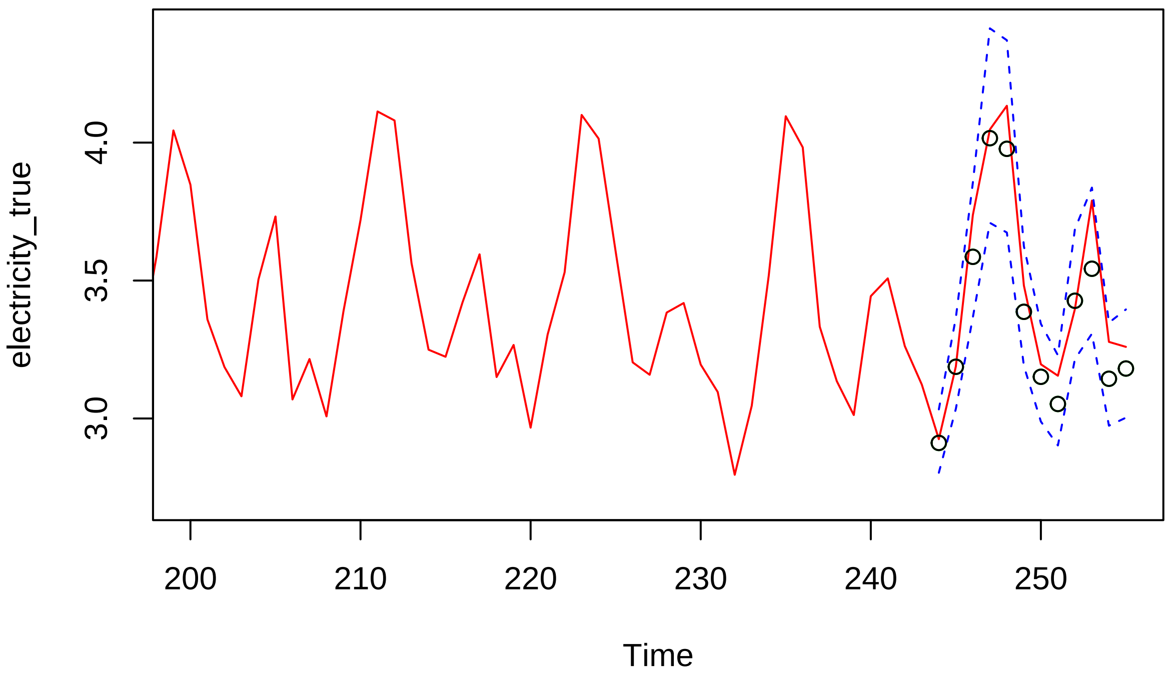

```r
# To produce graph with forecasts on original data:
pred.orig <- InvBoxCox(pred.tr1$pred, lambda)
U= InvBoxCox(U.tr, lambda)
L= InvBoxCox(L.tr, lambda)
plot.ts(electricity1, xlim=c(1,length(electricity1)+12), ylim = c(min(electricity1),max(U)))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(electricity1)+1):(length(electricity1)+12), pred.orig, col="red",pch = 19)
```




```r
# To zoom the graph, starting from entry 200
ts.plot(electricity1, xlim = c(200,length(electricity1)+12), ylim = c(min(electricity1),max(U)))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(electricity1)+1):(length(electricity1)+12), pred.orig, col="red")
```




```
# To plot zoomed forecasts and true values (in electricity):
electricity_true <- electricity[1:255]/100000
plot.ts(electricity_true, xlim = c(200,length(electricity1)+12), ylim = c(2.7,max(U)), col="red")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(electricity1)+1):(length(electricity1)+12), pred.orig, col="green")
points((length(electricity1)+1):(length(electricity1)+12), pred.orig, col="black")
```


electricity_true
3.0
3.5
4.0
200
210
220
230
240
250
Time