Test

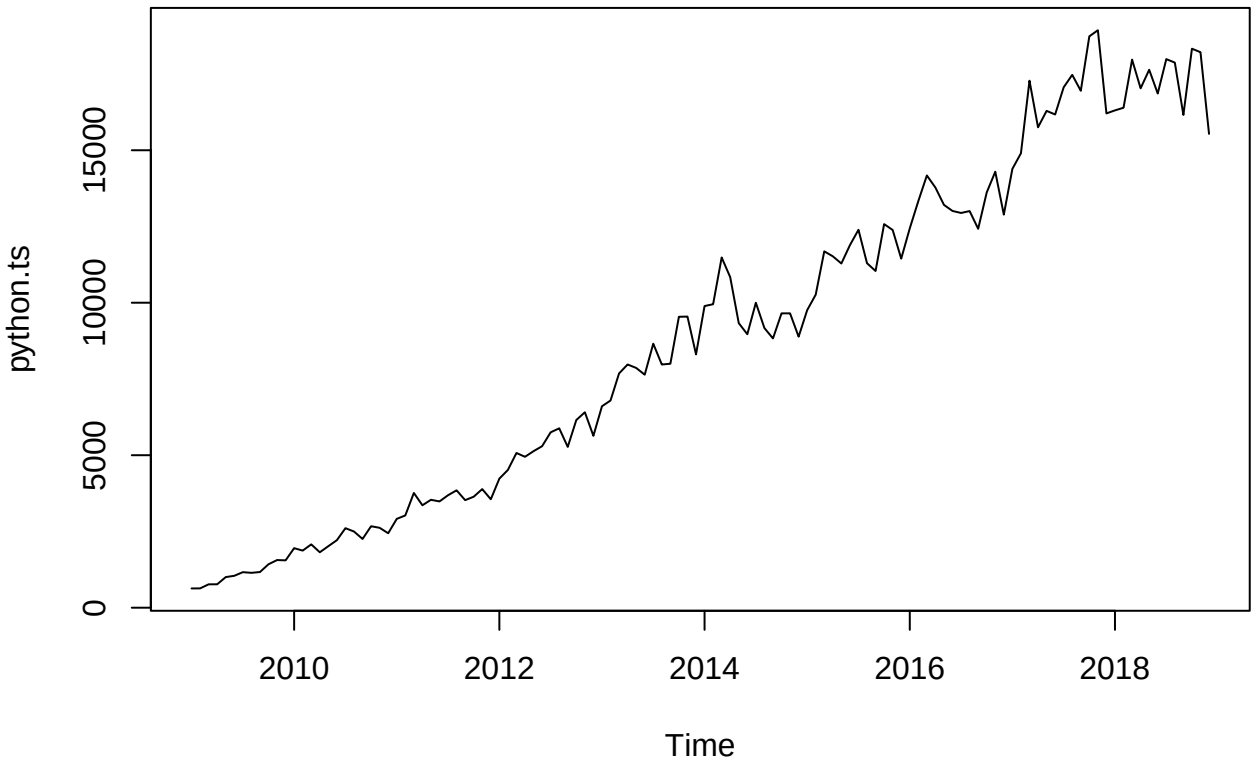
Kathy Wu

1/16/2022

```
counts = read.csv("MLTollsStackOverflow.csv")
python = counts[,5]
```

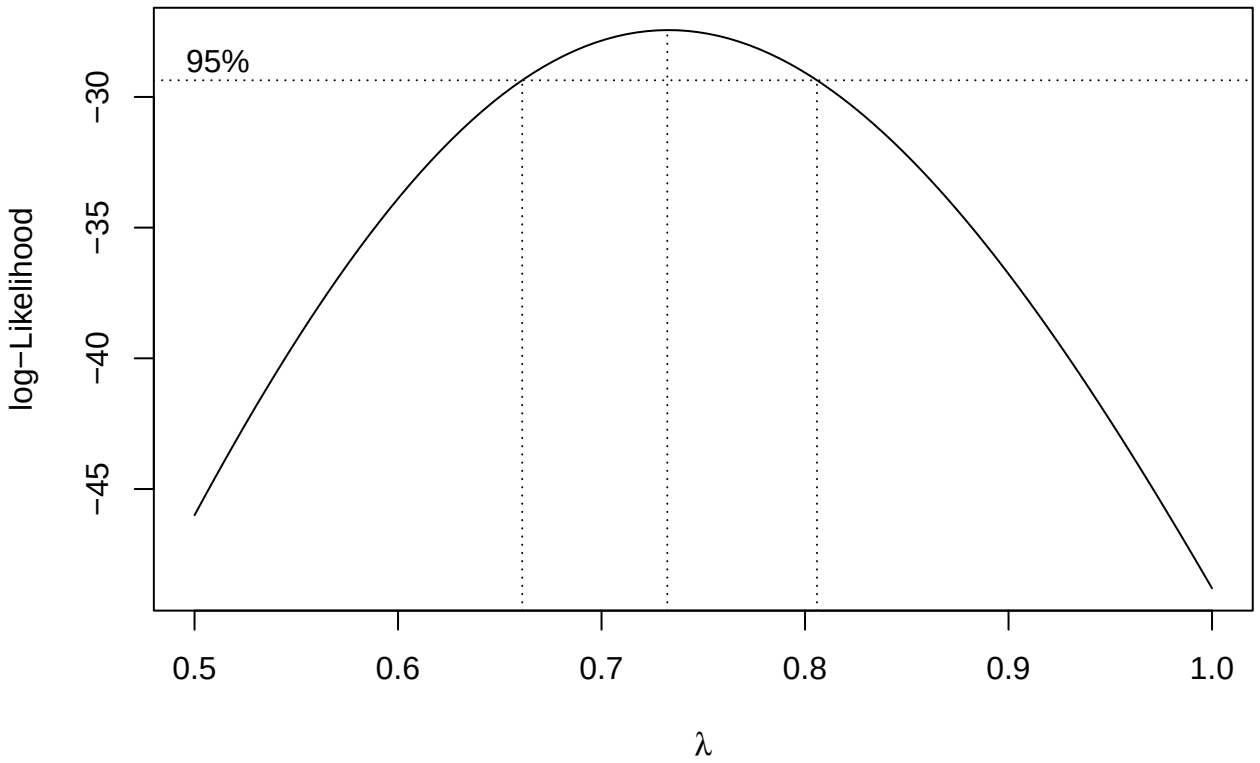
```
python.tr=python[1:120]
python.te=python[121:132]
```

```
python.ts = ts(python.tr, start=c(2009,1), frequency=12)
ts.plot(python.ts)
```



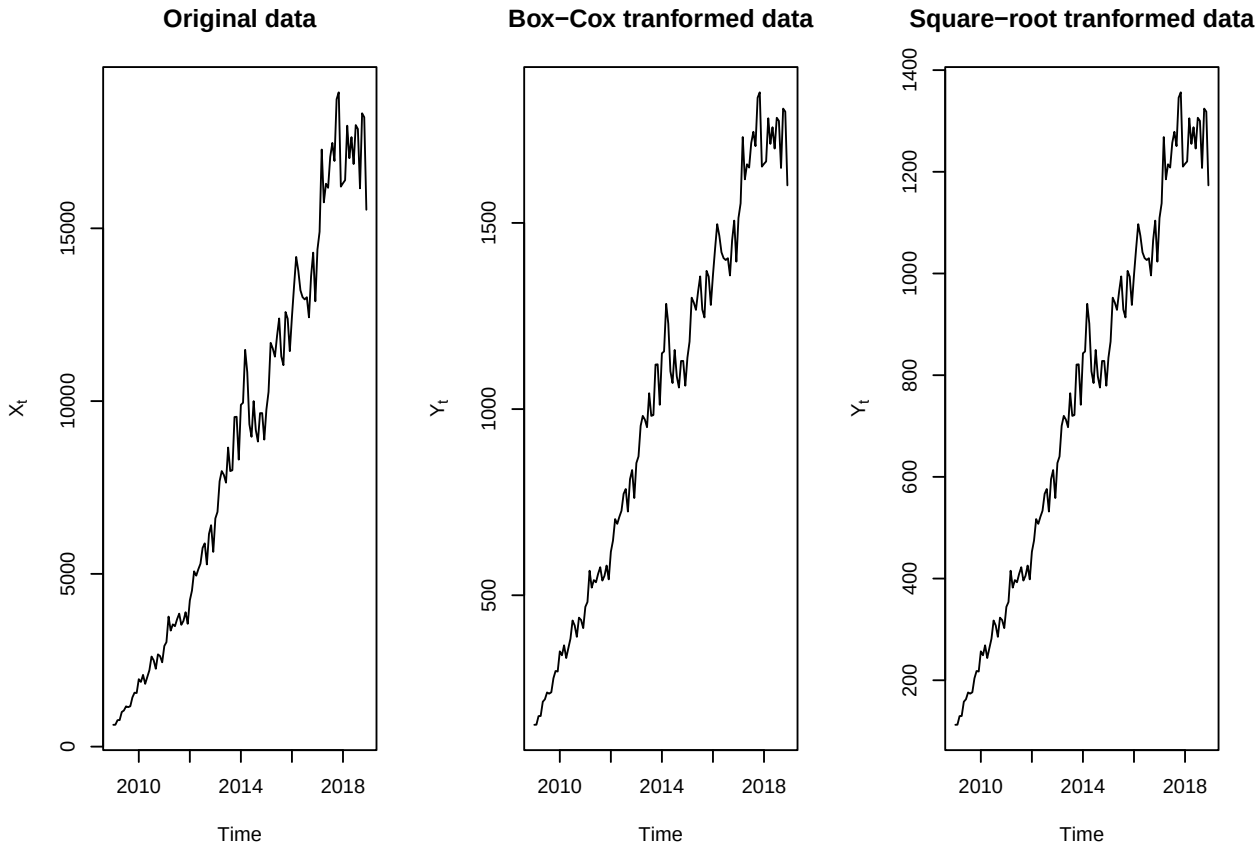
Transformation

```
library(MASS)
t = 1:length(python.ts)
fit = lm(python.ts ~ t)
bcTransform = boxcox(python.ts ~ t,plotit = TRUE, lambda = seq(0.5, 1, 0.01))
```




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

```
python.lam=(python.ts)^(lambda)
# compare the original and boxcox transform
op <- par(mfrow = c(1,3))
ts.plot(python.ts, main = "Original data", ylab = expression(X[t]))
ts.plot(python.bc, main = "Box-Cox tranformed data", ylab = expression(Y[t]))
ts.plot(python.lam, main = "Square-root tranformed data", ylab = expression(Y[t]))
```




```
var(python.ts)
```

```
## [1] 31122294
```

```
var(python.bc)
```

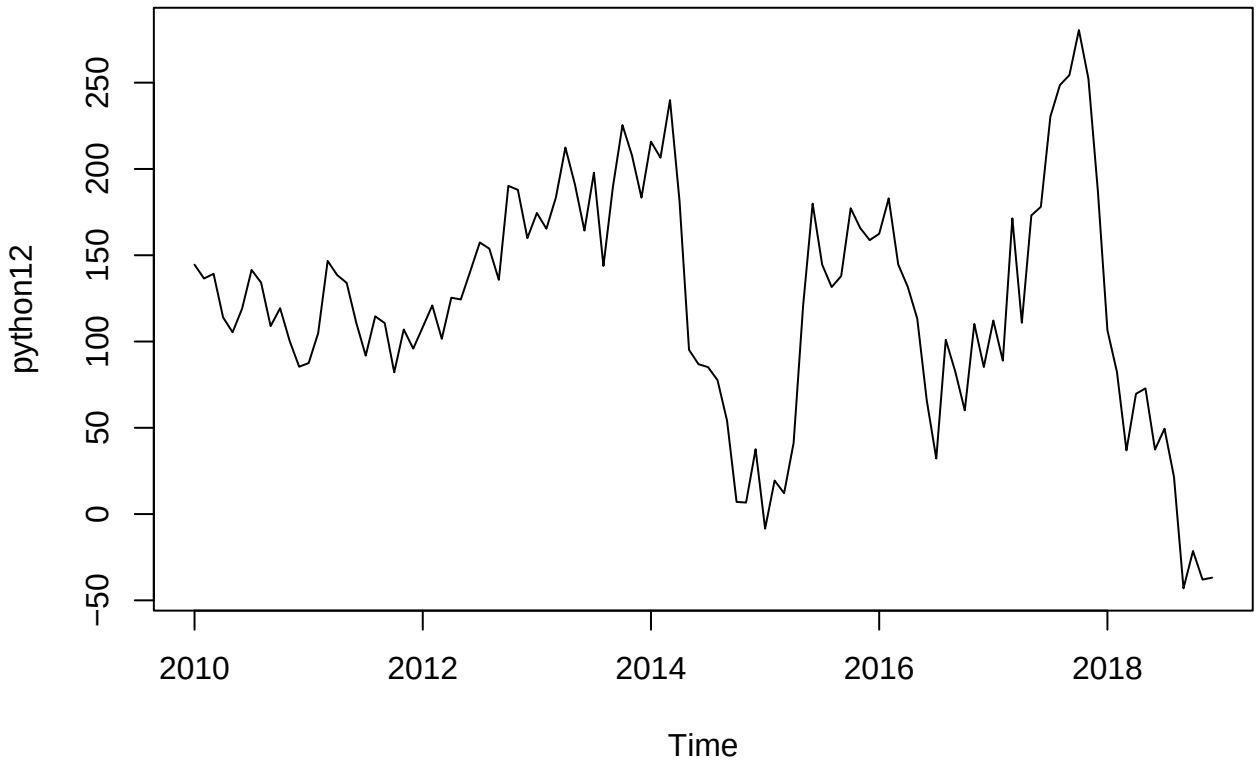
```
## [1] 259399
```

```
var(python.lam)
```

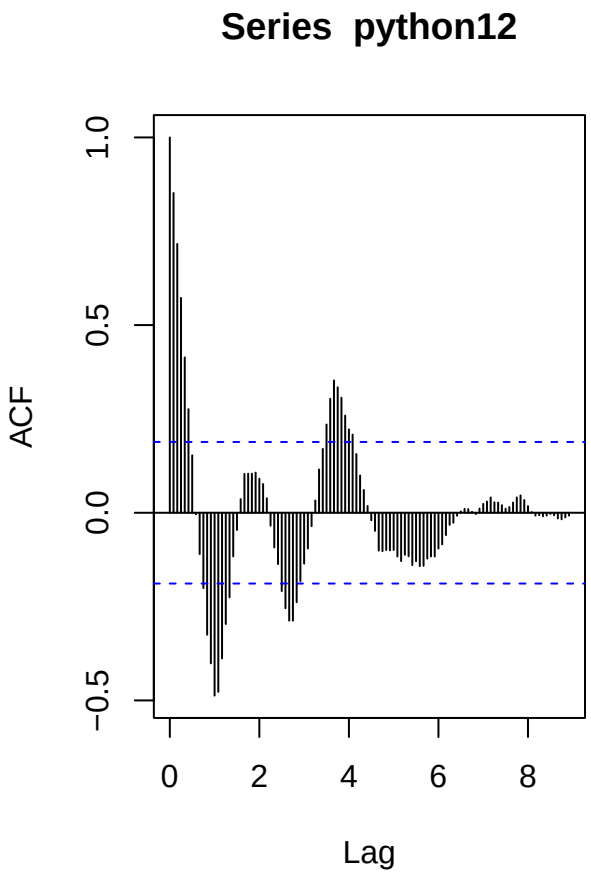
```
## [1] 139115
```

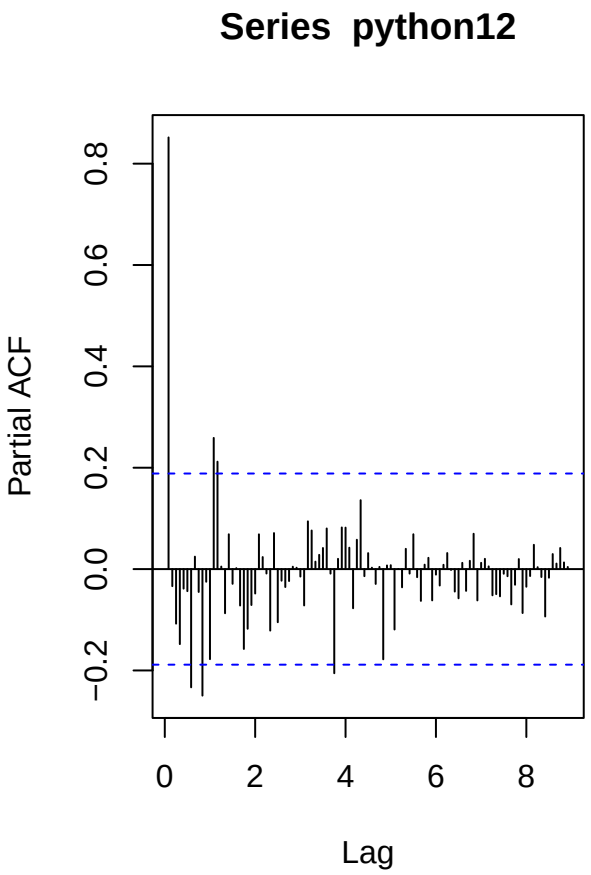
De-seasonalize first, then de-trend

```
python12=diff(python.lam, 12)
plot(python12)
```

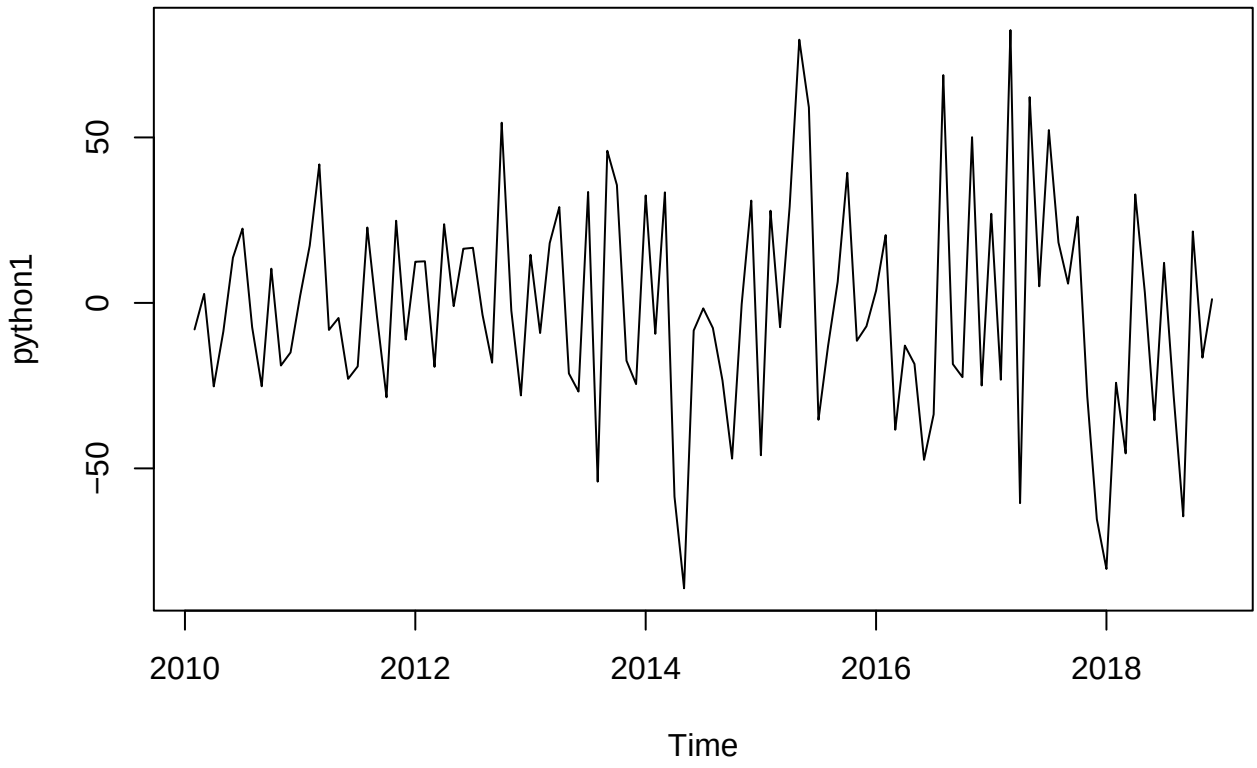



```
# plot the acf/pacf for deseasonalize dataset
par(mfrow=c(1,2))
acf(python12, lag.max=120)
pacf(python12, lag.max=120)
```

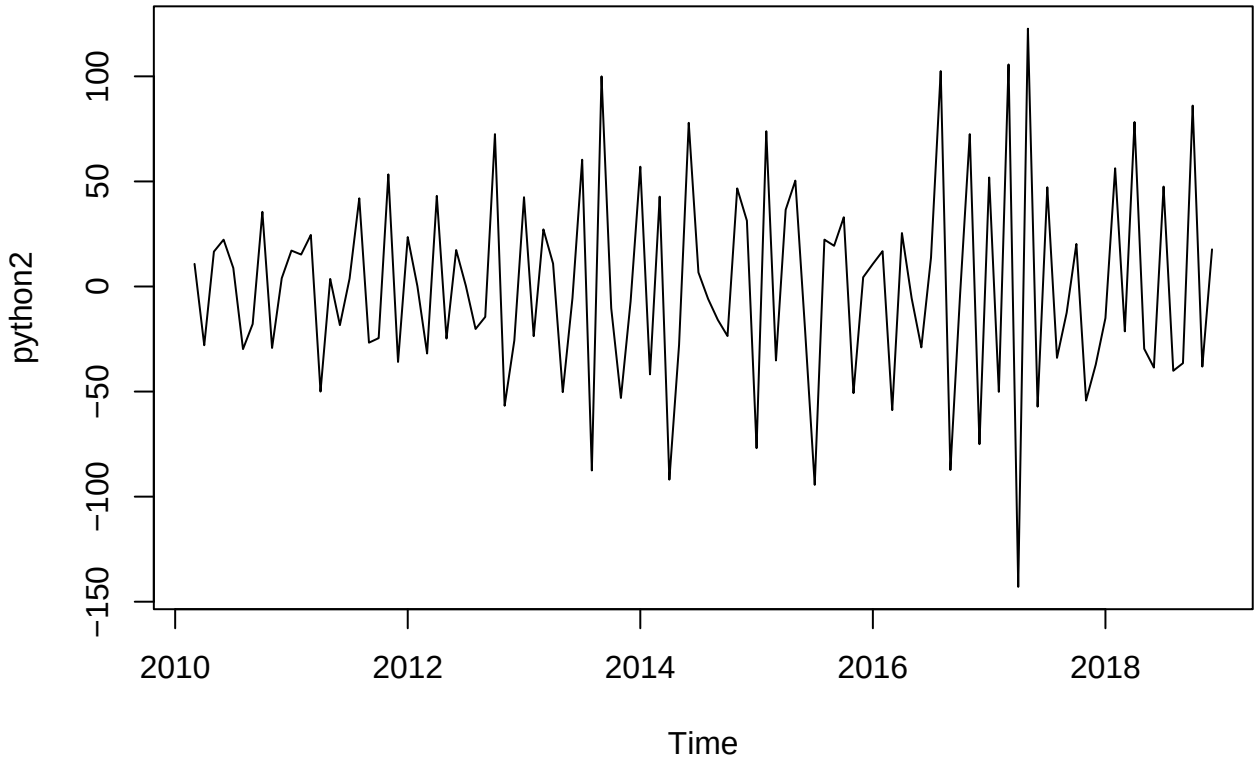





```
# de-trend the data once
python1 = diff(python12,1)
plot(python1)
```




```
# de-trend the data again
python2 = diff(python1, 1)
plot(python2)
```




```
# Check the variance
var(python.lam)
```

```
## [1] 139115
```

```
var(python12) # de-seasonal data
```

```
## [1] 4433
```

```
var(python1) # deseasonal first, then trend
```

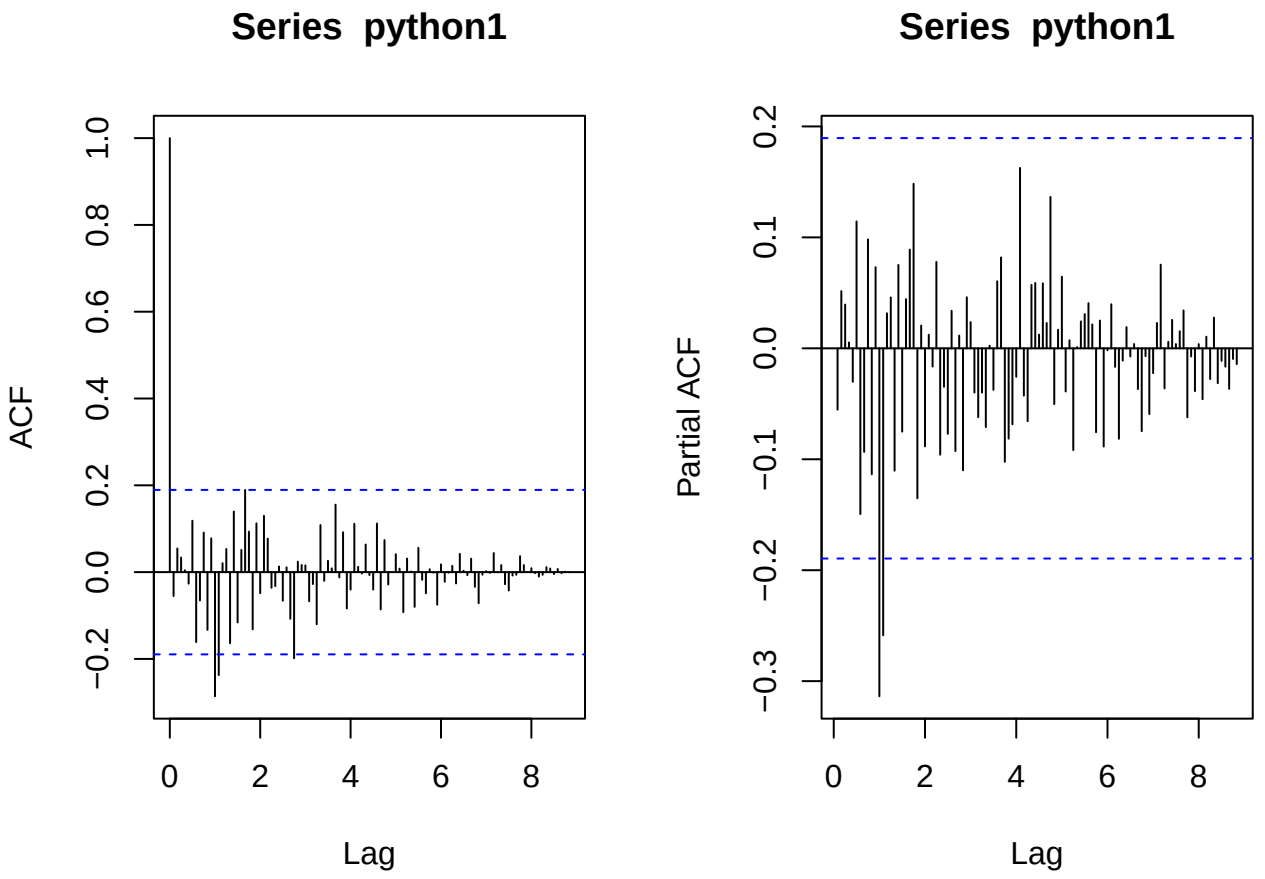
```
## [1] 1081
```

```
var(python2)
```

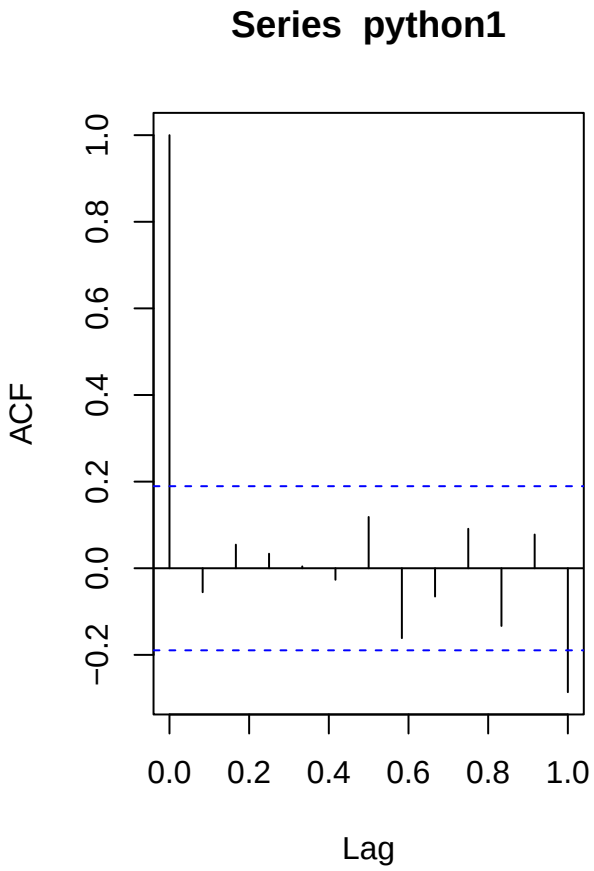
```
## [1] 2302
```

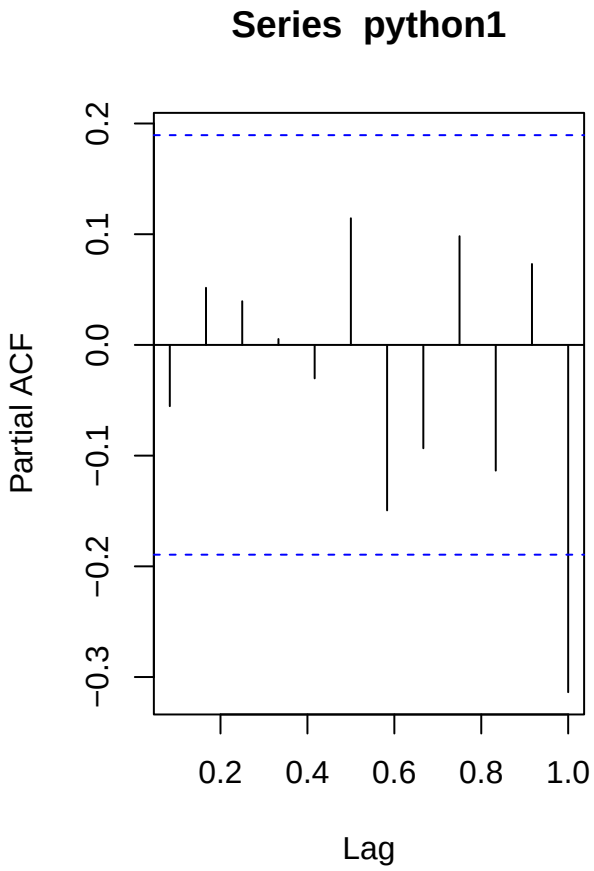
```
# increasing variance for second de-trend
# d=1, D=1
```

```
par(mfrow=c(1,2))
acf(python1, lag.max=120)
pacf(python1, lag.max=120)
```




```
par(mfrow=c(1,2))
acf(python1, lag.max=12)
pacf(python1, lag.max=12)
```





Fit the model

```
fit.coef=arima(python1, order=c(13,1,13),
               fixed=c(rep(0,11),NA,NA,NA,NA,rep(0,9),NA,NA),
               seasonal = list(order = c(0,1,0), period = 12), method="ML")
```

```
## Warning in arima(python1, order = c(13, 1, 13), fixed = c(rep(0, 11), NA, : some
## AR parameters were fixed: setting transform.pars = FALSE
```

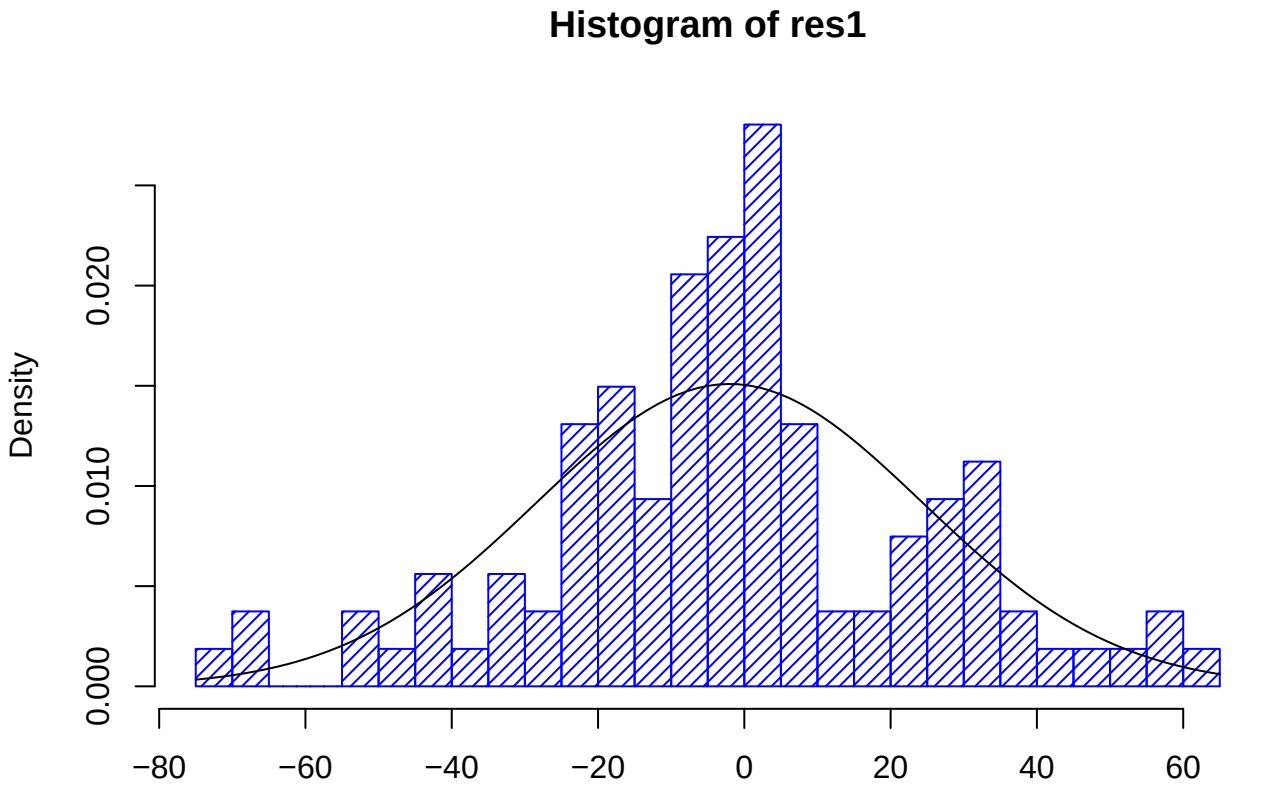
```
fit.coef
```

```
##
## Call:
## arima(x = python1, order = c(13, 1, 13), seasonal = list(order = c(0, 1, 0),
##     period = 12), fixed = c(rep(0, 11), NA, NA, NA, NA, rep(0, 9), NA, NA),
##     method = "ML")
##
## Coefficients:
##       ar1  ar2  ar3  ar4  ar5  ar6  ar7  ar8  ar9  ar10  ar11    ar12    ar13
##         0    0    0    0    0    0    0    0    0     0     0  -0.361  -0.255
## s.e.    0    0    0    0    0    0    0    0    0     0     0   0.096   0.093
##          ma1    ma2  ma3  ma4  ma5  ma6  ma7  ma8  ma9  ma10  ma11    ma12
##       -1.200  0.234    0    0    0    0    0    0    0     0     0  -1.059
## s.e.   0.121  0.119    0    0    0    0    0    0    0     0     0   0.155
##        ma13
##       1.040
## s.e.  0.153
##
```

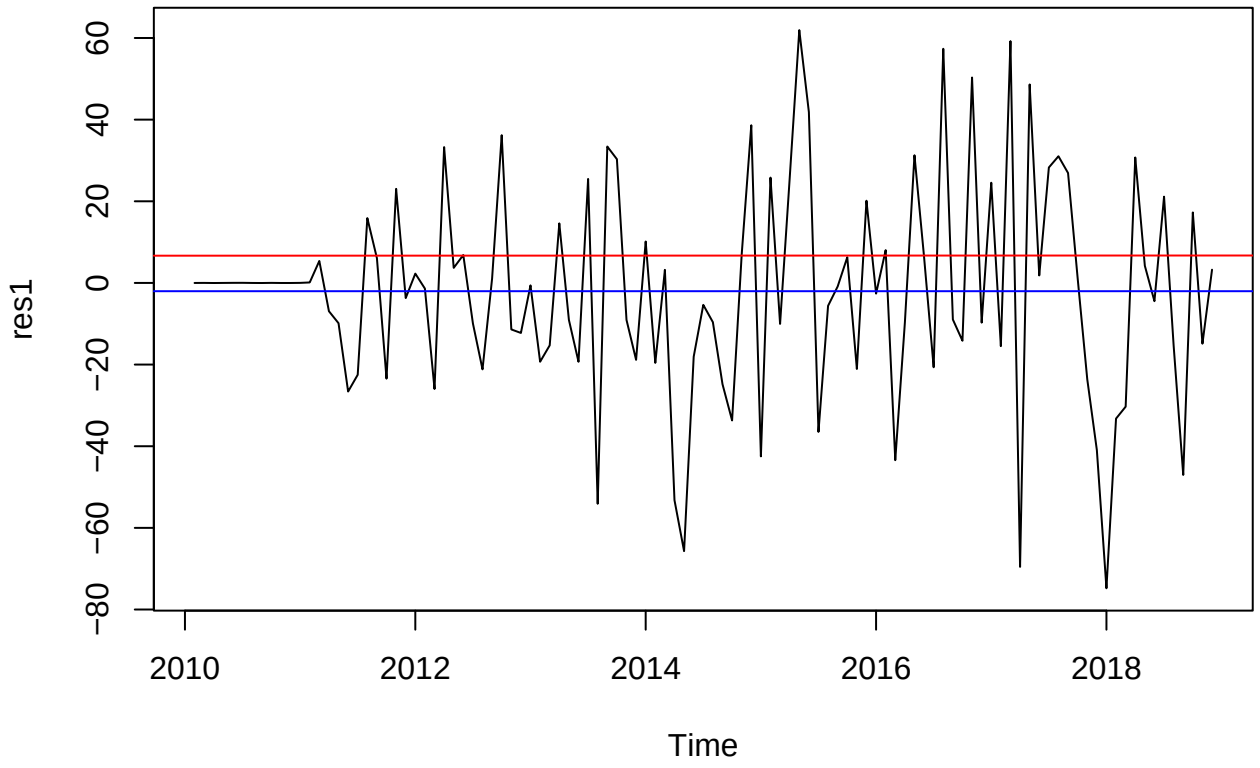
```
## sigma^2 estimated as 793:  log likelihood = -471.5,  aic = 957
```

```
# calculate the residual
res1=residuals(fit.coef)
#res2=residuals(fit2)
```

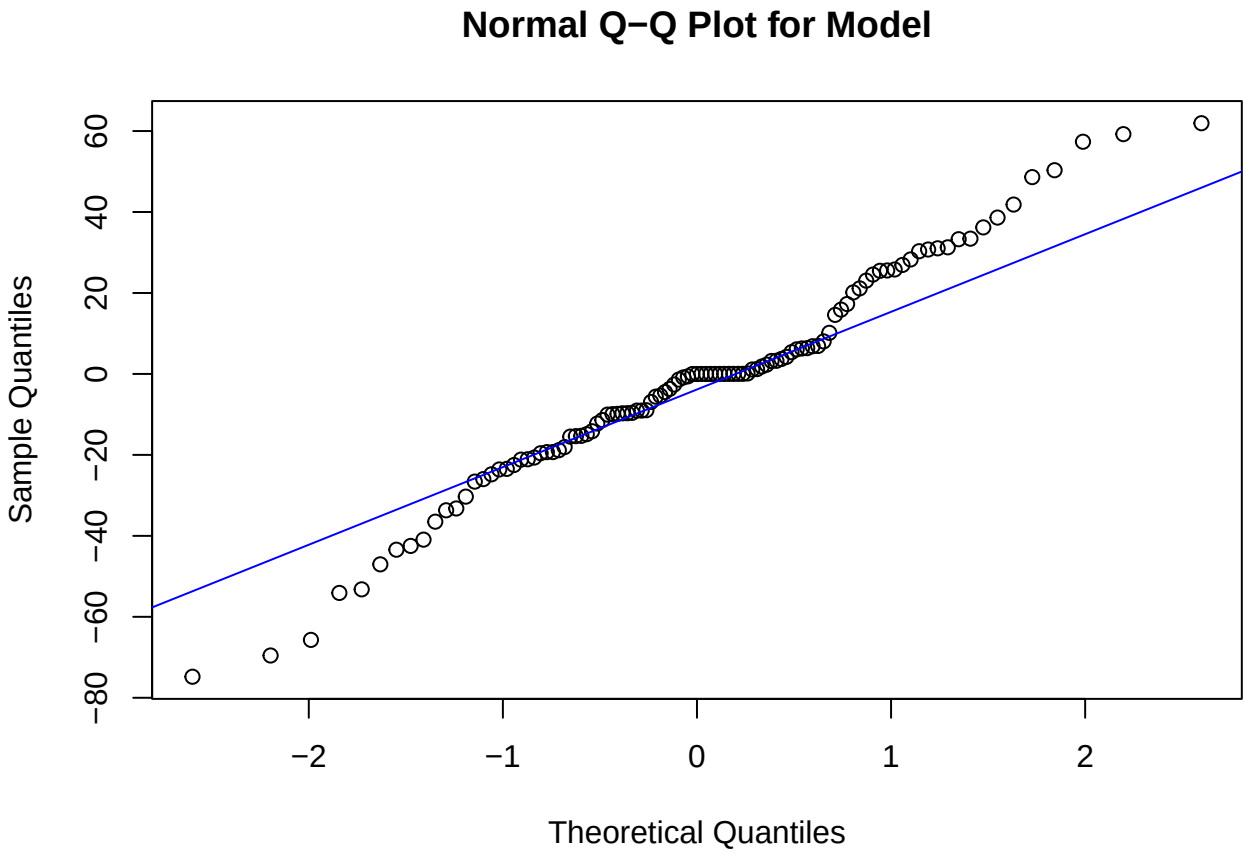
```
# plot the histogram of the residual
hist(res1,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m=mean(res1)
std=sqrt(var(res1))
curve(dnorm(x,m,std), add=TRUE)
```



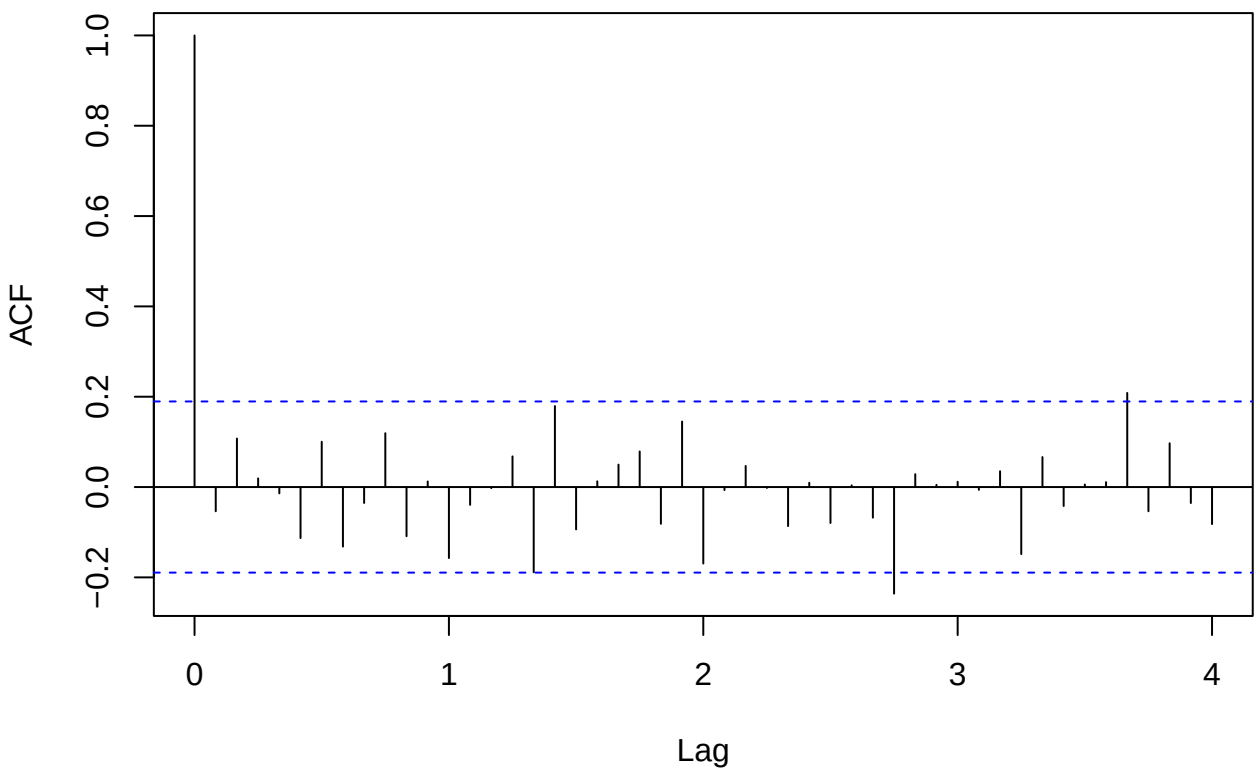

```
# check if the residual performs white noise
plot.ts(res1)
fitt <- lm(res1 ~ as.numeric(1:length(res1))); abline(fitt, col="red")
abline(h=mean(res1), col="blue")
```



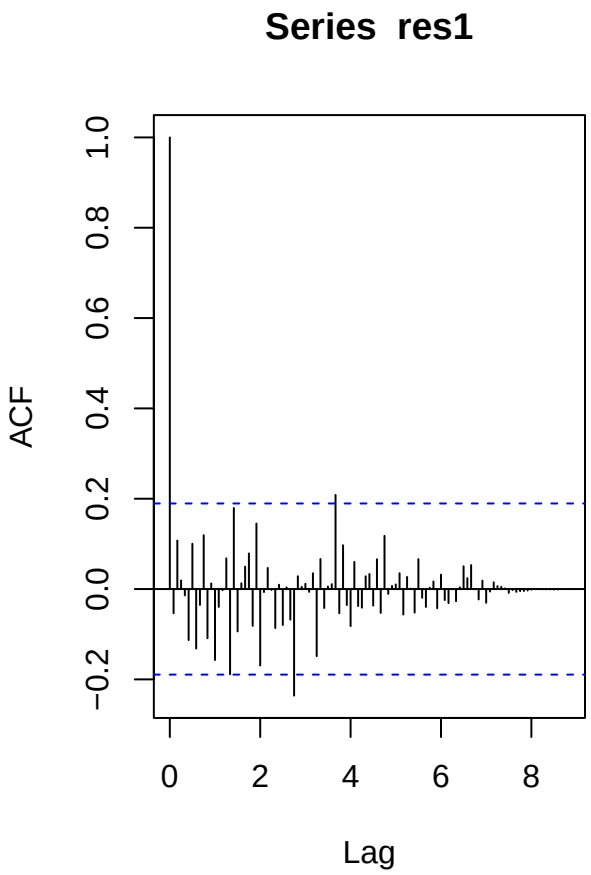

```
# Q-Q plot
qqnorm(res1,main= "Normal Q-Q Plot for Model")
qqline(res1,col="blue")
```

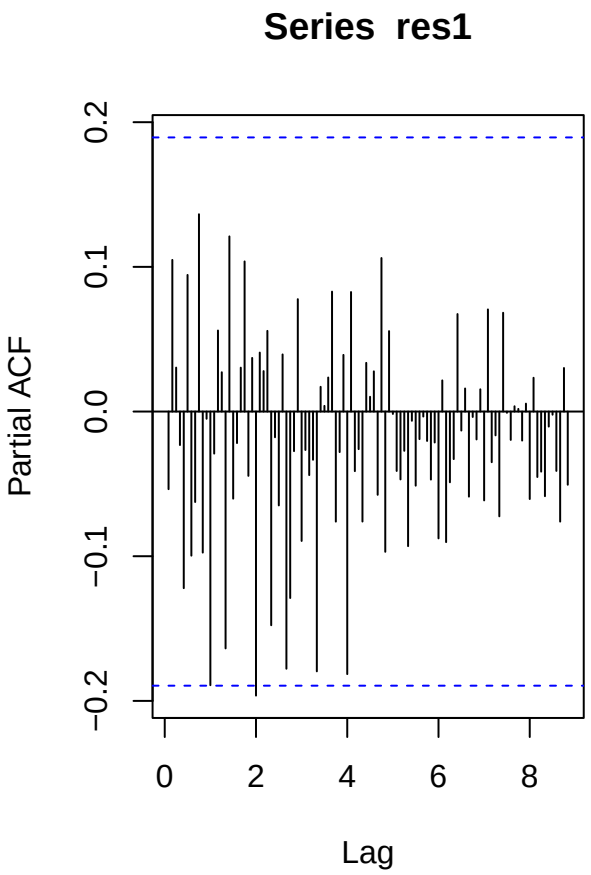



```
# plot the acf/pacf, small p and q
#par(mfrow=c(1,2))
acf(res1, lag.max=48)
```




```
#pacf(res1, lag.max=24)
# plot the acf/pacf, big P and Q
par(mfrow=c(1,2))
acf(res1, lag.max=120)
pacf(res1, lag.max=120)
```






```
# test if residual is normally distributed
shapiro.test(res1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res1
## W = 0.98, p-value = 0.09
```

```
# Box test
# lag= sqrt(number of observations)
Box.test(res1, lag = 11, type = c("Box-Pierce"), fitdf = 6)
```

```
##
##  Box-Pierce test
##
## data:  res1
## X-squared = 8.8, df = 5, p-value = 0.1
```

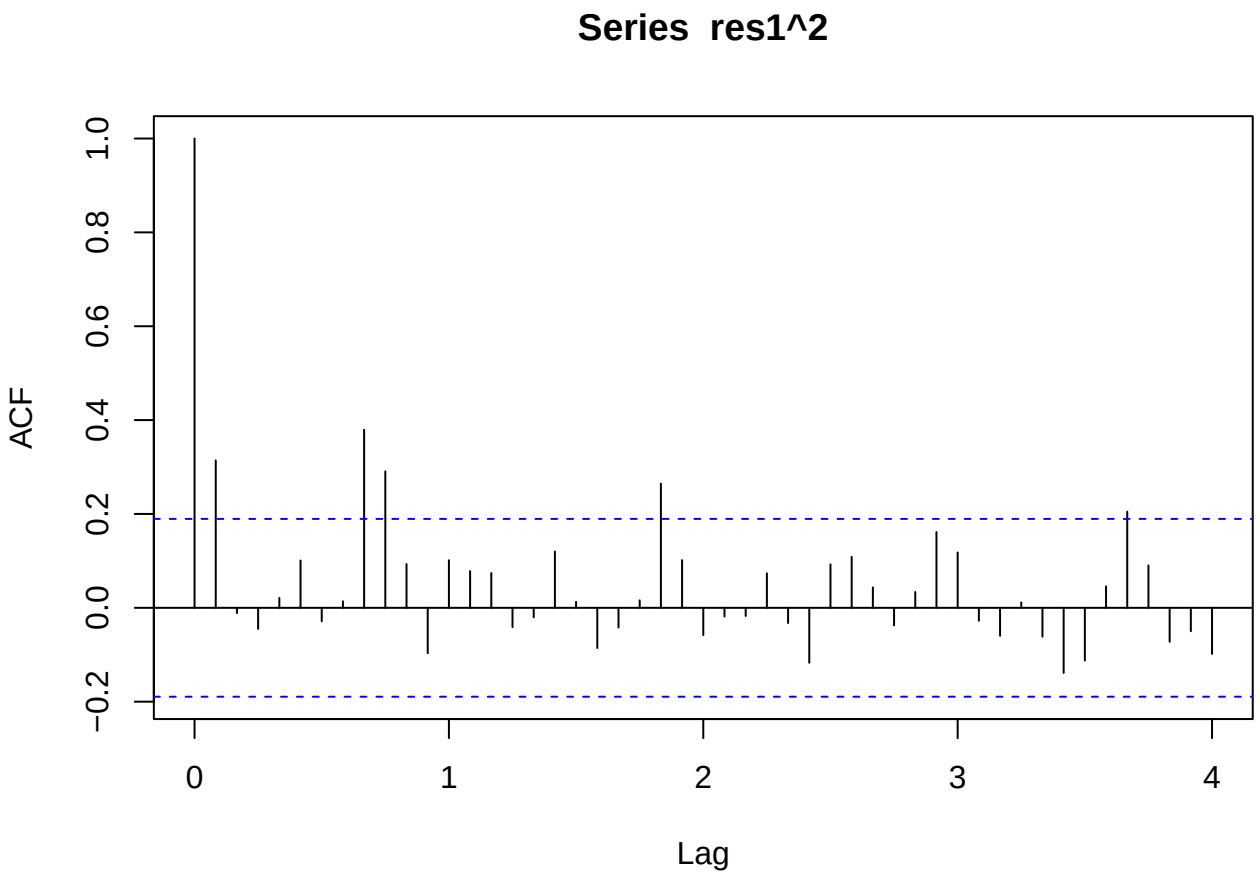
```
Box.test(res1, lag = 11, type = c("Ljung-Box"), fitdf = 6)
```

```
##
##  Box-Ljung test
##
## data:  res1
## X-squared = 9.6, df = 5, p-value = 0.09
```

```
Box.test(res1^2, lag = 11, type = c("Ljung-Box"), fitdf = 0)
```

```
##
##  Box-Ljung test
##
## data:  res1^2
## X-squared = 42, df = 11, p-value = 2e-05
```

```
acf(res1^2, lag.max=48)
```




```
#pacf(res1^2, lag.max =48)
ar(res1, aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

```
##
## Call:
## ar(x = res1, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  699
```

Forecast