443 Project

2022-11-06

Data Cleaning and Preprocessing

```
df <- read.csv("Data_Group11.csv")
```

We first convert the data in to a more usable long format.

```
# convert month and day to MM-DD format
singledig <- which(df$X.1 < 10)
df$day <- as.character(df$X.1)
df$day[singledig] <- paste0("0", df$day[singledig])

df$month <- rep(c("01","02","03","04", "05","06","07","08","09","10","11","12"), times=c(31,29,31,30,31

df$MMDD <- paste0(df$month, "-", df$day)

# drop unnecessary columns
to_drop <- c("X", "X.1", "X.2", "X1981.2010.mean", "X1981.2010.median", "month", "day")
df <- subset(df, select=!(names(df) %in% to_drop))

# melt years to be one column
library(tidyr)
df <- pivot_longer(data=df, cols=!MMDD, names_to="year", values_to="extent")

# format year and creating a column with YY-MM-DD format
library(stringr)
df$year = str_replace(df$year, "X", "")
df$YYMMDD <- paste0(df$year, "-", df$MMDD)

# tell R YYMMDD is a date
df$YYMMDD = as.Date(df$YYMMDD) # also conveniently rids of non leap year Feb 29's

# drop unnecessary columns
to_drop <- c("MMDD", "year")
df <- subset(df, select=!(names(df) %in% to_drop))

# order data by date
df <- df[order(df$YYMMDD),]

head(df)
```

```
## # A tibble: 6 x 2
##   extent YYMMDD
##    <dbl> <date>
## 1     NA 1978-01-01
## 2     NA 1978-01-02
```

```
## 3     NA 1978-01-03
## 4     NA 1978-01-04
## 5     NA 1978-01-05
## 6     NA 1978-01-06
```

We then deal with NA values.

```
# drop initial and ending NAs because we don't have data collected for these dates
library(zoo)
```

```
##
## Attaching package: 'zoo'

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

```
df <- na.trim(df)
```

There seems to be a change point at 1988-01-13 where a new method of measurement may of been put in to place. Before this date, there is a long stretch of NA values, and all measurements previously were recorded every second day.

```
# There is a period between 1987-12-03 and 1988-01-12 with a stretch of NAs. The following allows this
df_subset <- subset(df, YYMMDD>as.Date("1987-12-01") & YYMMDD<as.Date("1988-01-15"))
print(df_subset)
```

```
## # A tibble: 44 x 2
##    extent YYMMDD
##     <dbl> <date>
##  1   12.6 1987-12-02
##  2   NA   1987-12-03
##  3   NA   1987-12-04
##  4   NA   1987-12-05
##  5   NA   1987-12-06
##  6   NA   1987-12-07
##  7   NA   1987-12-08
##  8   NA   1987-12-09
##  9   NA   1987-12-10
## 10   NA   1987-12-11
## # ... with 34 more rows
```

To deal with this long stretch of NAs and the NA values caused by measurement every second day, we propose two options. Either, we impute the missing data using cubic splines, or drop all observations before the possible changepoint.

```
# impute missing values using cubic splines as one option
df_imputed <- df
df_imputed$extent <- na.spline(df_imputed$extent)

# drop all data before 1988-01-13 as another option
df <- subset(df, YYMMDD>=as.Date("1988-01-13"))
```

We are interested in overall trend, not day to day fluctuations, so we consider aggregating values by month.

```
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
```

```
library(lubridate)
```

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

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

```
df_aggregated <- df %>%
  group_by(year=year(YYMMDD), month=month(YYMMDD)) %>%
  mutate(avg_extent = mean(extent)) %>%
  distinct(year, month, .keep_all=TRUE) %>%
  subset(select=c(year, month, avg_extent))
```

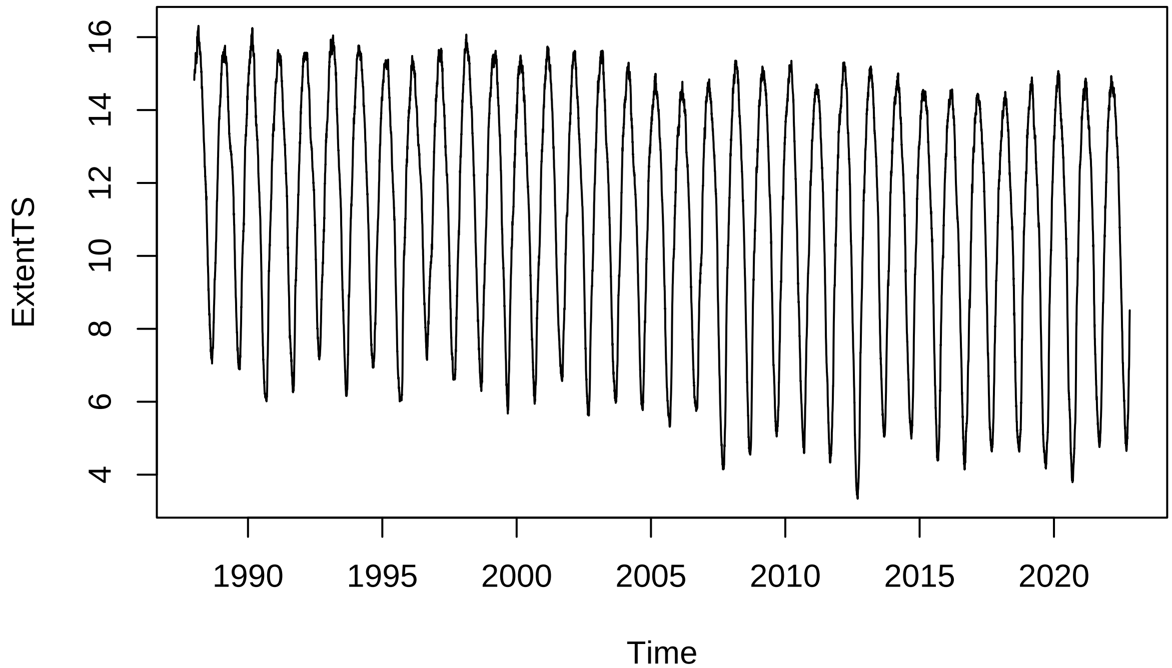
Now that our data is cleaned and processed, we may proceed with analysis.

Unaggregated Data

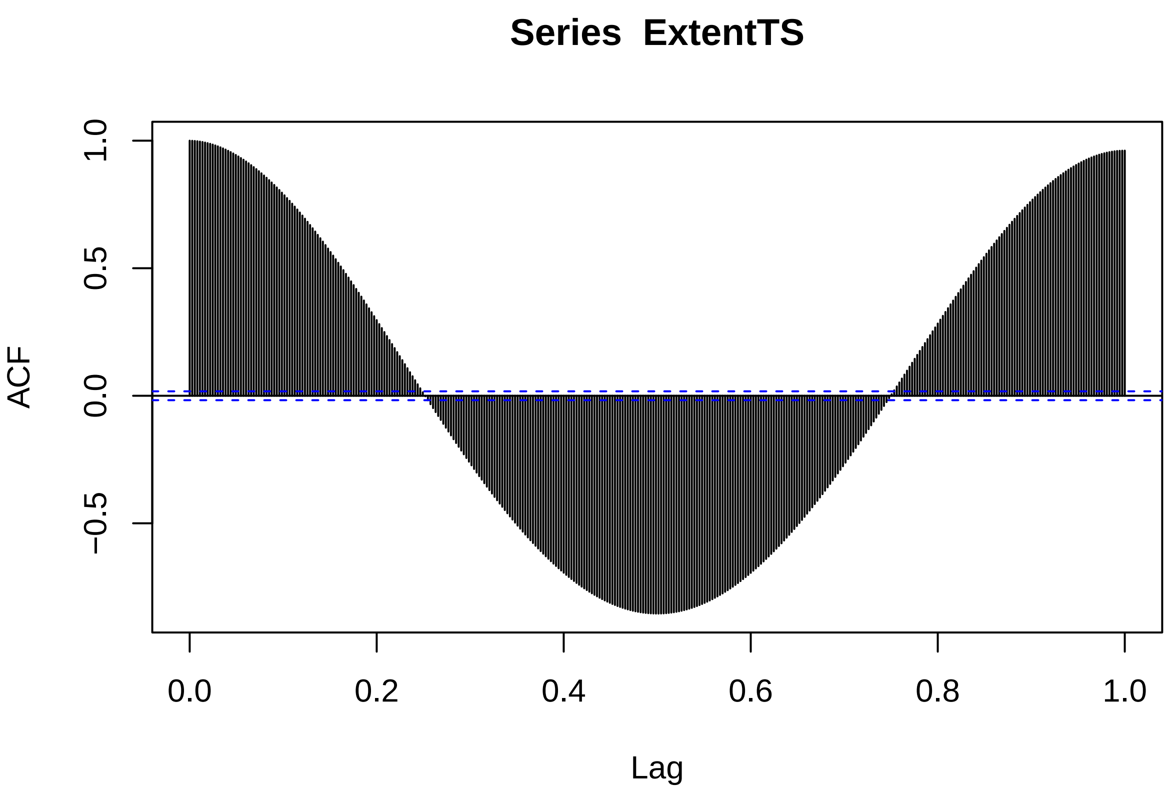
We first analyze the unaggregated data.

```
# make a TS object
ExtentTS <- ts(df$extent, frequency=365, start=year(df$YYMMDD[1]))
```

```
plot(ExtentTS)
```




```
acf(ExtentTS, lag.max=365)
```



Variance

From the plot, we see a clear seasonal pattern, and perhaps a decreasing linear trend.

It is unclear whether variance is constant. We test this using the Fligner-Keileen test.

```
# do Fligner test for constant variance.
segments = factor(c(rep(1:4, each=2542), rep(5, times=2543)))
fligner.test(ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  ExtentTS and segments
## Fligner-Killeen:med chi-squared = 170.73, df = 4, p-value < 2.2e-16
```

```
segments = factor(c(rep(1:9, each=1271), rep(10, times=1272)))
fligner.test(ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  ExtentTS and segments
## Fligner-Killeen:med chi-squared = 219.12, df = 9, p-value < 2.2e-16
```

```
segments = factor(c(rep(1:49, each=254), rep(50, times=265)))
fligner.test(ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  ExtentTS and segments
```

```
## Fligner-Killeen:med chi-squared = 1326.7, df = 49, p-value < 2.2e-16
```

```
segments = factor(c(rep(1:99, each=127), rep(100, times=138)))
fligner.test(ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  ExtentTS and segments
## Fligner-Killeen:med chi-squared = 3386.9, df = 99, p-value < 2.2e-16
```

```
segments = factor(c(rep(1:34, each=364), rep(35, times=335))) # corresponds more closely to each "wave"
fligner.test(ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  ExtentTS and segments
## Fligner-Killeen:med chi-squared = 368.79, df = 34, p-value < 2.2e-16
```

All give really low p-value so may conclude that variance is not constant. However, this could be due to the amount of data we have.

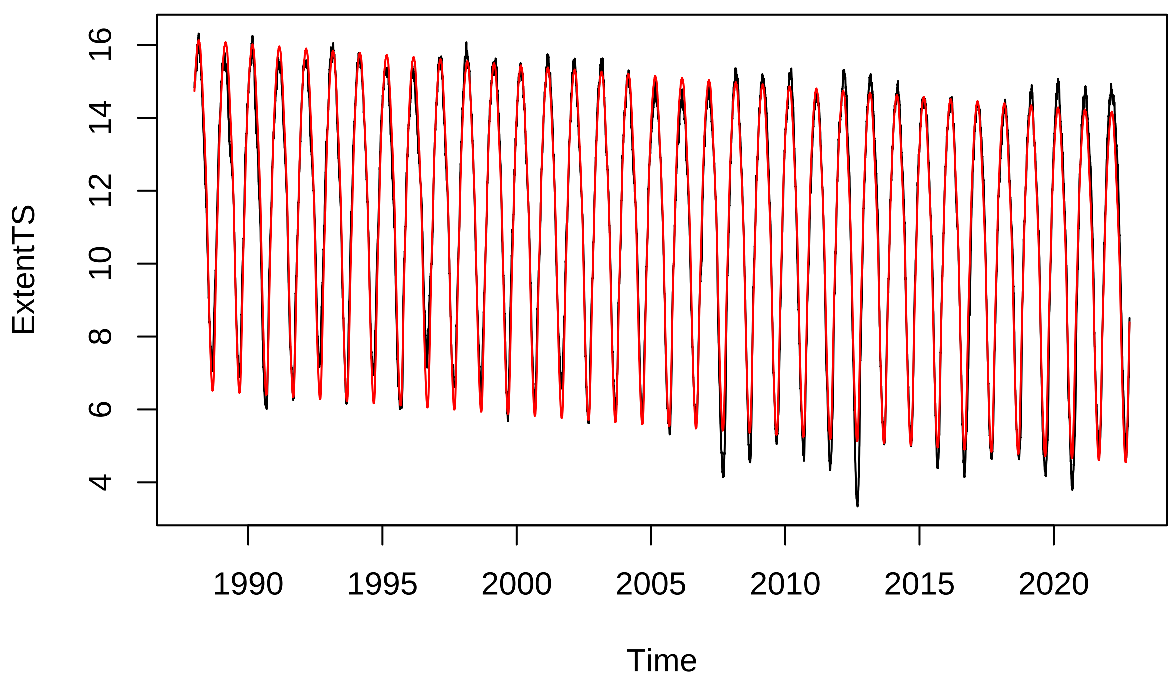
Regression

Try to remove non-stationarity using Regression (Multiple Linear, Ridge, Lasso, Elastic Net).

Multiple Linear Regression

```
mlr <- lm(ExtentTS~time(ExtentTS)+factor(cycle(ExtentTS)))
#summary(mlr) #verrrryyyyy long output and complicated model.
```

```
plot(ExtentTS)
points(time(ExtentTS),predict.lm(mlr),type='l',col='red')
```

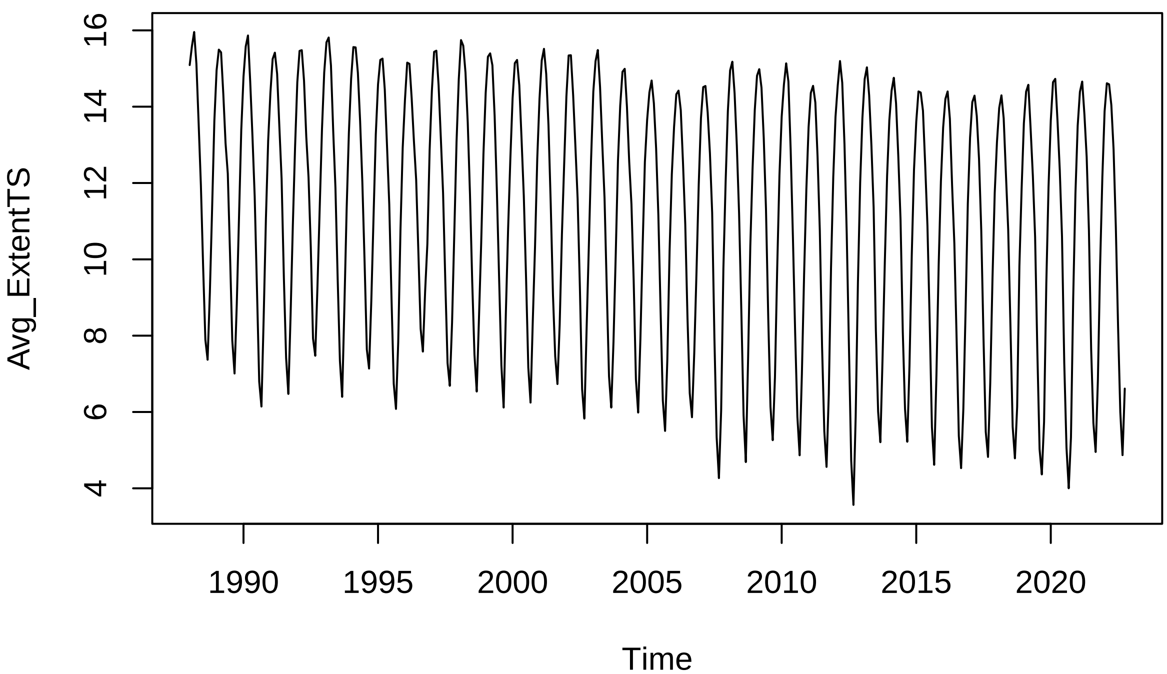


We see from the regression, that including daily data leads to a very complicated regression model, and acf plot which has to go way beyond recommended lag to observe an entire period. For this reason, and because we care mostly about overall trend and not daily fluctuation, we proceed with the aggregated data.

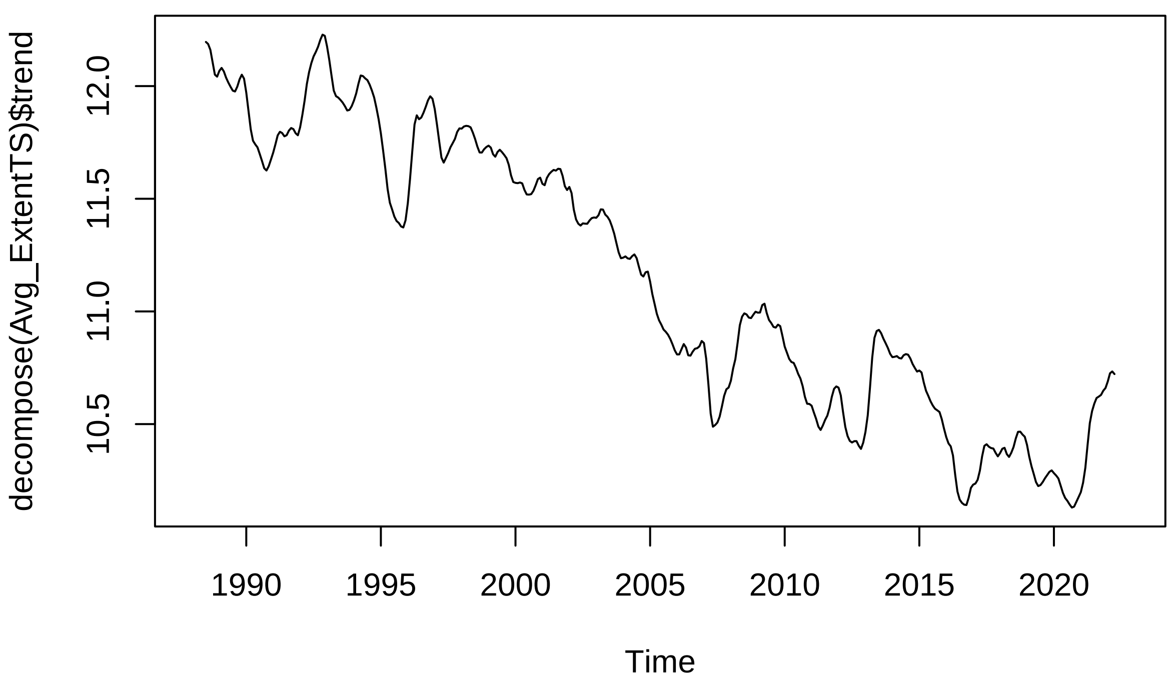
Aggregated Data

We proceed with analyzing the monthly aggregated data.

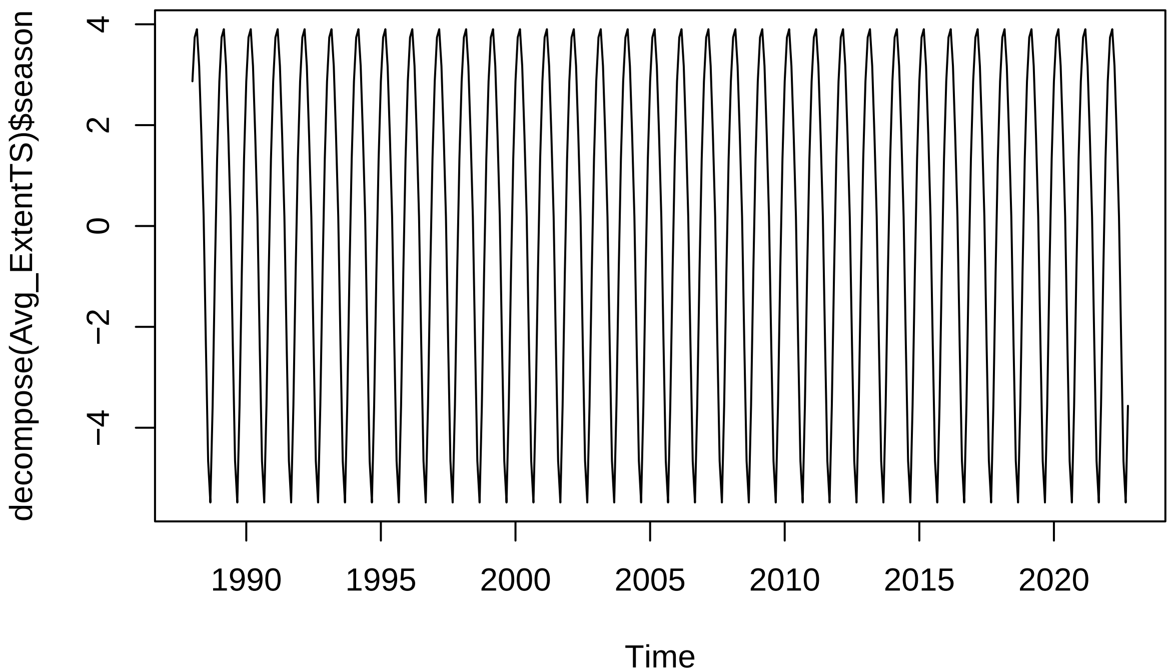
```r
# make a ts object
Avg_ExtentTS <- ts(df_aggregated$avg_extent, frequency=12, start=year(df$YYMMDD[1]))
plot(Avg_ExtentTS)
```



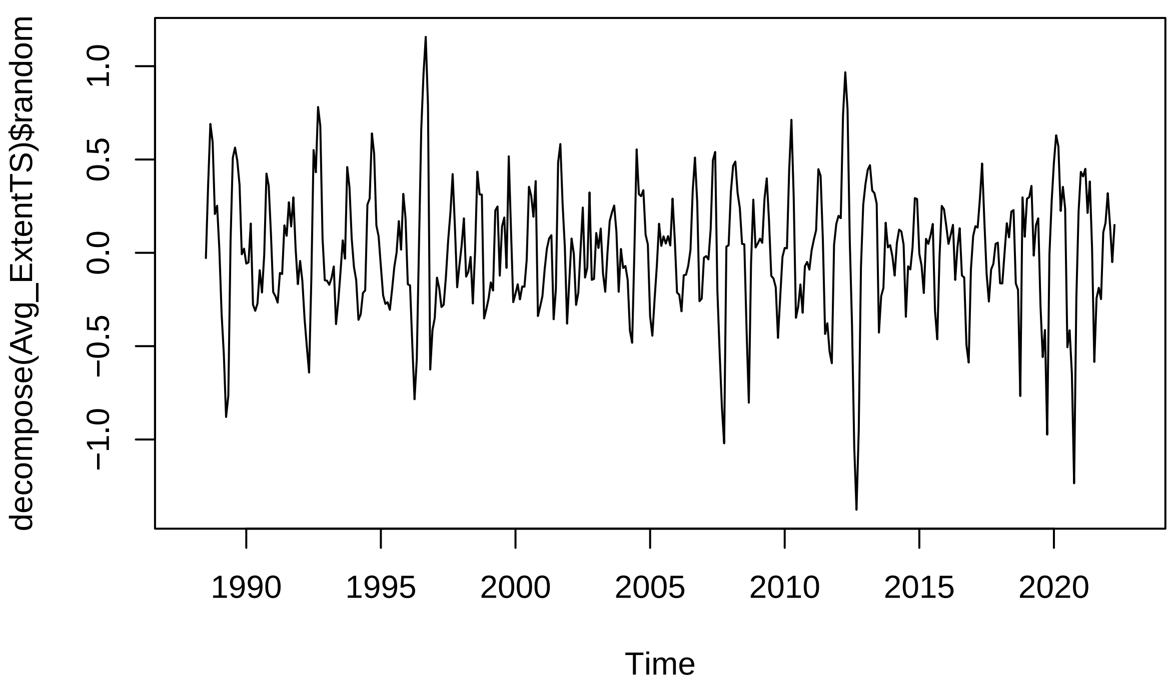

```r
plot(decompose(Avg_ExtentTS)$trend)
```



```
plot(decompose(Avg_ExtentTS)$season)
```




```
plot(decompose(Avg_ExtentTS)$random)
```



From the decomposition, we see there is a significant seasonal pattern, and likely significant trend.

Variance

From the plot, we see a clear seasonal pattern, and perhaps a decreasing linear trend.

It is unclear whether variance is constant. We test this using the Fligner-Killeen test.

```r
# do Fligner test for constant variance.
segments = factor(c(rep(1:4, each=84), rep(5, times=82)))
fligner.test(Avg_ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Avg_ExtentTS and segments
## Fligner-Killeen:med chi-squared = 5.8565, df = 4, p-value = 0.2101
```

```r
segments = factor(c(rep(1:9, each=42), rep(10, times=40)))
fligner.test(Avg_ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Avg_ExtentTS and segments
## Fligner-Killeen:med chi-squared = 8.2771, df = 9, p-value = 0.5065
```

```r
segments = factor(c(rep(1:19, each=21), rep(20, times=19)))
fligner.test(Avg_ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Avg_ExtentTS and segments
## Fligner-Killeen:med chi-squared = 13.213, df = 19, p-value = 0.8275
```

```r
segments = factor(c(rep(1:34, each=12), rep(35, times=10))) # corresponds to number of years of data
fligner.test(Avg_ExtentTS, segments)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Avg_ExtentTS and segments
## Fligner-Killeen:med chi-squared = 12.564, df = 34, p-value = 0.9997
```

All give high p-value so may conclude that variance is relatively constant. This is against expectation, but perhaps this is because the change in variance is not significant over such a small time frame.

```r
# define mse function for future use
mse <- function(y, yhat) {
  return(mean((as.vector(y)-as.vector(yhat))^2))
}
```

First, split the data, in to train and test set.

```r
Avg_ExtentTS_Train <- window(Avg_ExtentTS, 1988, 2020+11/12)
Avg_ExtentTS_Test <- window(Avg_ExtentTS, 2021, 2022+9/12)
```

Regression

Try to remove non-stationarity using Regression (Multiple Linear, Ridge, Lasso, Elastic Net).

Multiple Linear Regression

```r
tim <- as.vector(time(Avg_ExtentTS_Train))
season <- factor(cycle(Avg_ExtentTS_Train))
```

```r
# degree 1 polynomial of time
mlr_train <- lm(Avg_ExtentTS_Train~tim+season)

new <- data.frame(tim=as.vector(time(Avg_ExtentTS_Test)), season=factor(cycle(Avg_ExtentTS_Test)))
mse(Avg_ExtentTS_Test, predict.lm(mlr_train, new))
```

```
## [1] 0.3661747
```

```r
# degree 2 polynomial of time
mlr_train_2 <- lm(Avg_ExtentTS_Train~poly(tim,2)+season)

mse(Avg_ExtentTS_Test, predict.lm(mlr_train_2, new))
```

```
## [1] 0.4553482
```

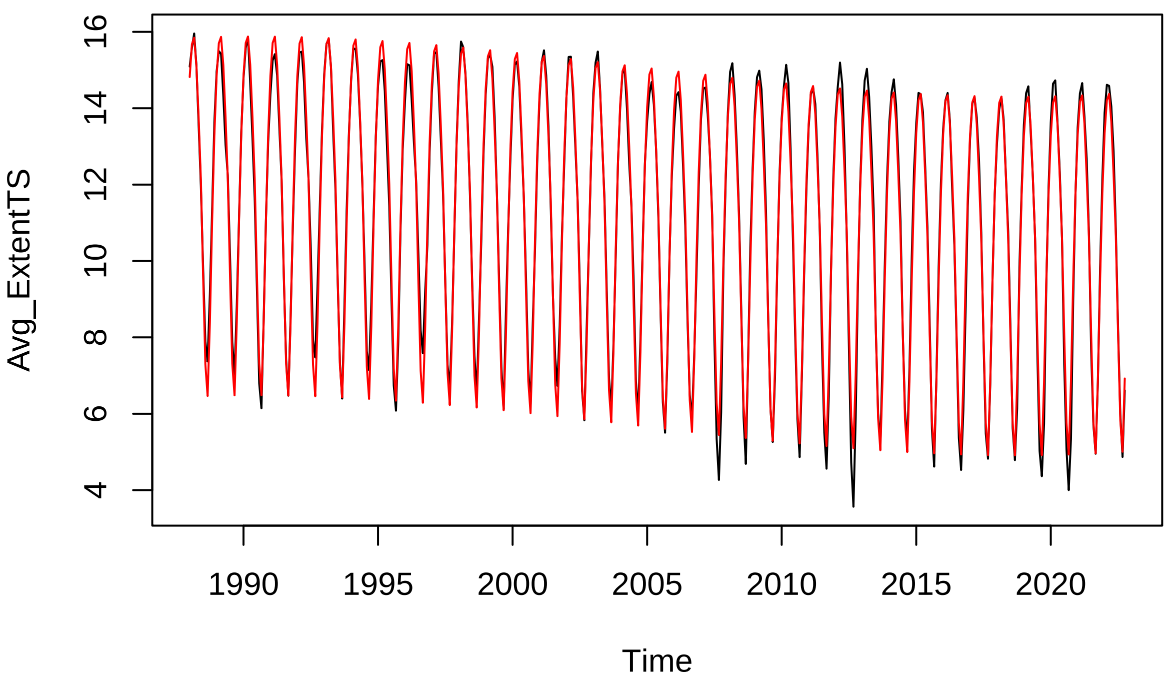
```r
# degree 3 polynomial of time
mlr_train_3 <- lm(Avg_ExtentTS_Train~poly(tim,3)+season)

mse(Avg_ExtentTS_Test, predict.lm(mlr_train_3, new))
```

```
## [1] 0.2334811
```

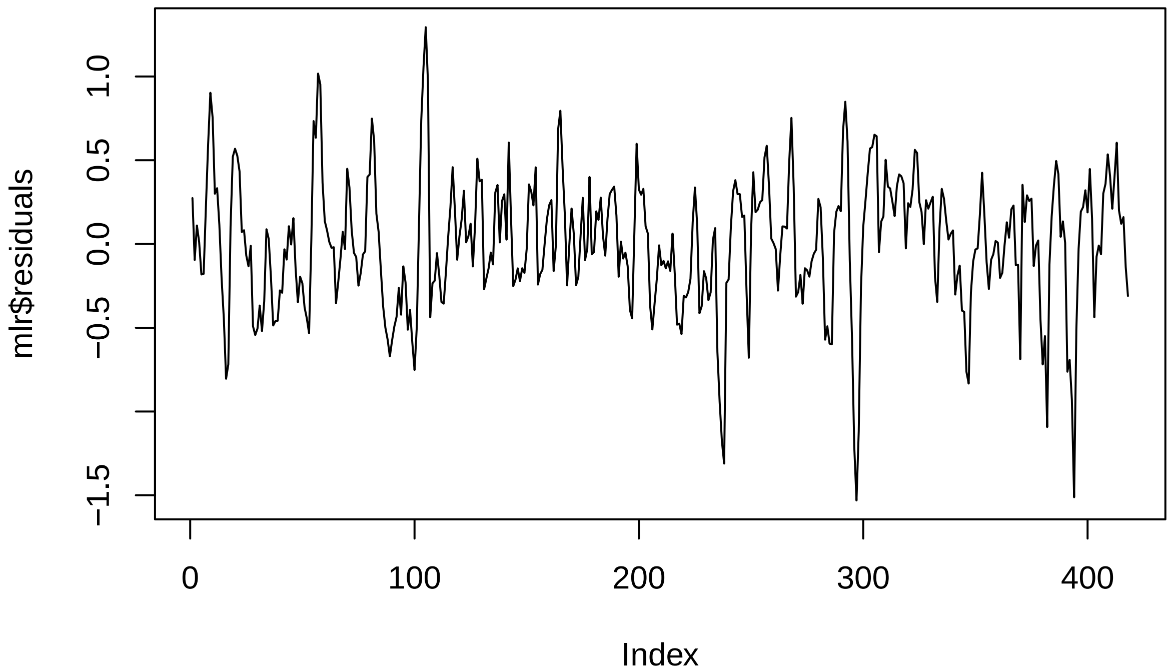
```r
#TODO residual diagnostics if proceed with this model
```

The cubic model performs best on the hold out set.

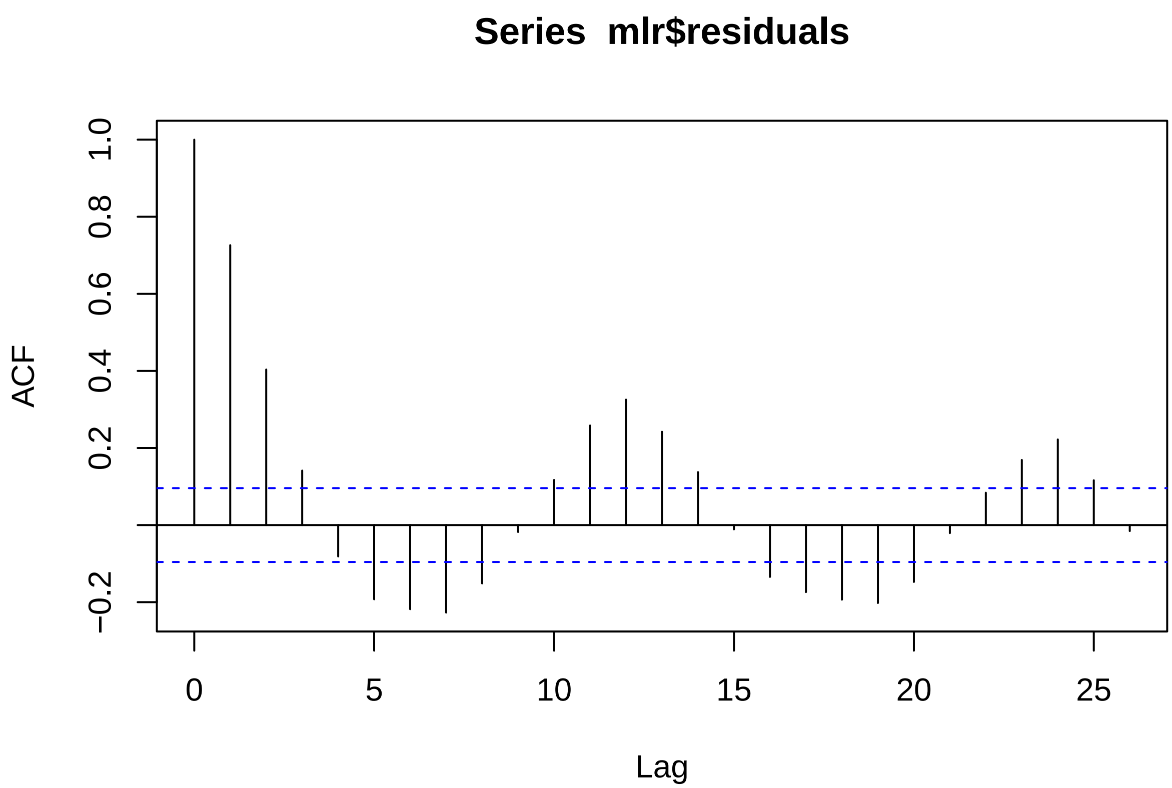
```r
tim <- as.vector(time(Avg_ExtentTS))
season <- factor(cycle(Avg_ExtentTS))
mlr <- lm(Avg_ExtentTS~poly(tim, 3)+season)

plot(Avg_ExtentTS)
points(time(Avg_ExtentTS), predict.lm(mlr), type='l', col='red')
```




```r
plot(mlr$residuals, type="l")
```




```
acf(mlr$residuals)
```



Ridge

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-4
```

```r
# fit to training data
tim <- as.vector(time(Avg_ExtentTS_Train))
season <- factor(cycle(Avg_ExtentTS_Train))
X <- model.matrix(as.vector(Avg_ExtentTS_Train)~tim+season)
X_2 <- model.matrix(as.vector(Avg_ExtentTS_Train)~poly(tim,2)+season)
X_3 <- model.matrix(as.vector(Avg_ExtentTS_Train)~poly(tim,3)+season)
ridge_train <- glmnet(X, as.vector(Avg_ExtentTS_Train), alpha=0)
ridge_train_2 <- glmnet(X_2, as.vector(Avg_ExtentTS_Train), alpha=0)
ridge_train_3 <- glmnet(X_3, as.vector(Avg_ExtentTS_Train), alpha=0)

# compute mse on training data for each value of lambda
ridge_train_fitted <- predict(ridge_train, X)
ridge_train_fitted_2 <- predict(ridge_train_2, X_2)
ridge_train_fitted_3 <- predict(ridge_train_3, X_3)

mses <- c()
mses_2 <- c()
mses_3 <- c()
for(i in 1:100) {
  mses <- c(mses, mse(Avg_ExtentTS_Train, ridge_train_fitted[,i]))
  mses_2 <- c(mses_2, mse(Avg_ExtentTS_Test, ridge_train_fitted_2[,i]))
  mses_3 <- c(mses_3, mse(Avg_ExtentTS_Test, ridge_train_fitted_3[,i]))
}

min10_mses <- head(sort(mses), 10)
min10_mses_2 <- head(sort(mses_2), 10)
min10_mses_3 <- head(sort(mses_3), 10)

ridge_train_lambdas <- c()
ridge_train_lambdas_2 <- c()
ridge_train_lambdas_3 <- c()

for(m in min10_mses) {
  ridge_train_lambdas <- c(ridge_train_lambdas, ridge_train$lambda[which(mses==m)])
}
for(m in min10_mses_2) {
  ridge_train_lambdas_2 <- c(ridge_train_lambdas_2, ridge_train_2$lambda[which(mses_2==m)])
}
for(m in min10_mses_3) {
  ridge_train_lambdas_3 <- c(ridge_train_lambdas_3, ridge_train_3$lambda[which(mses_3==m)])
}
```

```r
# retrain using lambdas that gave the 10 best fits
ridge_train <- glmnet(X, as.vector(Avg_ExtentTS_Train), alpha=0, lambda=ridge_train_lambdas)
ridge_train_2 <- glmnet(X_2, as.vector(Avg_ExtentTS_Train), alpha=0, lambda=ridge_train_lambdas_2)
ridge_train_3 <- glmnet(X_3, as.vector(Avg_ExtentTS_Train), alpha=0, lambda=ridge_train_lambdas_3)
```

```r
# predict the test set
tim <- as.vector(time(Avg_ExtentTS_Test))
season <- factor(cycle(Avg_ExtentTS_Test))
X <- model.matrix(as.vector(Avg_ExtentTS_Test)~tim+season)
X_2 <- model.matrix(as.vector(Avg_ExtentTS_Test)~poly(tim,2)+season)
X_3 <- model.matrix(as.vector(Avg_ExtentTS_Test)~poly(tim,3)+season)
ridge_predictions <- predict(ridge_train, X)
ridge_predictions_2 <- predict(ridge_train_2, X_2)
ridge_predictions_3 <- predict(ridge_train_3, X_3)

# compute pmse on test set
pmses <- c()
pmses_2 <- c()
pmses_3 <- c()
for(i in 1:10) {
  pmses <- c(pmses, mse(Avg_ExtentTS_Test, ridge_predictions[,i]))
  pmses_2 <- c(pmses_2, mse(Avg_ExtentTS_Test, ridge_predictions_2[,i]))
  pmses_3 <- c(pmses_3, mse(Avg_ExtentTS_Test, ridge_predictions_3[,i]))
}

ridge_train$lambda[which.min(pmses)]
```

```
## [1] 0.1653056
```

```r
pmses[which.min(pmses)]
```

```
## [1] 0.5345
```

```r
ridge_train_2$lambda[which.min(pmses_2)]
```

```
## [1] 715.5685
```

```r
pmses_2[which.min(pmses_2)]
```

```
## [1] 12.49462
```

```r
ridge_train_3$lambda[which.min(pmses_3)]
```

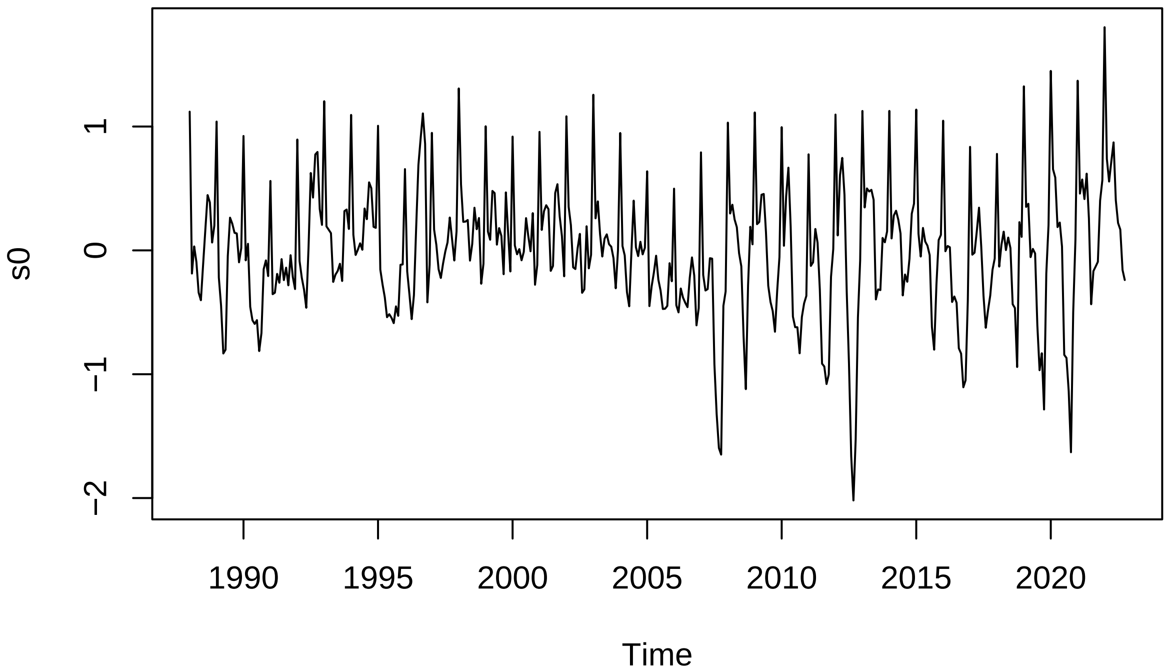
```
## [1] 715.5685
```

```r
pmses_3[which.min(pmses_3)]
```

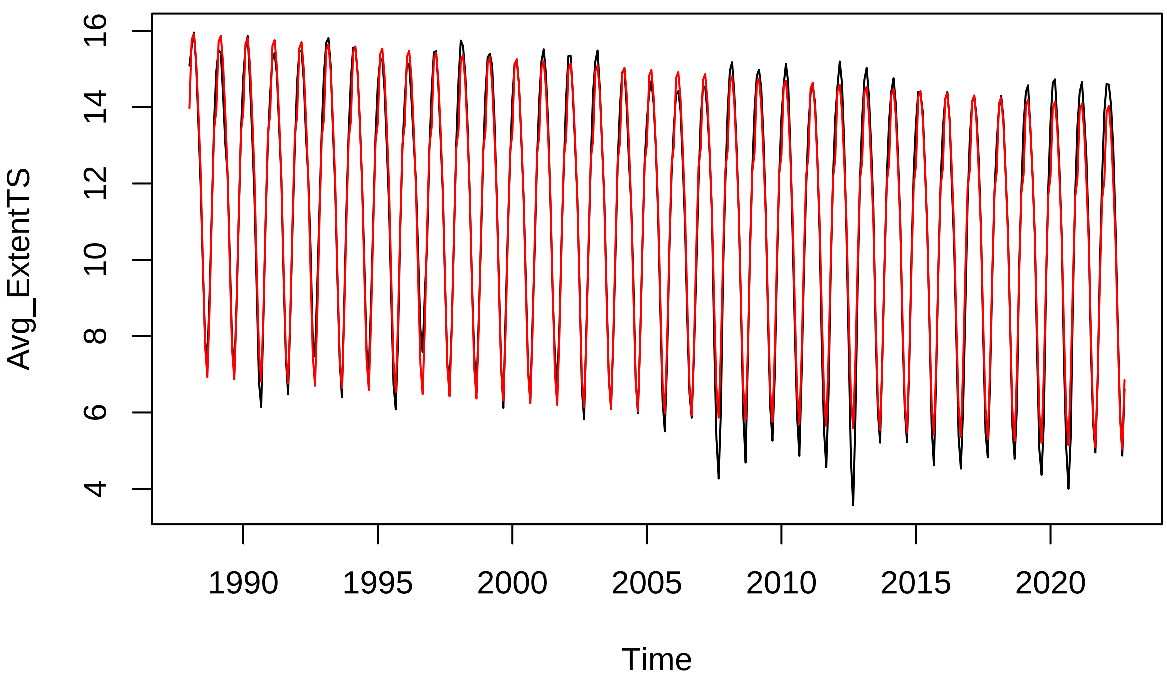
```
## [1] 12.4914
```

We see that the degree 1 polynomial gives the lowest MSE.

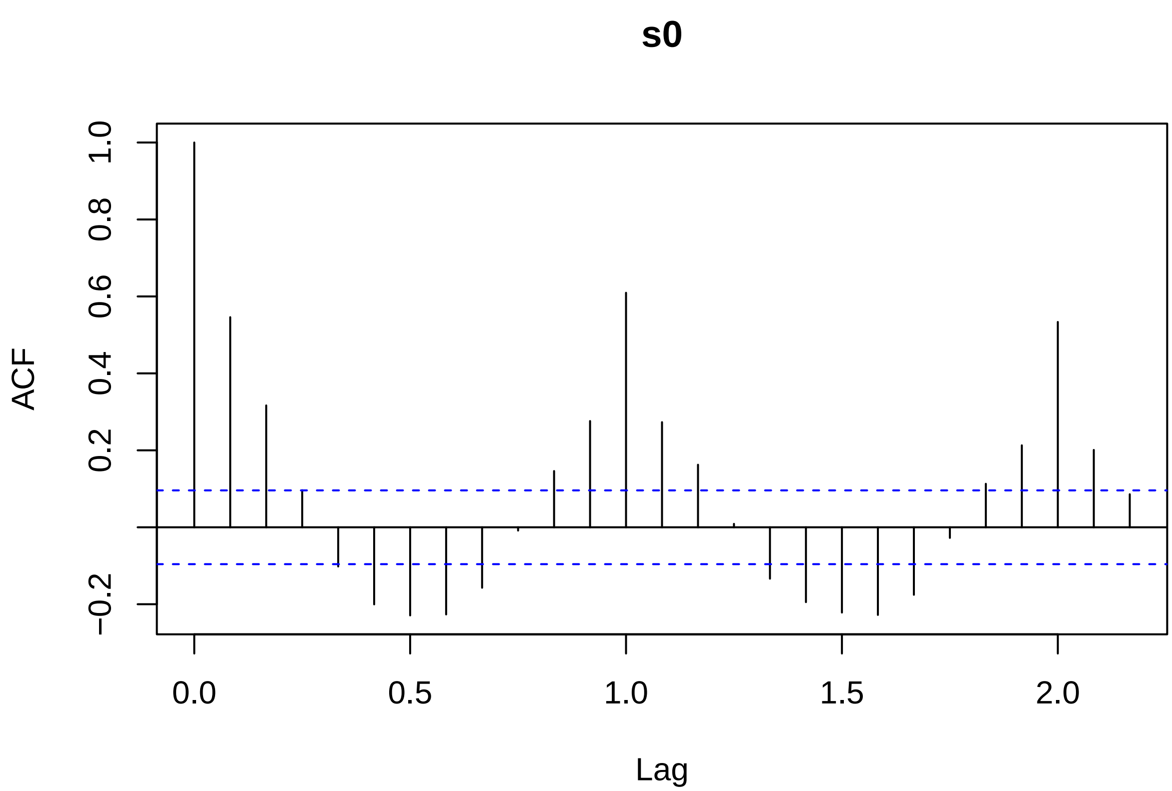
```r
#degree 1 polynomial
tim <- as.vector(time(Avg_ExtentTS))
season <- factor(cycle(Avg_ExtentTS))
X <- model.matrix(as.vector(Avg_ExtentTS)~tim+season)
ridge <- glmnet(X, as.vector(Avg_ExtentTS), alpha=0, lambda=0.1653056)
ridge_fitted <- predict(ridge, X, type="response")
ridge_residuals <- Avg_ExtentTS - ridge_fitted
plot(ridge_residuals, type="l")
```




```
plot(Avg_ExtentTS)
points(time(Avg_ExtentTS), ridge_fitted, type='l', col='red')
```




```
acf(ridge_residuals)
```




```
# TODO this plot but with train test split and acf of train
```

Lasso

```
library(glmnet)

# fit to training data
tim <- as.vector(time(Avg_ExtentTS_Train))
season <- factor(cycle(Avg_ExtentTS_Train))
X <- model.matrix(as.vector(Avg_ExtentTS_Train)~tim+season)
X_2 <- model.matrix(as.vector(Avg_ExtentTS_Train)~poly(tim,2)+season)
X_3 <- model.matrix(as.vector(Avg_ExtentTS_Train)~poly(tim,3)+season)
lasso_train <- glmnet(X, as.vector(Avg_ExtentTS_Train), alpha=1)
lasso_train_2 <- glmnet(X_2, as.vector(Avg_ExtentTS_Train), alpha=1)
lasso_train_3 <- glmnet(X_3, as.vector(Avg_ExtentTS_Train), alpha=1)

# compute mse on training data for each value of lambda
lasso_train_fitted <- predict(lasso_train, X)
lasso_train_fitted_2 <- predict(lasso_train_2, X_2)
lasso_train_fitted_3 <- predict(lasso_train_3, X_3)

mses <- c()
mses_2 <- c()
mses_3 <- c()
for(i in 1:67) {
  mses <- c(mses, mse(Avg_ExtentTS_Train, lasso_train_fitted[,i]))
}
for(i in 1:68) {
  mses_2 <- c(mses_2, mse(Avg_ExtentTS_Test, lasso_train_fitted_2[,i]))
```

```r
  mses_3 <- c(mses_3, mse(Avg_ExtentTS_Test, lasso_train_fitted_3[,i]))
}

min10_mses <- head(sort(mses), 10)
min10_mses_2 <- head(sort(mses_2), 10)
min10_mses_3 <- head(sort(mses_3), 10)

lasso_train_lambdas <- c()
lasso_train_lambdas_2 <- c()
lasso_train_lambdas_3 <- c()

for(m in min10_mses) {
  lasso_train_lambdas <- c(lasso_train_lambdas, lasso_train$lambda[which(mses==m)])
}
for(m in min10_mses_2) {
  lasso_train_lambdas_2 <- c(lasso_train_lambdas_2, lasso_train_2$lambda[which(mses_2==m)])
}
for(m in min10_mses_3) {
  lasso_train_lambdas_3 <- c(lasso_train_lambdas_3, lasso_train_3$lambda[which(mses_3==m)])
}
```

```r
# retrain using lambdas that gave the 10 best fits
lasso_train <- glmnet(X, as.vector(Avg_ExtentTS_Train), alpha=0, lambda=lasso_train_lambdas)
lasso_train_2 <- glmnet(X_2, as.vector(Avg_ExtentTS_Train), alpha=0, lambda=lasso_train_lambdas_2)
lasso_train_3 <- glmnet(X_3, as.vector(Avg_ExtentTS_Train), alpha=0, lambda=lasso_train_lambdas_3)

# predict the test set
tim <- as.vector(time(Avg_ExtentTS_Test))
season <- factor(cycle(Avg_ExtentTS_Test))
X <- model.matrix(as.vector(Avg_ExtentTS_Test)~tim+season)
X_2 <- model.matrix(as.vector(Avg_ExtentTS_Test)~poly(tim,2)+season)
X_3 <- model.matrix(as.vector(Avg_ExtentTS_Test)~poly(tim,3)+season)
lasso_predictions <- predict(lasso_train, X)
lasso_predictions_2 <- predict(lasso_train_2, X_2)
lasso_predictions_3 <- predict(lasso_train_3, X_3)

# compute pmse on test set
pmses <- c()
pmses_2 <- c()
pmses_3 <- c()
for(i in 1:10) {
  pmses <- c(pmses, mse(Avg_ExtentTS_Test, lasso_predictions[,i]))
  pmses_2 <- c(pmses_2, mse(Avg_ExtentTS_Test, lasso_predictions_2[,i]))
  pmses_3 <- c(pmses_3, mse(Avg_ExtentTS_Test, lasso_predictions_3[,i]))
}

lasso_train$lambda[which.min(pmses)]
```

```
## [1] 0.003561401
```

```r
pmses[which.min(pmses)]
```

```
## [1] 0.368189
```

```r
lasso_train_2$lambda[which.min(pmses_2)]
```

```
## [1] 1.653056
```

```r
pmses_2[which.min(pmses_2)]
```

```
## [1] 3.662887
```

```r
lasso_train_3$lambda[which.min(pmses_3)]
```

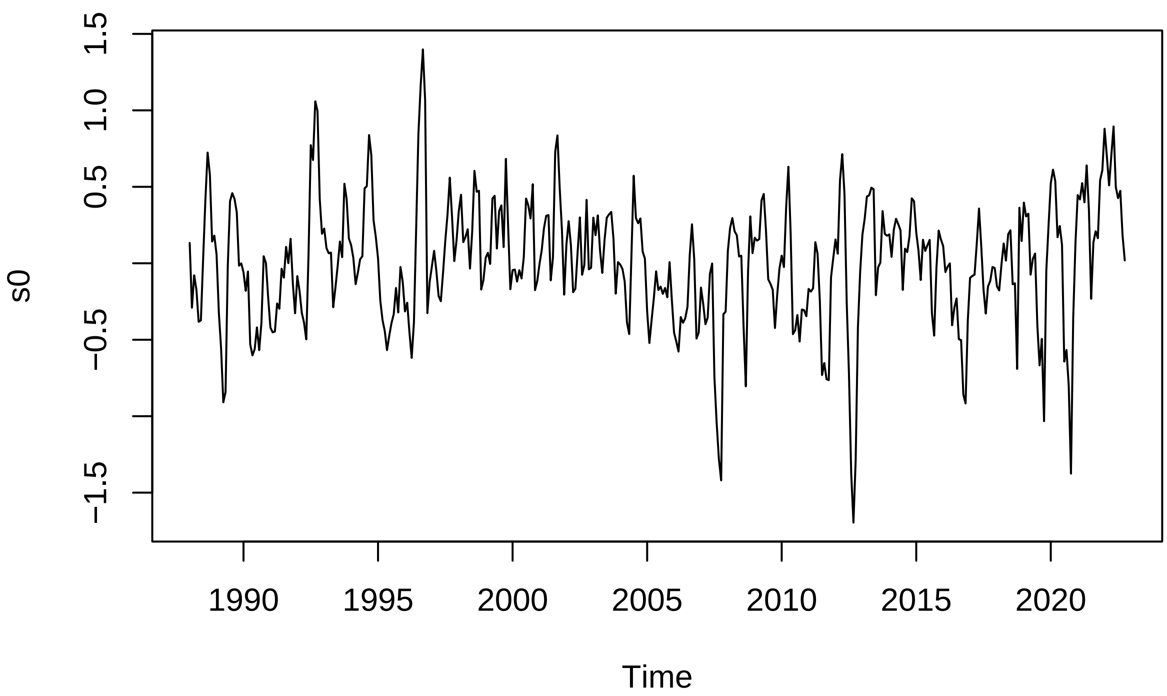
```
## [1] 1.653056
```

```r
pmses_3[which.min(pmses_3)]
```

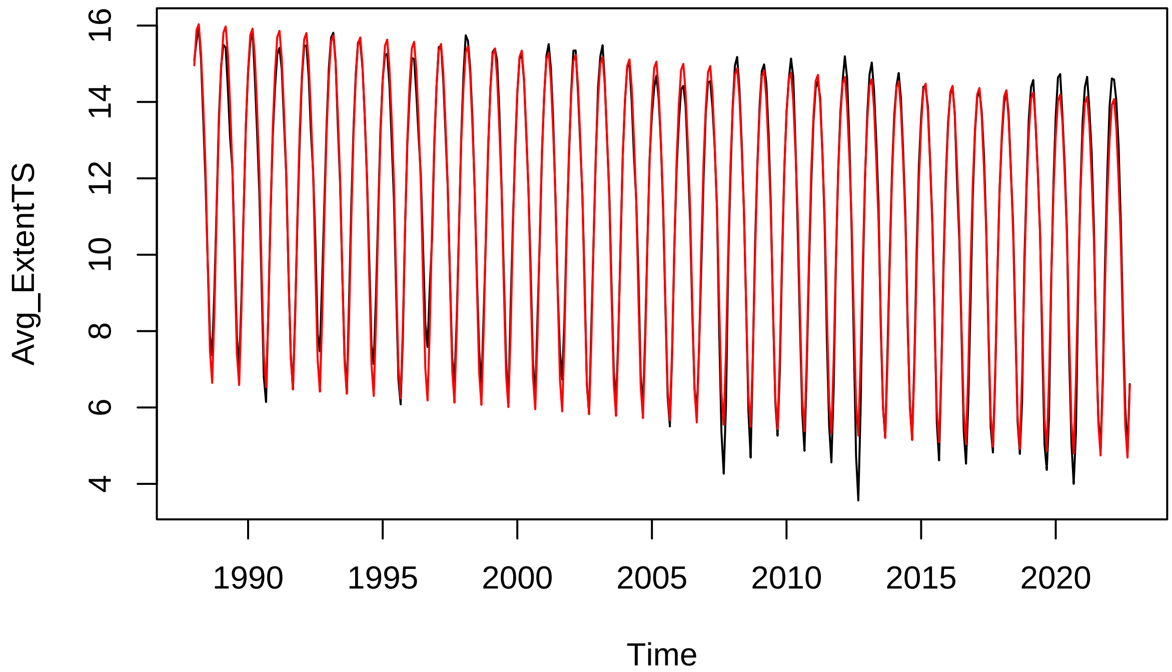
```
## [1] 3.770674
```

We see that the degree 1 polynomial gives the lowest MSE.

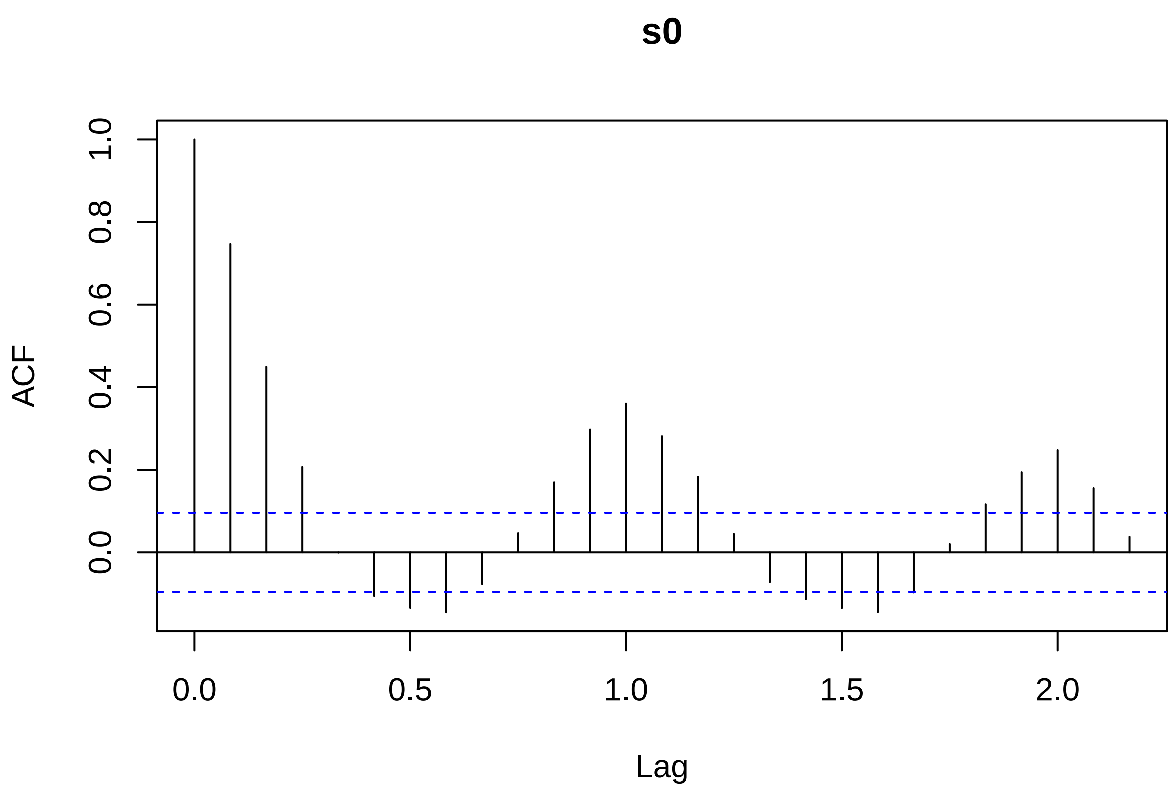
```r
# degree 1 polynomial of time
tim <- as.vector(time(Avg_ExtentTS))
season <- factor(cycle(Avg_ExtentTS))
X <- model.matrix(as.vector(Avg_ExtentTS)~tim+season)
lasso <- glmnet(X, as.vector(Avg_ExtentTS), alpha=1, lambda=0.003561401)
lasso_fitted <- predict(lasso, X, type="response")
lasso_residuals <- Avg_ExtentTS - lasso_fitted
plot(lasso_residuals, type="l")
```




```r
plot(Avg_ExtentTS)
points(time(Avg_ExtentTS), lasso_fitted, type='l', col='red')
```




```
acf(lasso_residuals)
```




```
# TODO this plot but with train test split and acf of train
```

Elastic Net

```
library(glmnet)
alpha_seq <- seq(0.1, 0.9, by=0.1)
```

```r
en_train_min_lamdas <- c()
en_train_min_lamdas_2 <- c()
en_train_min_lamdas_3 <- c()
min_pmses <- c()
min_pmses_2 <- c()
min_pmses_3 <- c()

for(a in alpha_seq){
  # fit to training data
  tim <- as.vector(time(Avg_ExtentTS_Train))
  season <- factor(cycle(Avg_ExtentTS_Train))
  X <- model.matrix(as.vector(Avg_ExtentTS_Train)~tim+season)
  X_2 <- model.matrix(as.vector(Avg_ExtentTS_Train)~poly(tim,2)+season)
  X_3 <- model.matrix(as.vector(Avg_ExtentTS_Train)~poly(tim,3)+season)
  en_train <- glmnet(X, as.vector(Avg_ExtentTS_Train), alpha=a)
  en_train_2 <- glmnet(X_2, as.vector(Avg_ExtentTS_Train), alpha=a)
  en_train_3 <- glmnet(X_3, as.vector(Avg_ExtentTS_Train), alpha=a)

  # compute mse on training data for each value of lambda
  en_train_fitted <- predict(en_train, X)
  en_train_fitted_2 <- predict(en_train_2, X_2)
  en_train_fitted_3 <- predict(en_train_3, X_3)

  mses <- c()
  mses_2 <- c()
  mses_3 <- c()
  for(i in 1:length(en_train$lambda)) {
    mses <- c(mses, mse(Avg_ExtentTS_Train, en_train_fitted[,i]))
  }
  for(i in 1:length(en_train_2$lambda)) {
    mses_2 <- c(mses_2, mse(Avg_ExtentTS_Test, en_train_fitted_2[,i]))
  }
  for(i in 1:length(en_train_3$lambda)) {
    mses_3 <- c(mses_3, mse(Avg_ExtentTS_Test, en_train_fitted_3[,i]))
  }

  min10_mses <- head(sort(mses), 10)
  min10_mses_2 <- head(sort(mses_2), 10)
  min10_mses_3 <- head(sort(mses_3), 10)

  en_train_lambdas <- c()
  en_train_lambdas_2 <- c()
  en_train_lambdas_3 <- c()

  for(m in min10_mses) {
    en_train_lambdas <- c(en_train_lambdas, en_train$lambda[which(mses==m)])
  }
  for(m in min10_mses_2) {
    en_train_lambdas_2 <- c(en_train_lambdas_2, en_train_2$lambda[which(mses_2==m)])
  }
  for(m in min10_mses_3) {
    en_train_lambdas_3 <- c(en_train_lambdas_3, en_train_3$lambda[which(mses_3==m)])
  }
```

```r
  # retrain using lambdas that gave the 10 best fits
  en_train <- glmnet(X, as.vector(Avg_ExtentTS_Train), alpha=a, lambda=en_train_lambdas)
  en_train_2 <- glmnet(X_2, as.vector(Avg_ExtentTS_Train), alpha=a, lambda=en_train_lambdas_2)
  en_train_3 <- glmnet(X_3, as.vector(Avg_ExtentTS_Train), alpha=a, lambda=en_train_lambdas_3)

  # predict the test set
  tim <- as.vector(time(Avg_ExtentTS_Test))
  season <- factor(cycle(Avg_ExtentTS_Test))
  X <- model.matrix(as.vector(Avg_ExtentTS_Test)~tim+season)
  X_2 <- model.matrix(as.vector(Avg_ExtentTS_Test)~poly(tim,2)+season)
  X_3 <- model.matrix(as.vector(Avg_ExtentTS_Test)~poly(tim,3)+season)
  en_predictions <- predict(en_train, X)
  en_predictions_2 <- predict(en_train_2, X_2)
  en_predictions_3 <- predict(en_train_3, X_3)

  # compute pmse on test set
  pmses <- c()
  pmses_2 <- c()
  pmses_3 <- c()
  for(i in 1:10) {
    pmses <- c(pmses, mse(Avg_ExtentTS_Test, en_predictions[,i]))
    pmses_2 <- c(pmses_2, mse(Avg_ExtentTS_Test, en_predictions_2[,i]))
    pmses_3 <- c(pmses_3, mse(Avg_ExtentTS_Test, en_predictions_3[,i]))
  }

  min_pmses <- c(min_pmses, pmses[which.min(pmses)])
  min_pmses_2 <- c(min_pmses_2, pmses_2[which.min(pmses_2)])
  min_pmses_3 <- c(min_pmses_3, pmses_3[which.min(pmses_3)])

  en_train_min_lamdas <- c(en_train_min_lamdas, en_train$lambda[which.min(pmses)])
  en_train_min_lamdas_2 <- c(en_train_min_lamdas_2, en_train_2$lambda[which.min(pmses_2)])
  en_train_min_lamdas_3 <- c(en_train_min_lamdas_3, en_train_3$lambda[which.min(pmses_3)])
}

min_pmses[which.min(min_pmses)]
```

```
## [1] 0.3659694
```

```r
alpha_seq[which.min(min_pmses)]
```

```
## [1] 0.9
```

```r
en_train_min_lamdas[which.min(min_pmses)]
```

```
## [1] 0.004342926
```

```r
min_pmses_2[which.min(min_pmses_2)]
```

```
## [1] 5.553512
```

```r
alpha_seq[which.min(min_pmses_2)]
```

```
## [1] 0.9
```

```r
en_train_min_lamdas_2[which.min(min_pmses_2)]
```

```
## [1] 0.7950761
```

```
min_pmses_3[which.min(min_pmses_3)]
```

```
## [1] 5.553512
```

```
alpha_seq[which.min(min_pmses_3)]
```

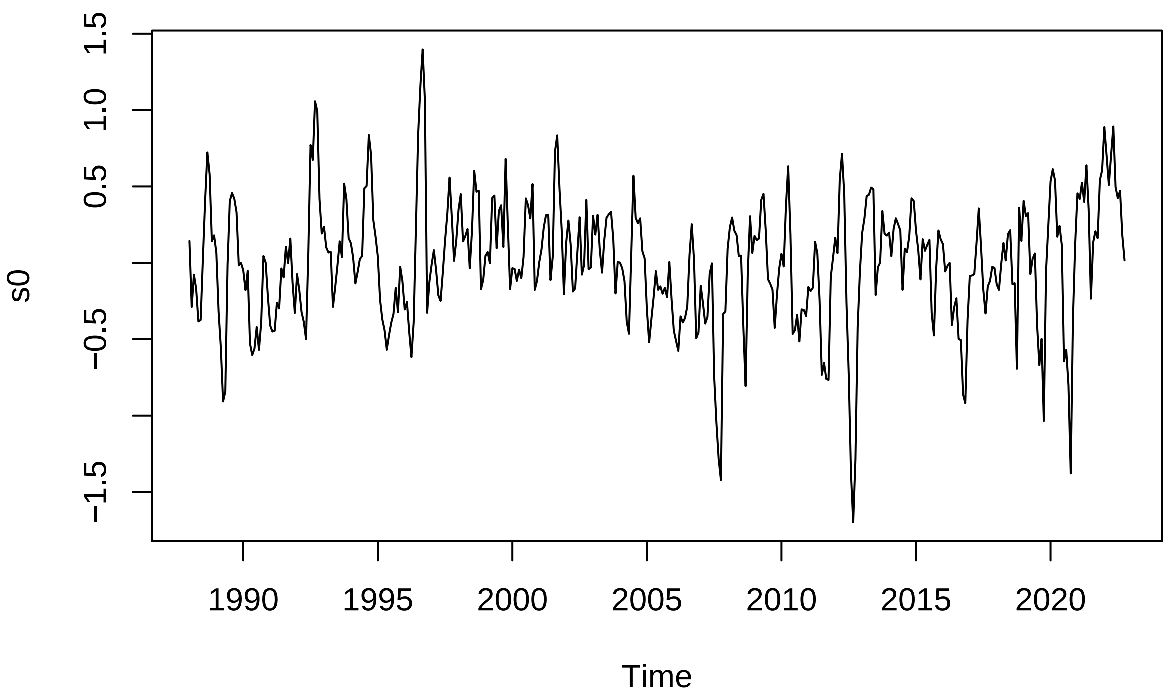
```
## [1] 0.9
```

```
en_train_min_lamdas_3[which.min(min_pmses_3)]
```

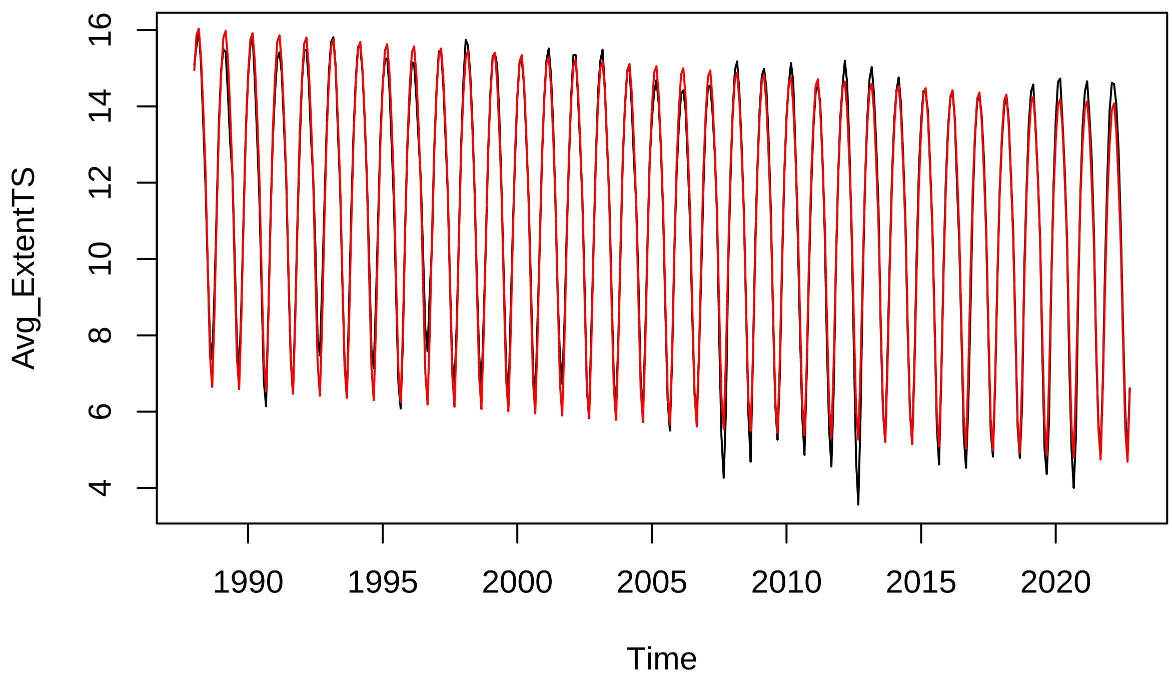
```
## [1] 0.7950761
```

We see that the degree 1 polynomial gives the lowest MSE.

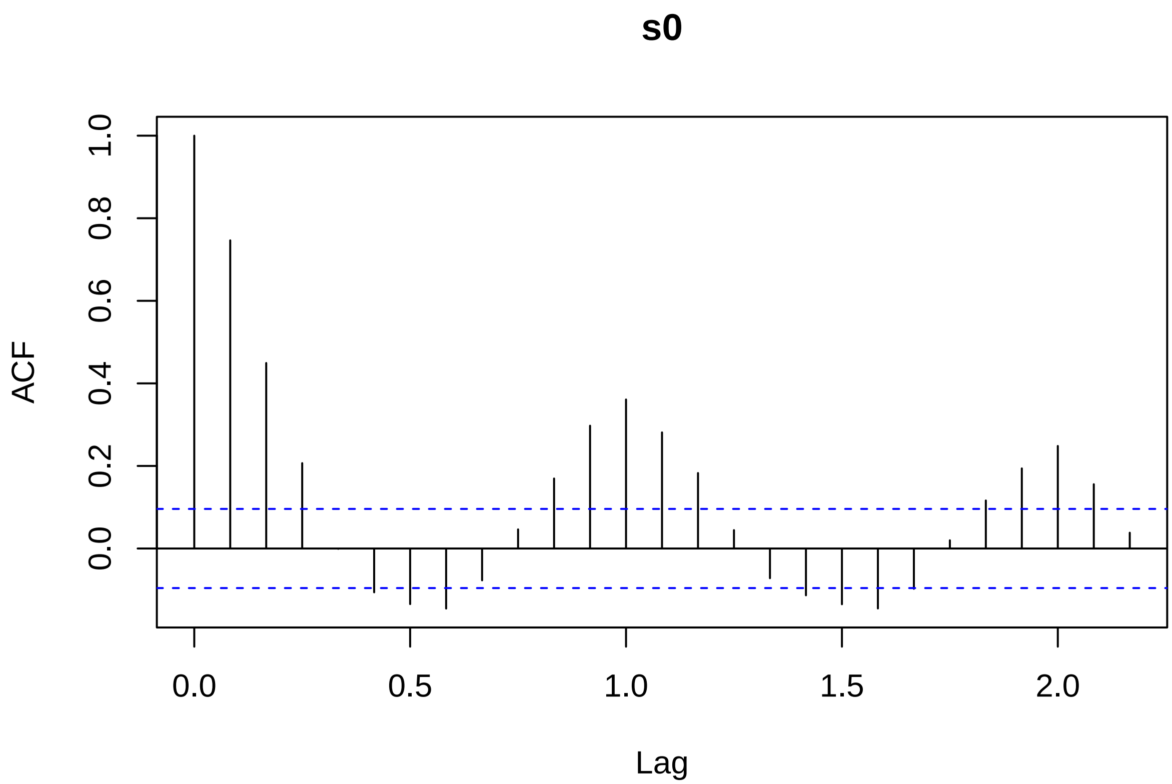
```
#degree 1 polynomial
tim <- as.vector(time(Avg_ExtentTS))
season <- factor(cycle(Avg_ExtentTS))
X <- model.matrix(as.vector(Avg_ExtentTS)~tim+season)
ridge <- glmnet(X, as.vector(Avg_ExtentTS), alpha=0.9, lambda=0.004342926)
ridge_fitted <- predict(ridge, X, type="response")
ridge_residuals <- Avg_ExtentTS - ridge_fitted
plot(ridge_residuals, type="l")
```




```
plot(Avg_ExtentTS)
points(time(Avg_ExtentTS), ridge_fitted, type='l', col='red')
```




```
acf(ridge_residuals)
```




```
# TODO this plot but with train test split and acf of train
```

Holt-Winters

Try to remove non-stationarity using exponential smoothing, double exponential smoothing, additive HW, and multiplicative HW.

Exponential Smoothing

```
es <- HoltWinters(Avg_ExtentTS_Train, gamma = FALSE , beta = FALSE)
predict_es = predict(es, n.ahead=22)
mse(Avg_ExtentTS_Test, predict_es)
```

```
## [1] 13.56088
```

Double Exponential Smoothing

```
des <- HoltWinters(Avg_ExtentTS_Train, gamma = FALSE)
predict_des = predict(des, n.ahead=22)
mse(Avg_ExtentTS_Test, predict_des)
```

```
## [1] 14.46845
```

No Trend

```
no_trend <- HoltWinters(Avg_ExtentTS_Train, beta = FALSE)
predict_no_trend = predict(no_trend, n.ahead=22)
mse(Avg_ExtentTS_Test, predict_no_trend)
```

```
## [1] 0.4592097
```

Additive Holt-Winters

```
additive <- HoltWinters(Avg_ExtentTS_Train, seasonal = "additive")
predict_additive = predict(additive, n.ahead=22)
mse(Avg_ExtentTS_Test, predict_additive)
```

```
## [1] 0.7777211
```

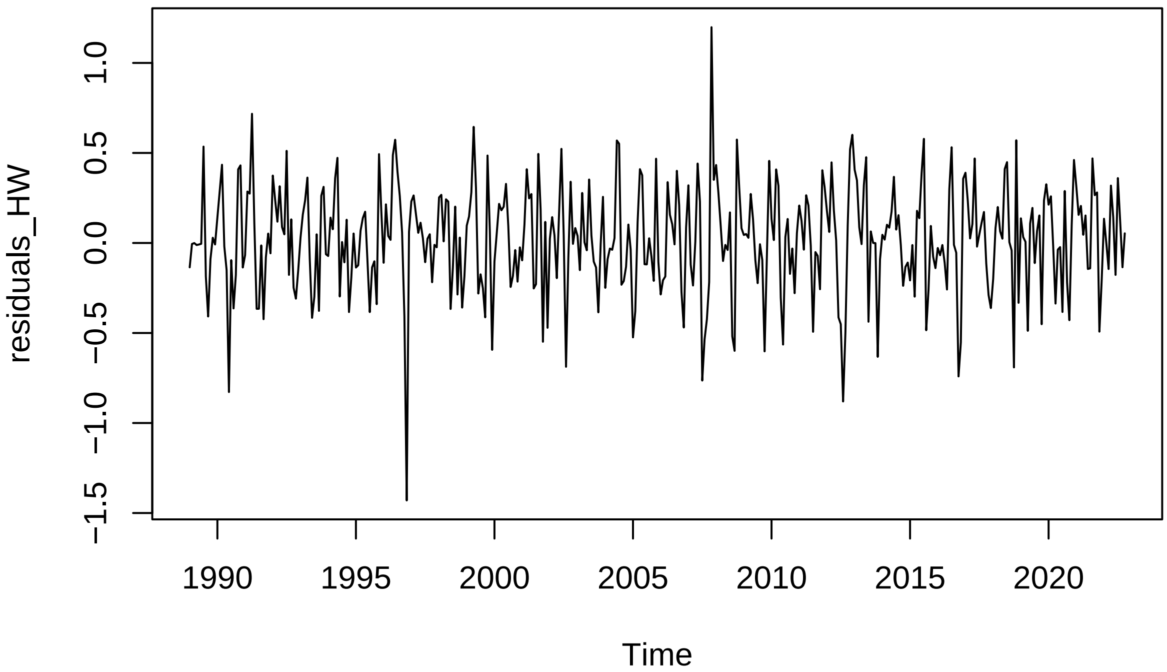
Multiplicative Holt-Winters

```
multiplicative <- HoltWinters(Avg_ExtentTS_Train, seasonal = "multiplicative")
predict_multiplicative = predict(multiplicative, n.ahead=22)
mse(Avg_ExtentTS_Test, predict_additive)
```

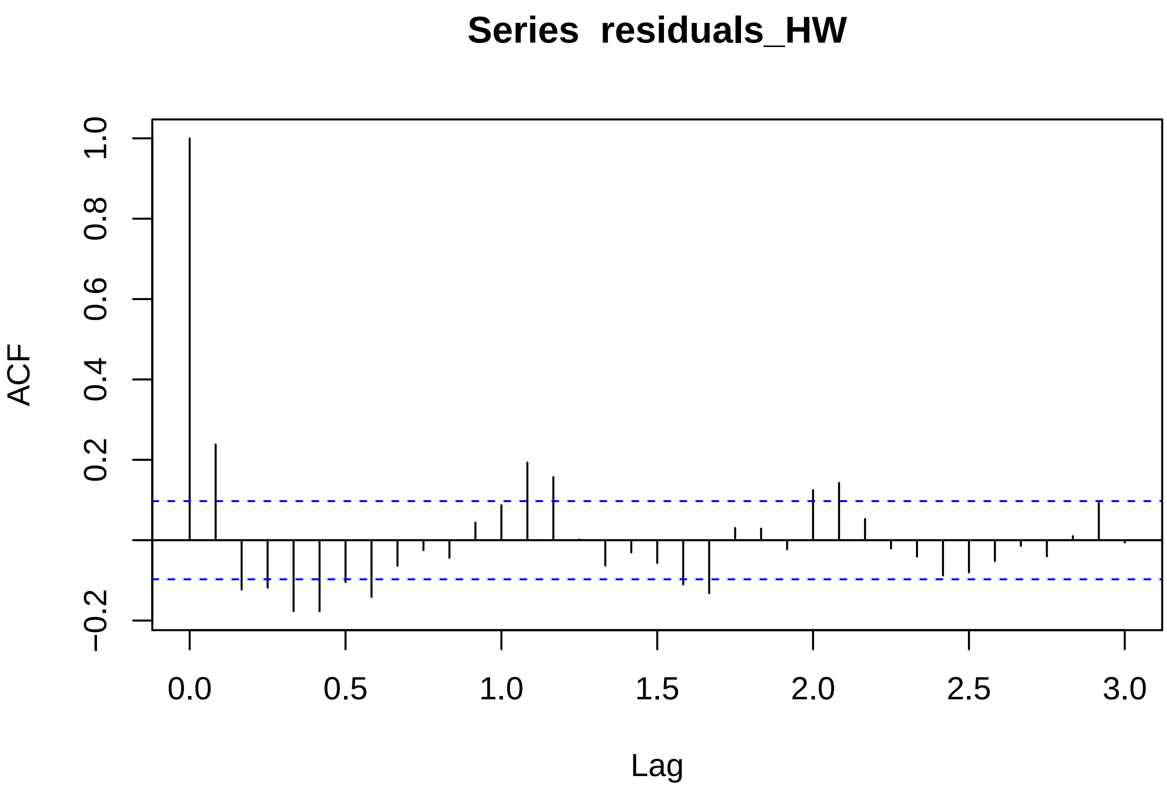
```
## [1] 0.7777211
```

Best of HW models seems to be additive/multiplicatice model. We fit this model to the entire data.

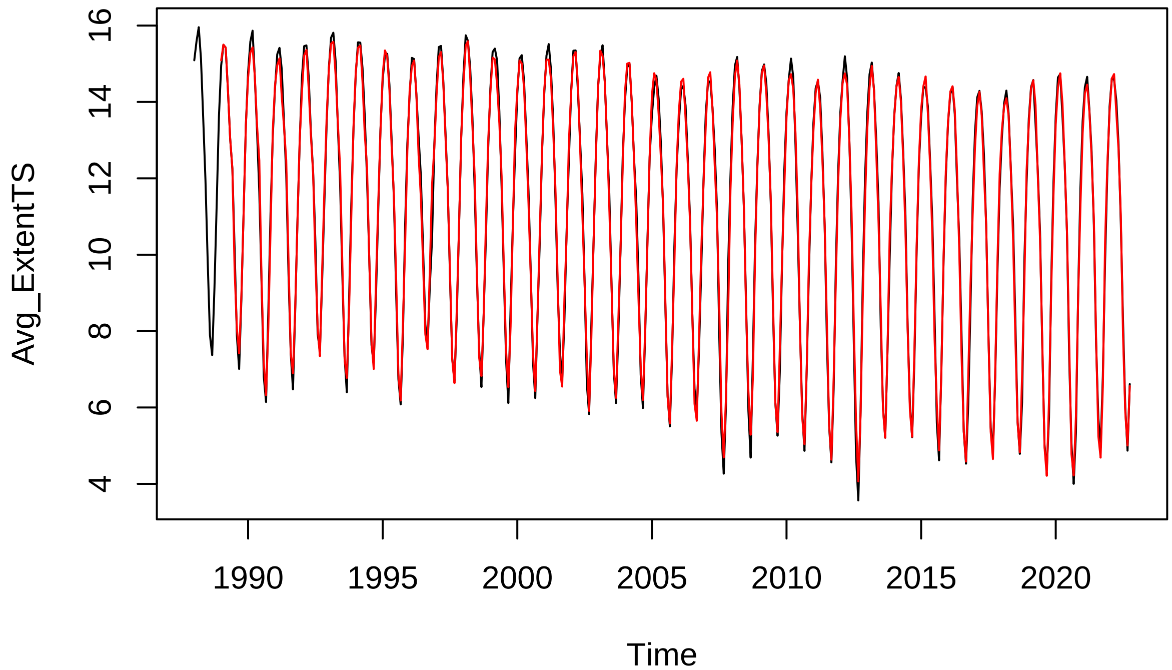
```
hw_additive <- HoltWinters(Avg_ExtentTS, seasonal = "additive")
residuals_HW <- as.vector(Avg_ExtentTS[which(time(Avg_ExtentTS)>=1989)]) - hw_additive$fitted[,1]
plot(residuals_HW, type="l")
```




```
acf(residuals_HW, lag.max=36)
```

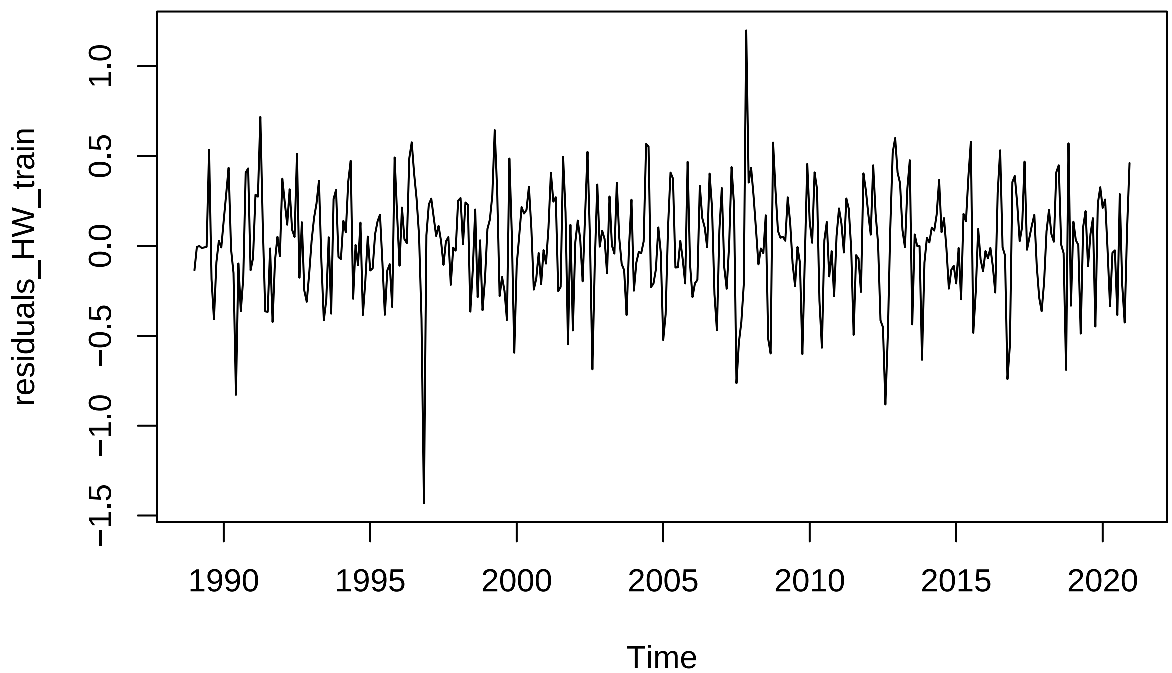



```
plot(Avg_ExtentTS)
points(time(Avg_ExtentTS)[which(time(Avg_ExtentTS)>=1989)], hw_additive$fitted[,1], type='l', col='red')
```

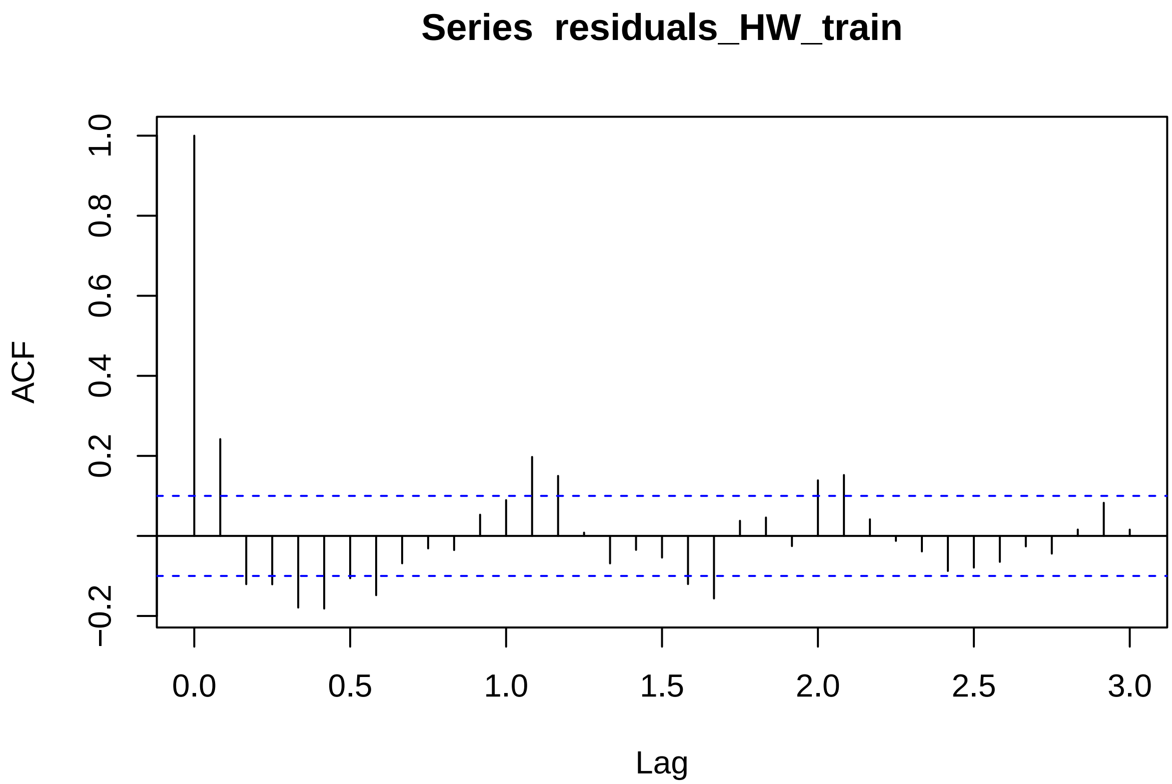



```
#plot(hw_additive)
```

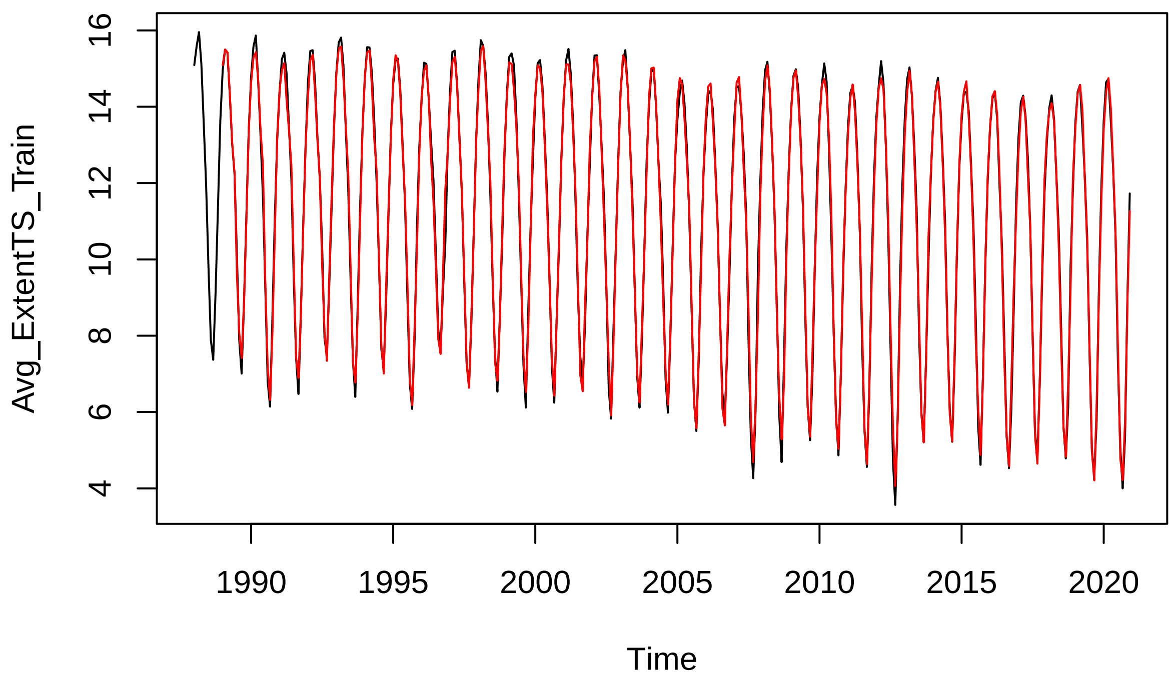
```
# Same as above but on training data
hw_additive_train <- HoltWinters(Avg_ExtentTS_Train, seasonal = "additive")
residuals_HW_train <- as.vector(Avg_ExtentTS_Train[which(time(Avg_ExtentTS_Train)>=1989)]) - hw_additive
plot(residuals_HW_train, type="l")
```




```
acf(residuals_HW_train, lag.max=36)
```

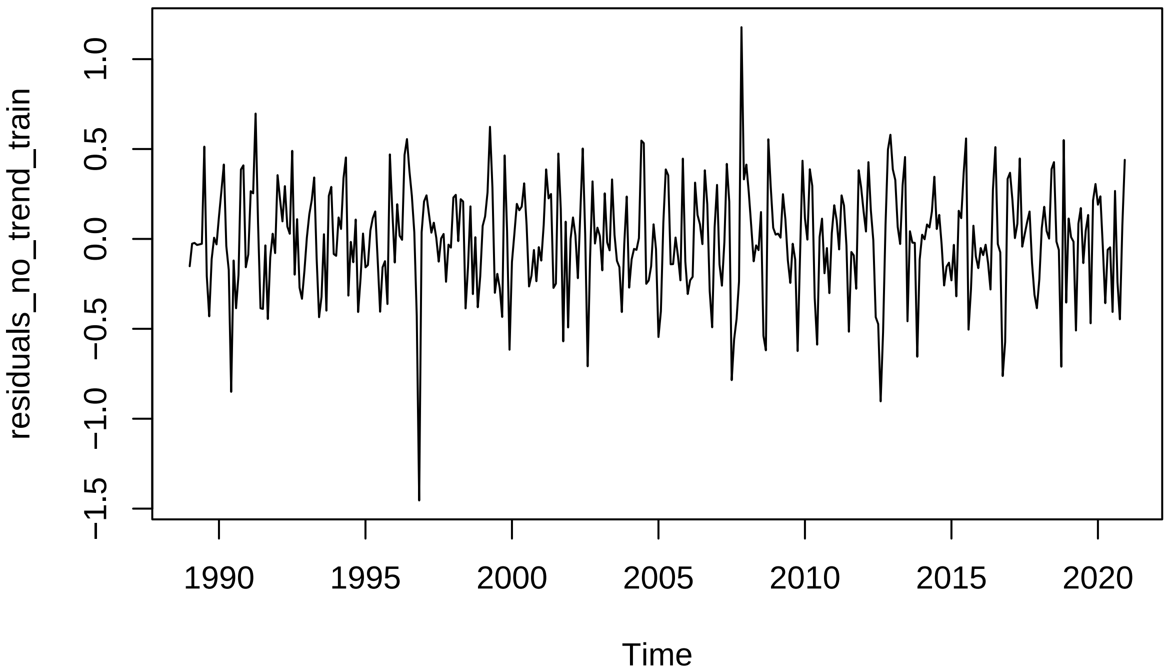



```
plot(Avg_ExtentTS_Train)
points(time(Avg_ExtentTS_Train)[which(time(Avg_ExtentTS_Train)>=1989)], hw_additive_train$fitted[,1], ty
```

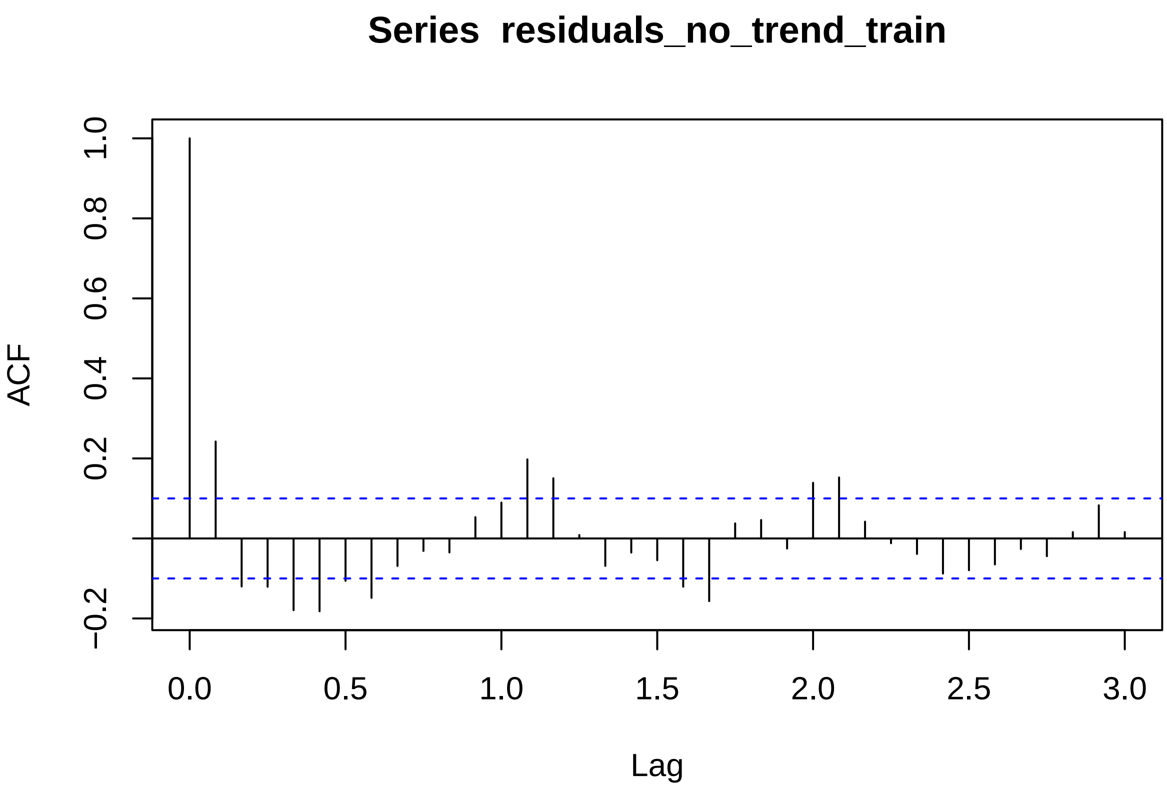



```
#plot(hw_additive)
```

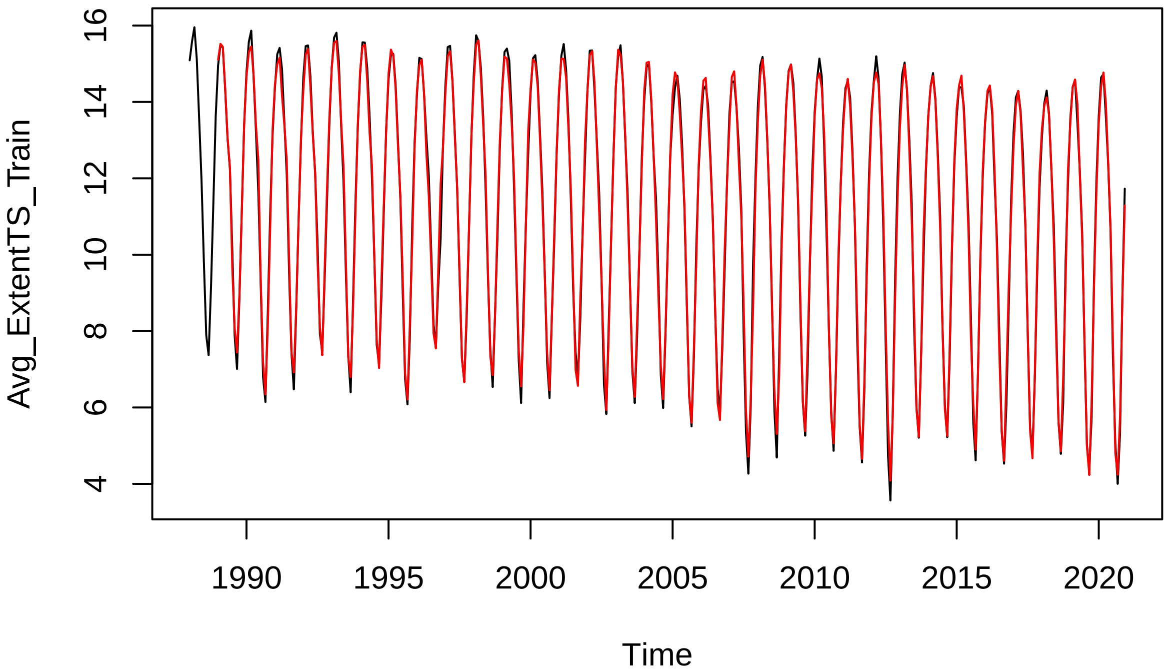
```
# no_trend on training data
residuals_no_trend_train <- as.vector(Avg_ExtentTS_Train[which(time(Avg_ExtentTS_Train)>=1989)]) - no_t
plot(residuals_no_trend_train, type="l")
```




```
acf(residuals_no_trend_train, lag.max=36)
```

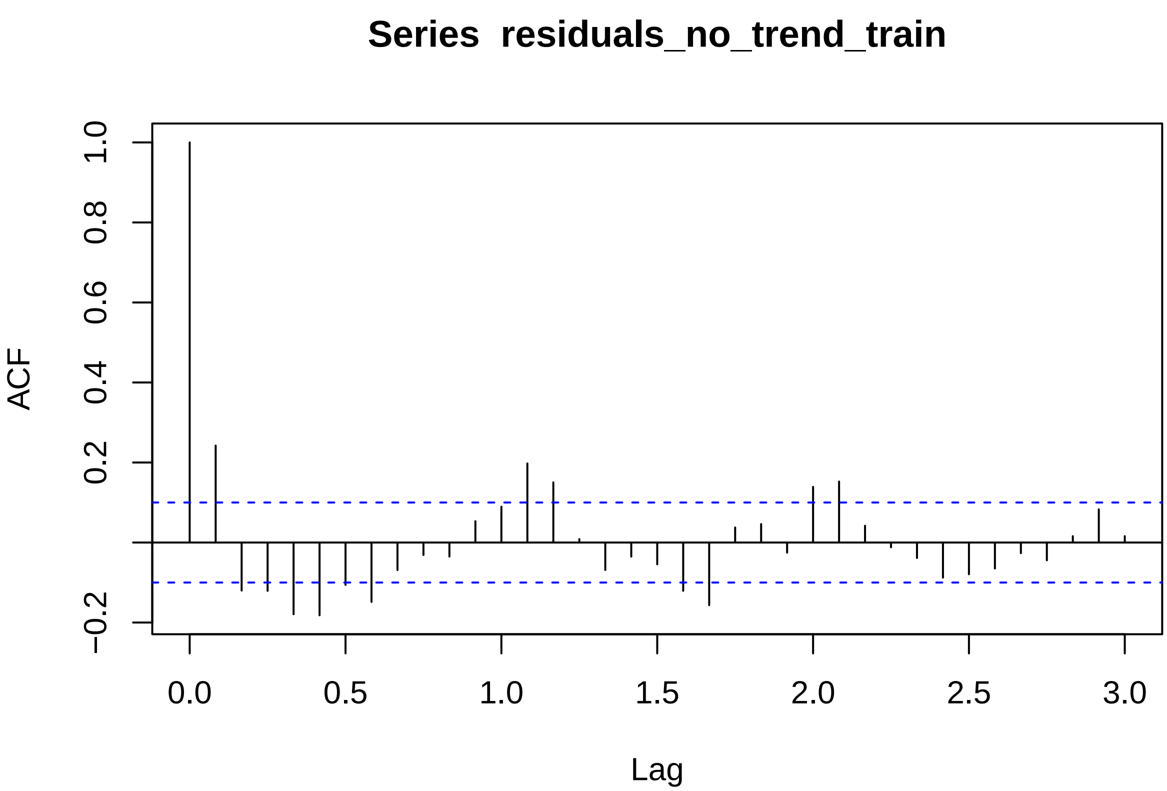



```
plot(Avg_ExtentTS_Train)
points(time(Avg_ExtentTS_Train)[which(time(Avg_ExtentTS_Train)>=1989)], no_trend$fitted[,1], type='l',
```

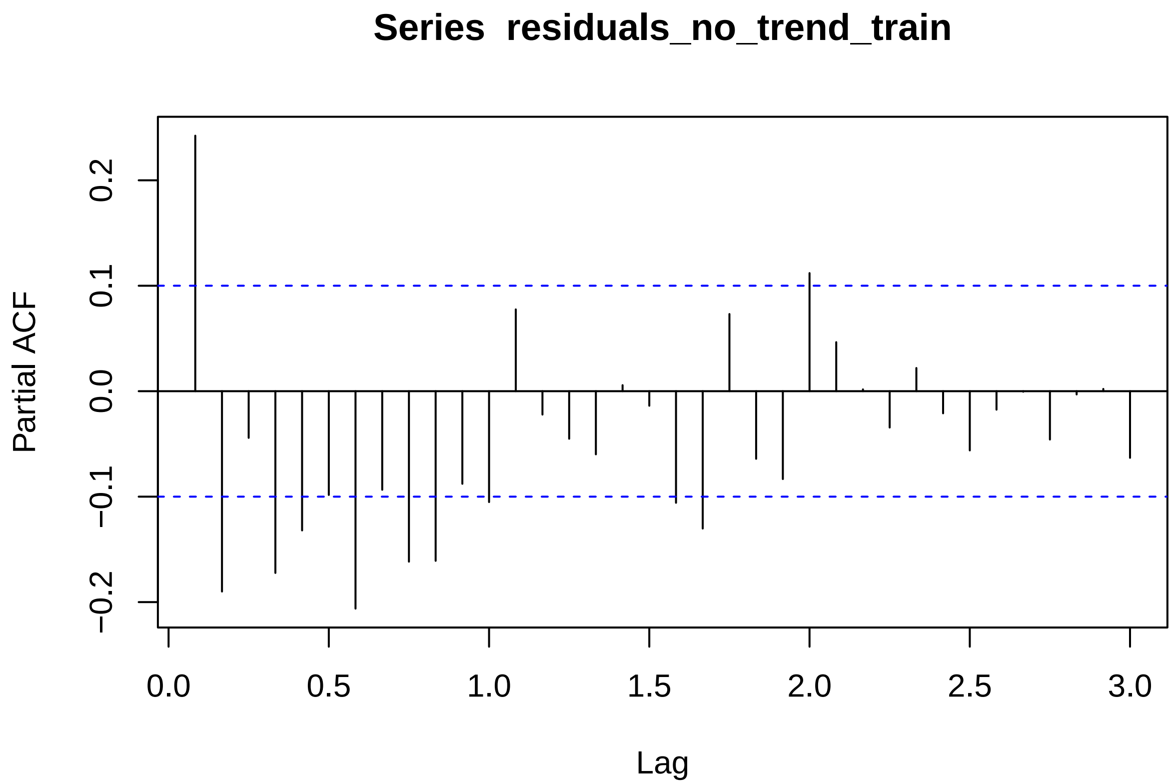



```
#plot(hw_additive)
```

```
#TODO COULD fit SARMA models on HW residuals but probably overkill
acf(residuals_no_trend_train, lag.max=36)
```



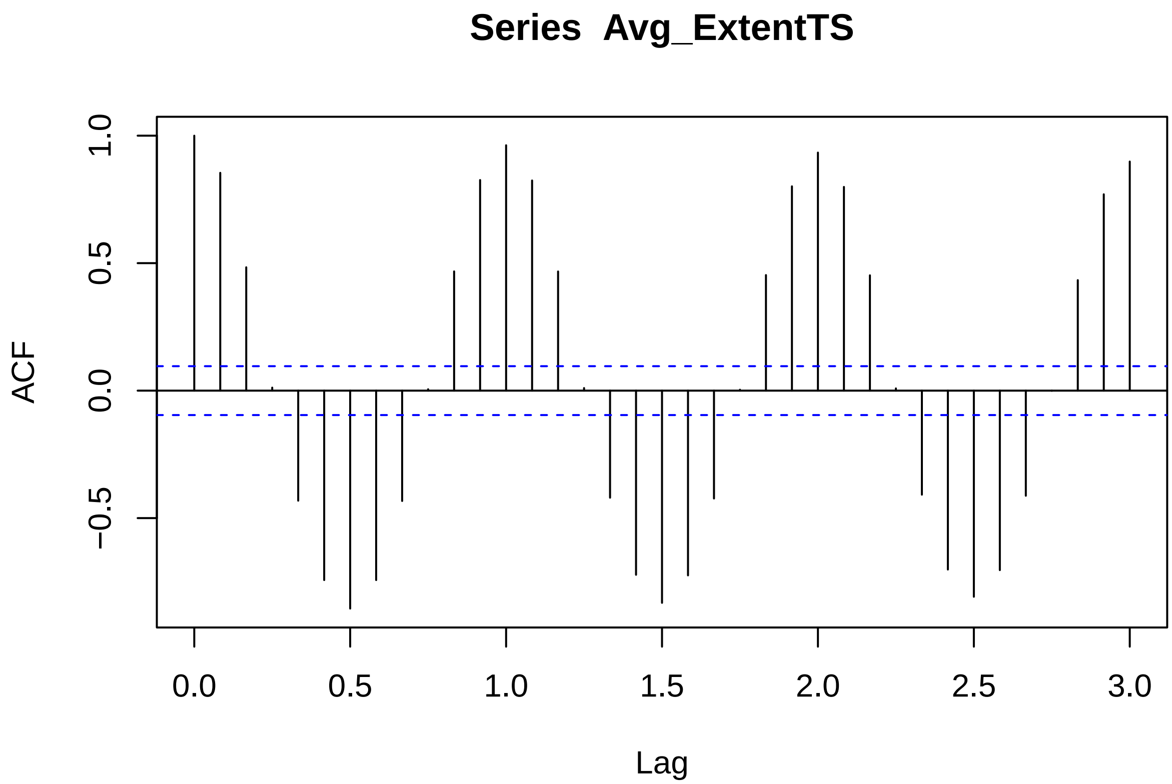

```
pacf(residuals_no_trend_train, lag.max=36)
```



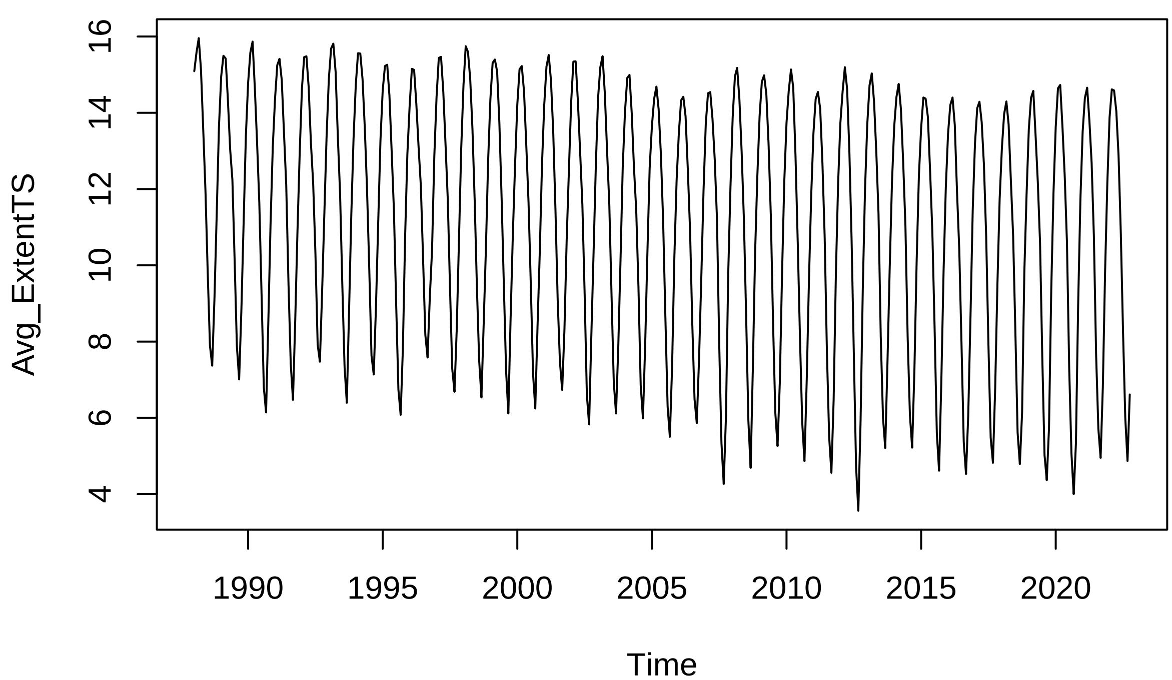
Differencing on Entire Data

Try differencing to remove non-stationarity.

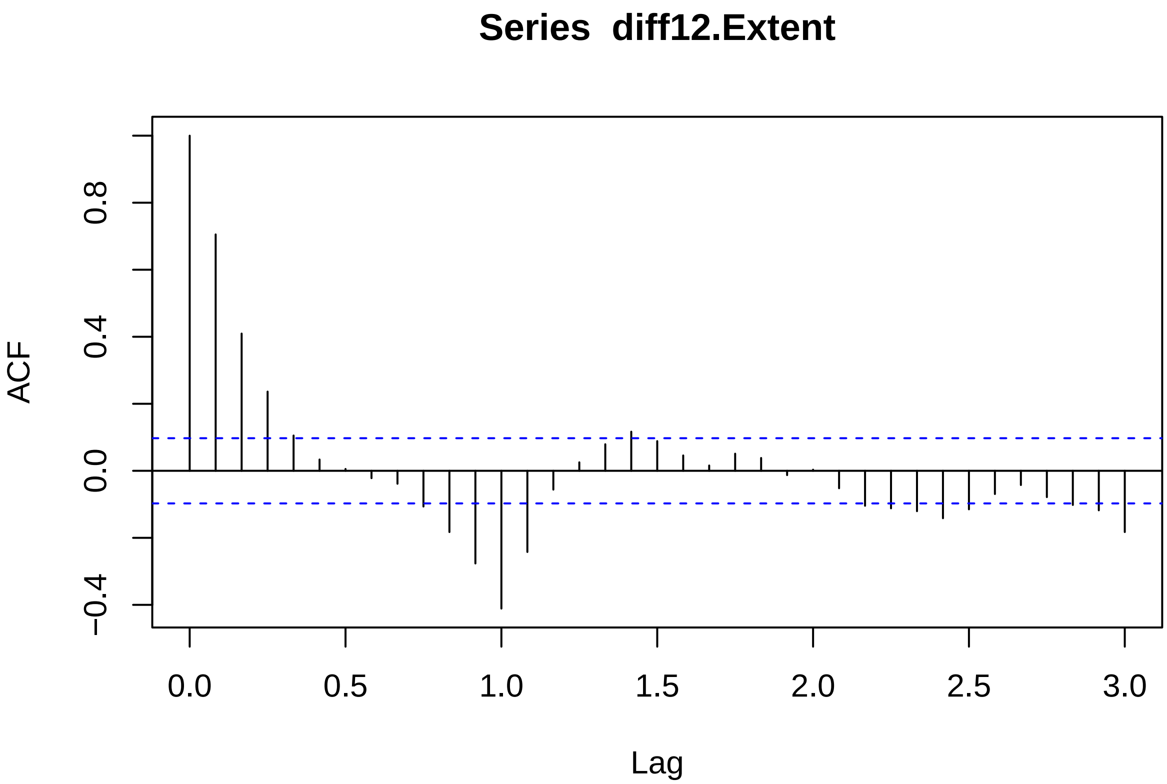
```
acf(Avg_ExtentTS, lag.max=36)
```



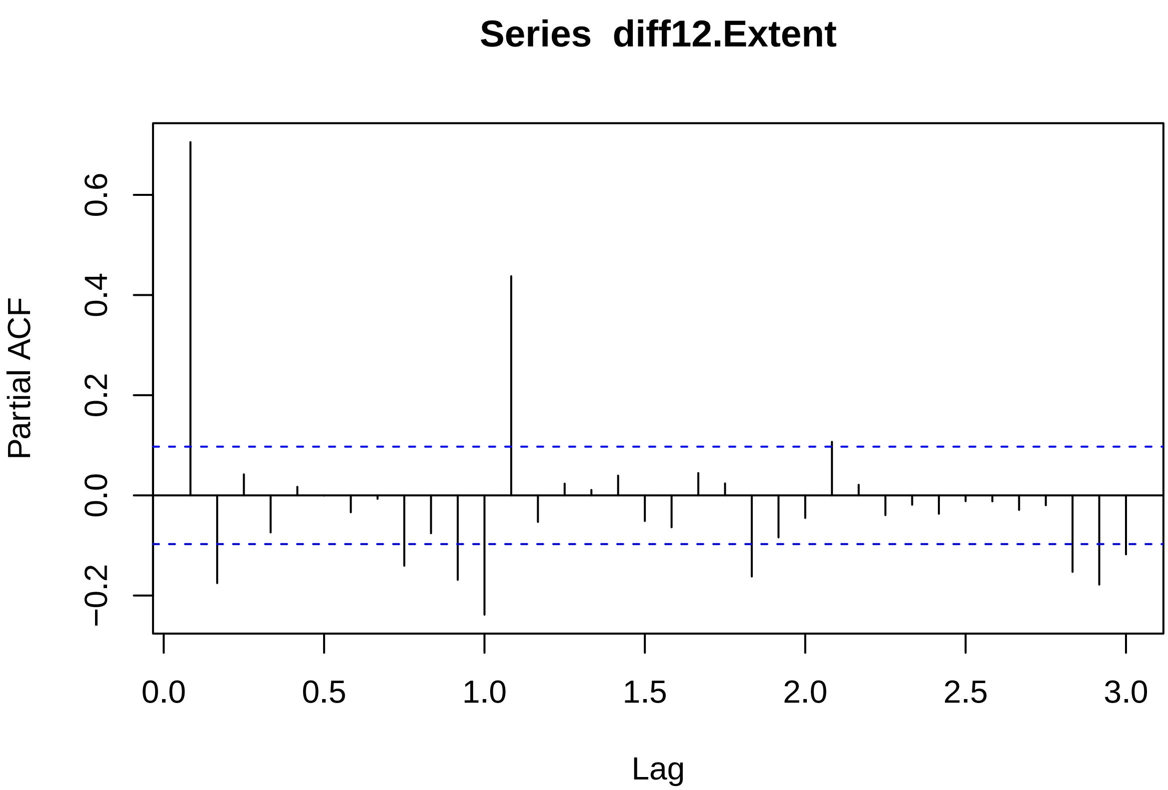

```r
plot(Avg_ExtentTS)
```



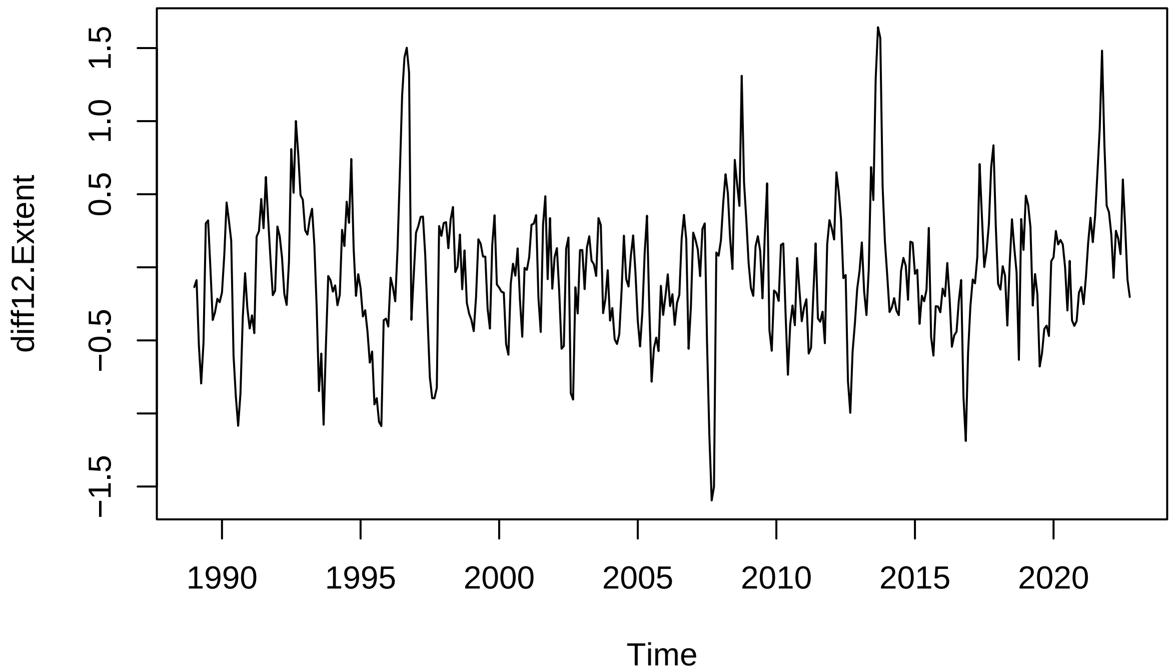

```r
#differencing in lag of season
diff12.Extent=diff(Avg_ExtentTS, lag=12)
acf(diff12.Extent, lag.max=36)
```



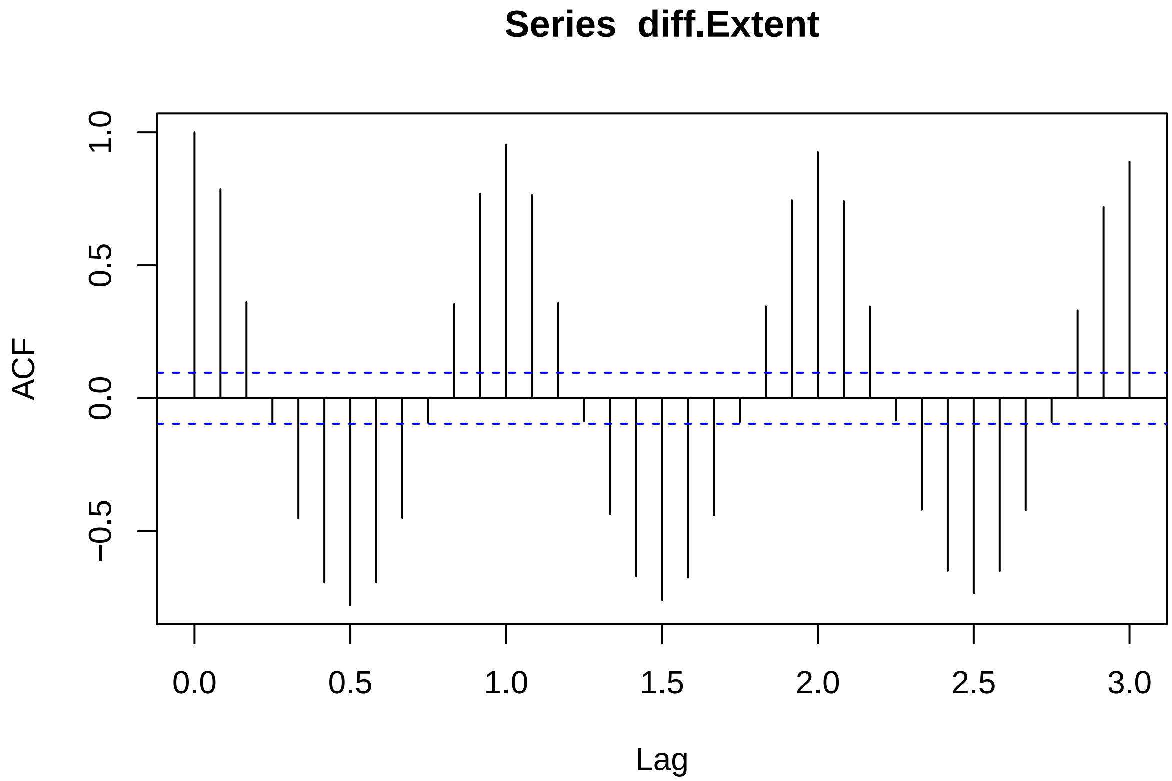

```
pacf(diff12.Extent, lag.max=36)
```



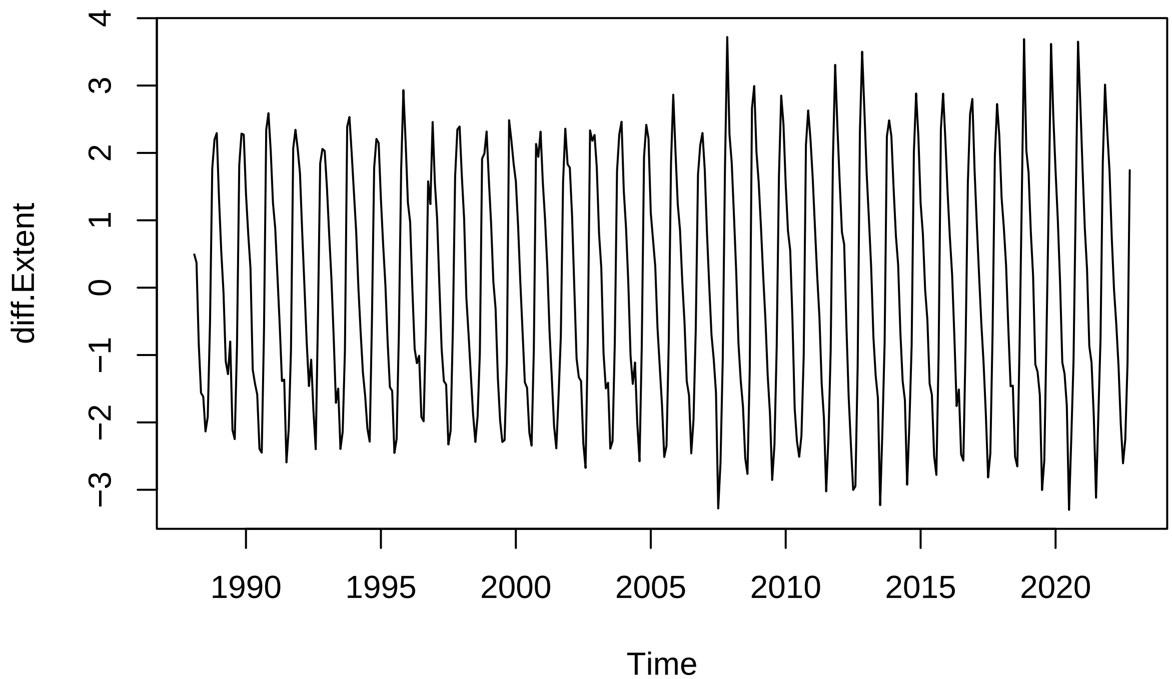

```
plot(diff12.Extent)
```



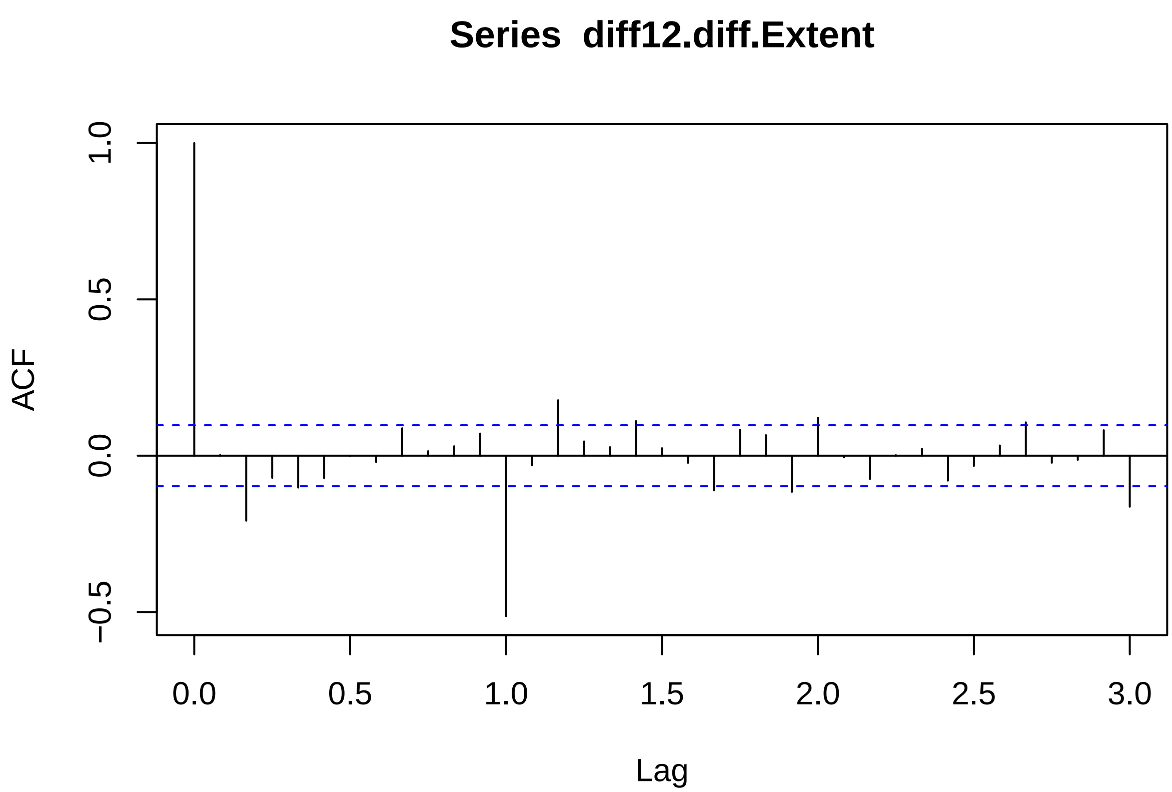

```
#regular differencing
diff.Extent=diff(Avg_ExtentTS)
acf(diff.Extent, lag.max=36)
```



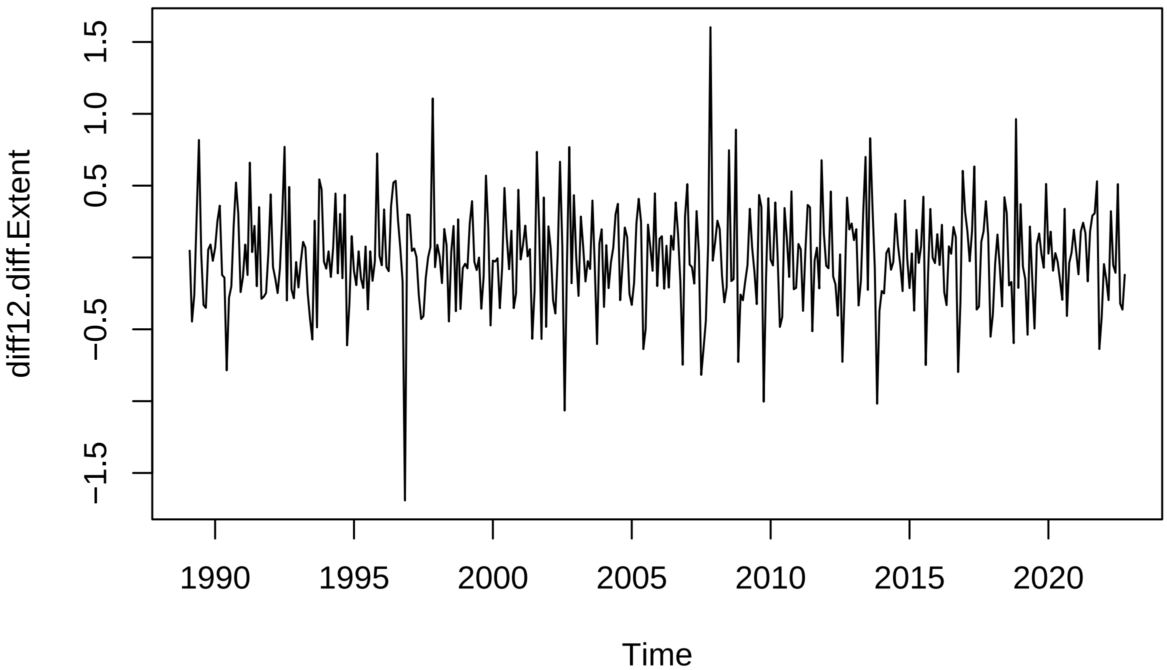

```
plot(diff.Extent)
```




```
#seasonal+regular differencing
diff12.diff.Extent=diff(diff12.Extent)
acf(diff12.diff.Extent, lag.max=36)
```

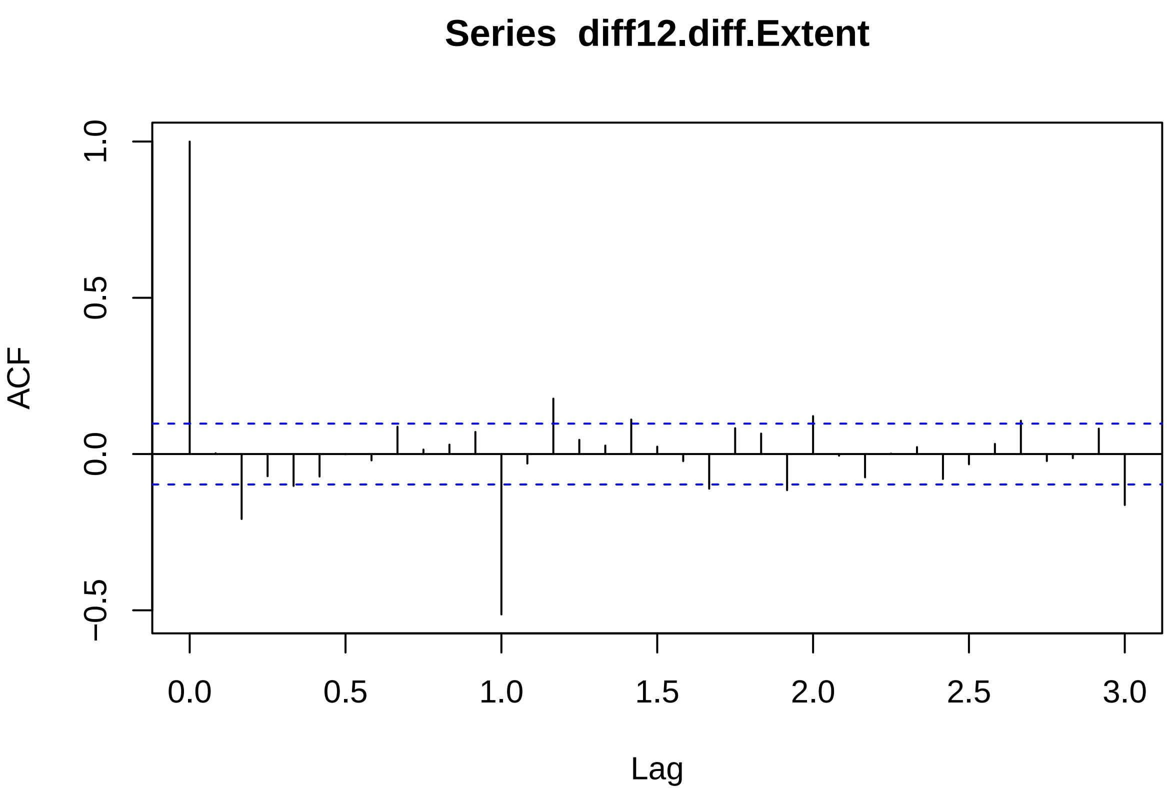



```
plot(diff12.diff.Extent)
```

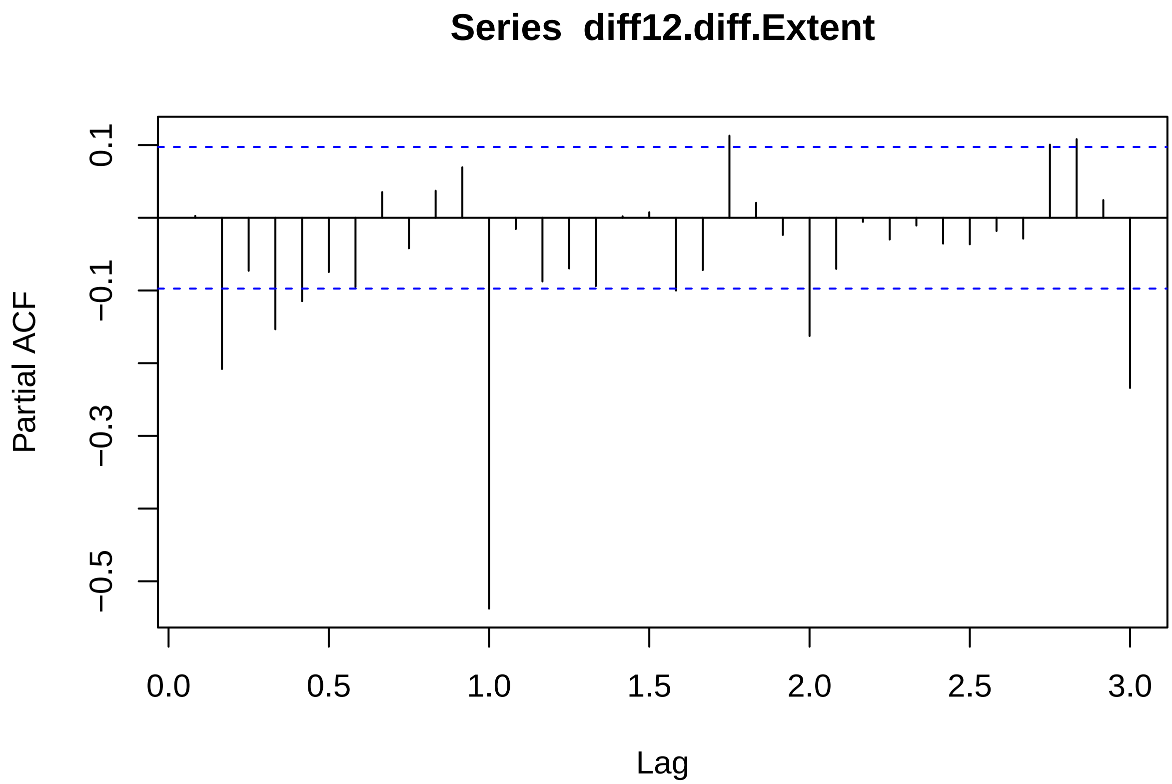


It seems that regression performs poorly in terms of removing non-stationarity compared to HW and Differencing. HW and differencing seem to perform similarly. For simplicity, we proceed with differencing.

```
acf(diff12.diff.Extent, lag.max=36)
```




```
pacf(diff12.diff.Extent, lag.max=36)
```

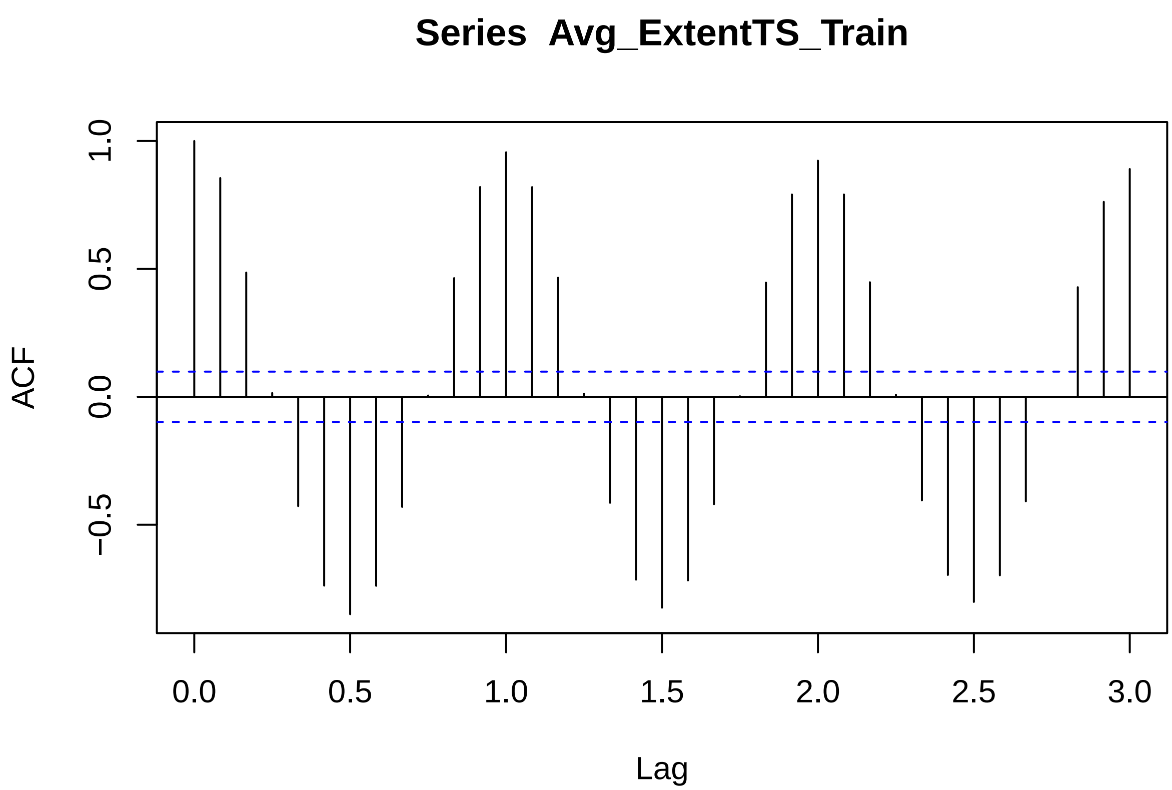


##

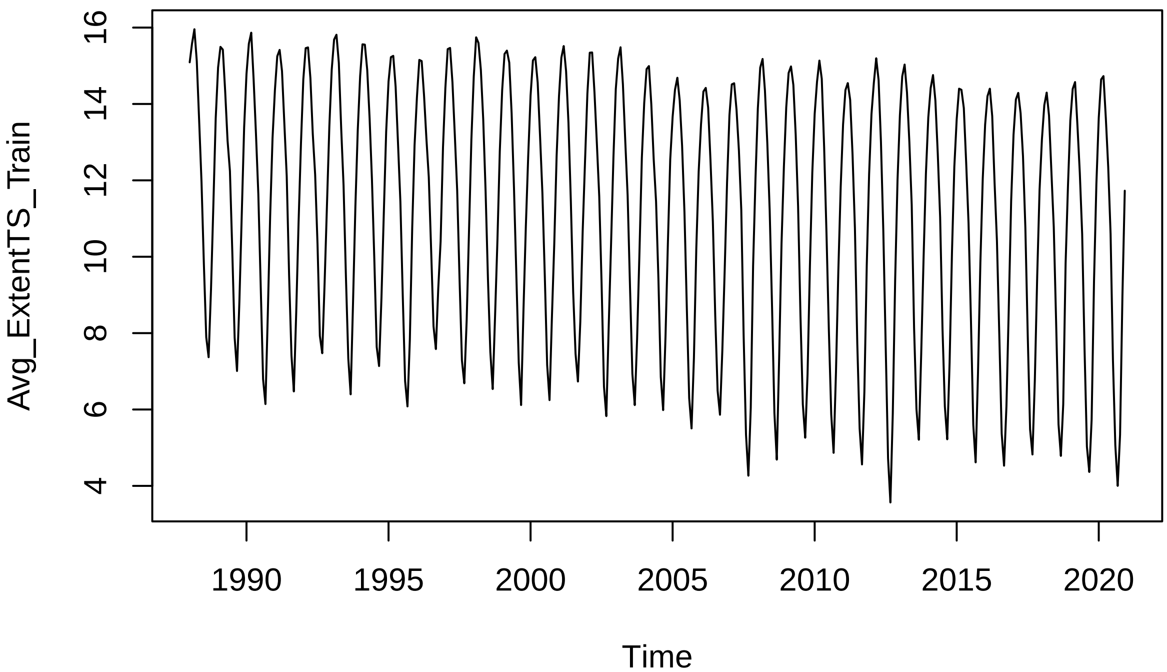
Differencing on Train Data

Try differencing to remove non-stationarity.

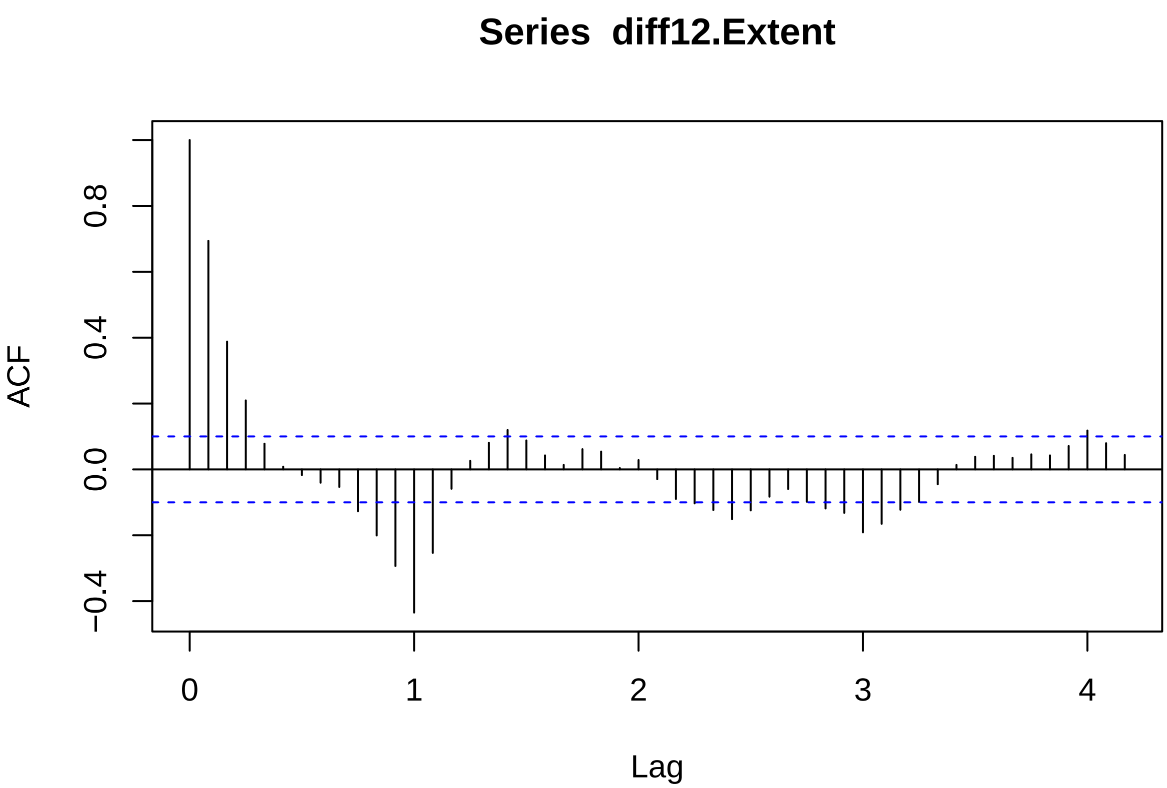
```
acf(Avg_ExtentTS_Train, lag.max=36)
```



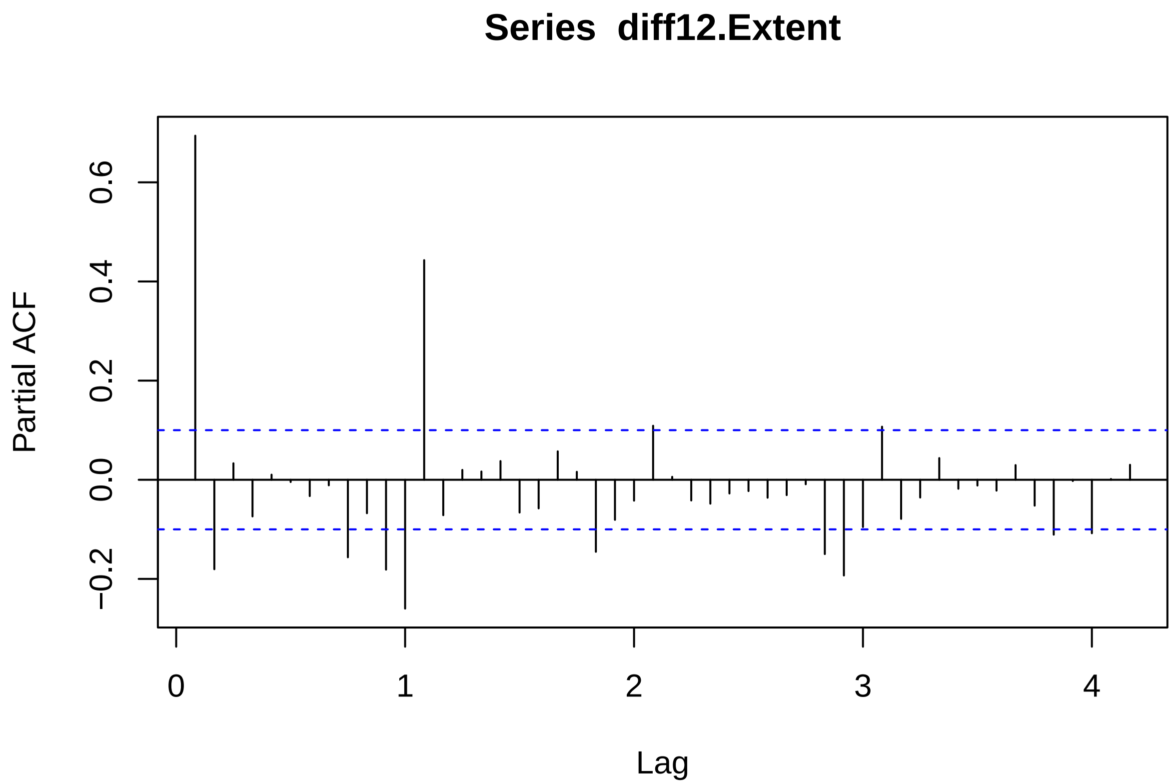
```
plot(Avg_ExtentTS_Train)
```



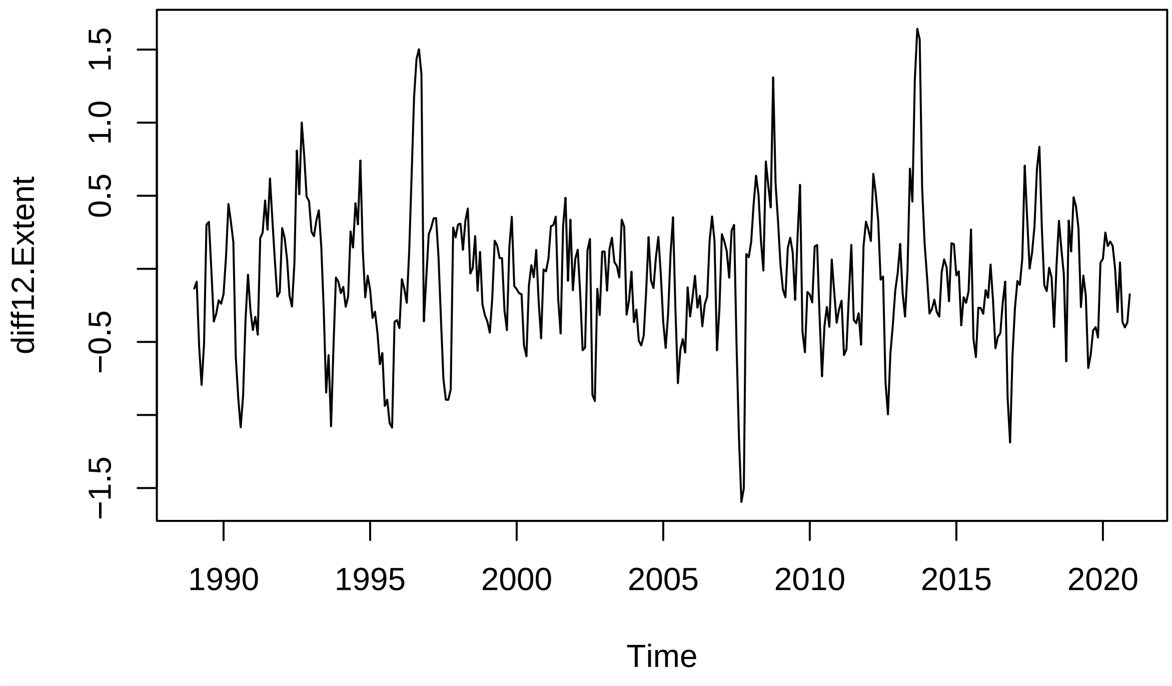

```
#differencing in lag of season
diff12.Extent=diff(Avg_ExtentTS_Train, lag=12)
acf(diff12.Extent, lag.max=50)
```



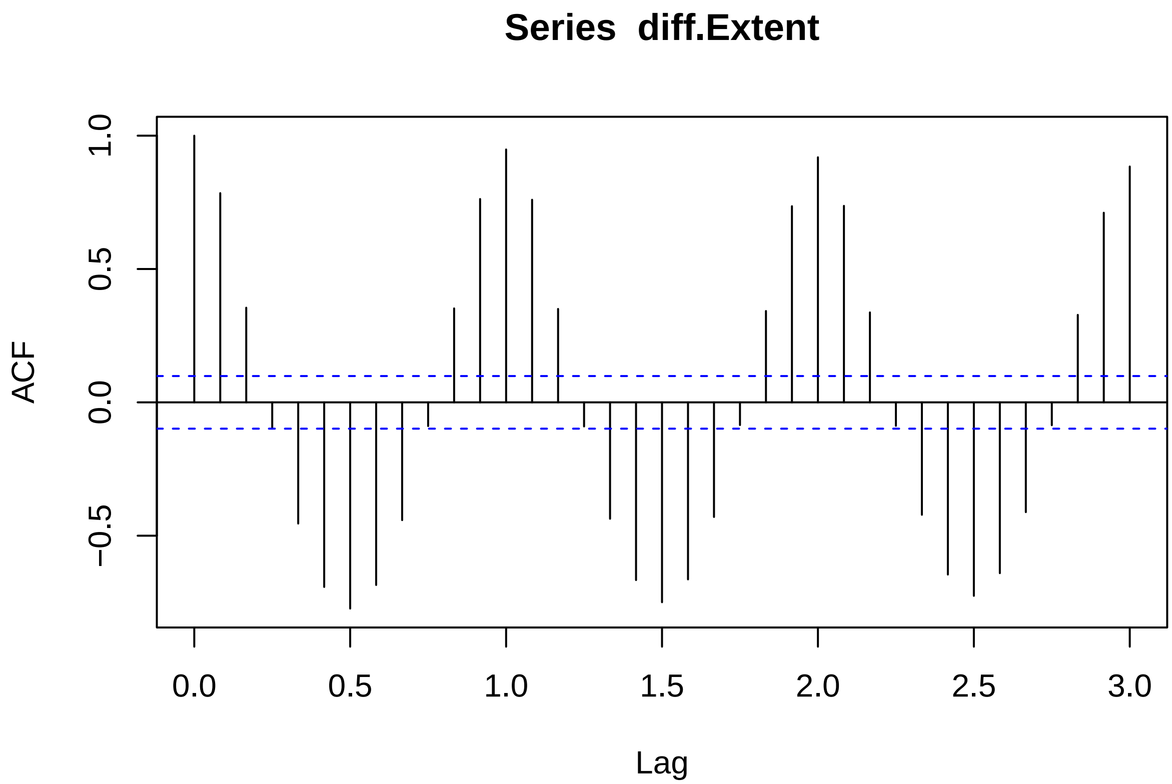

```
pacf(diff12.Extent, lag.max=50)
```



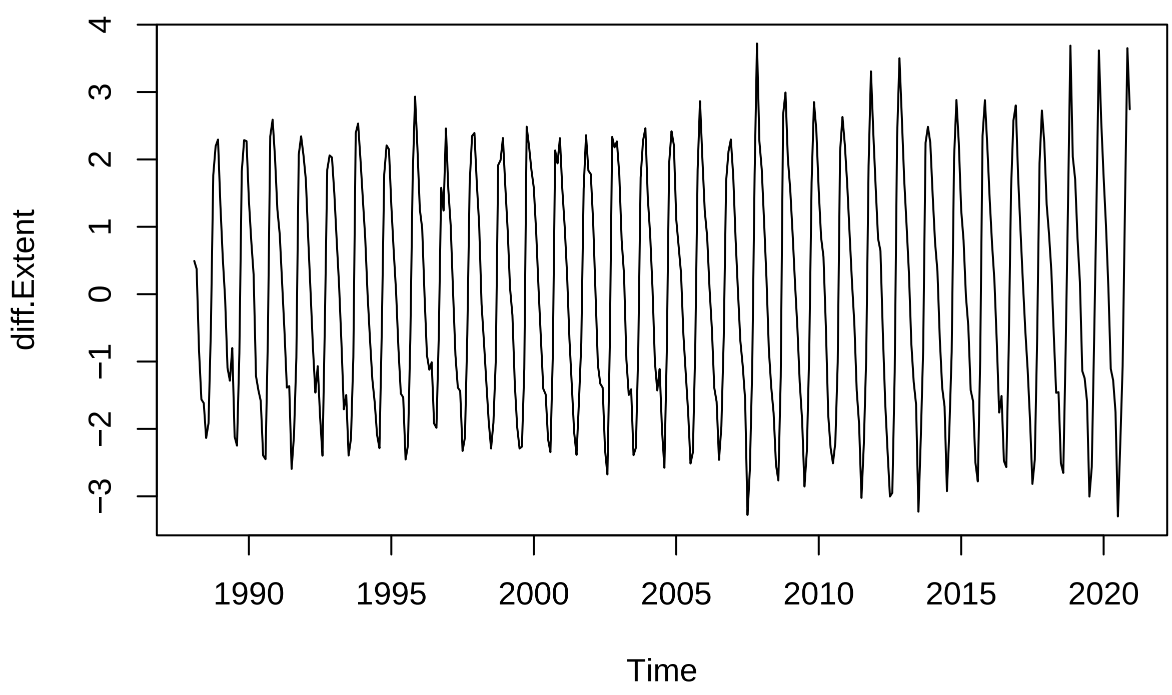

```
plot(diff12.Extent)
```



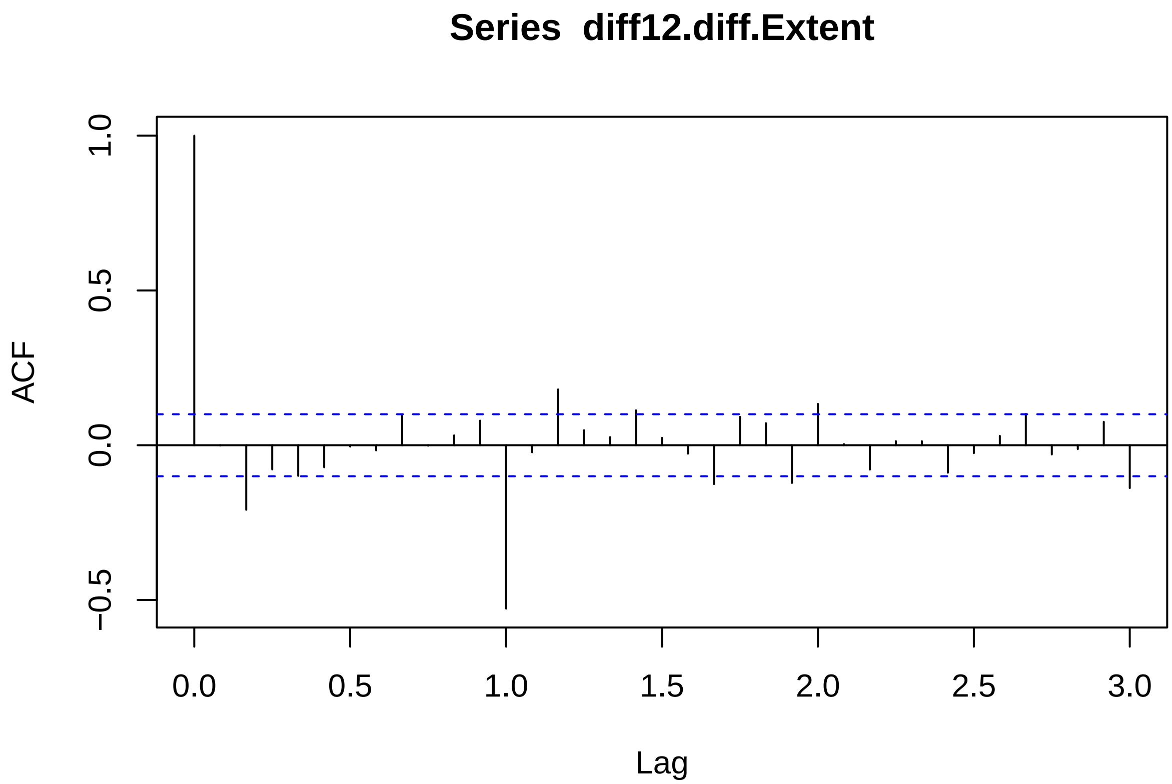

```
#regular differencing
diff.Extent=diff(Avg_ExtentTS_Train)
acf(diff.Extent, lag.max=36)
```



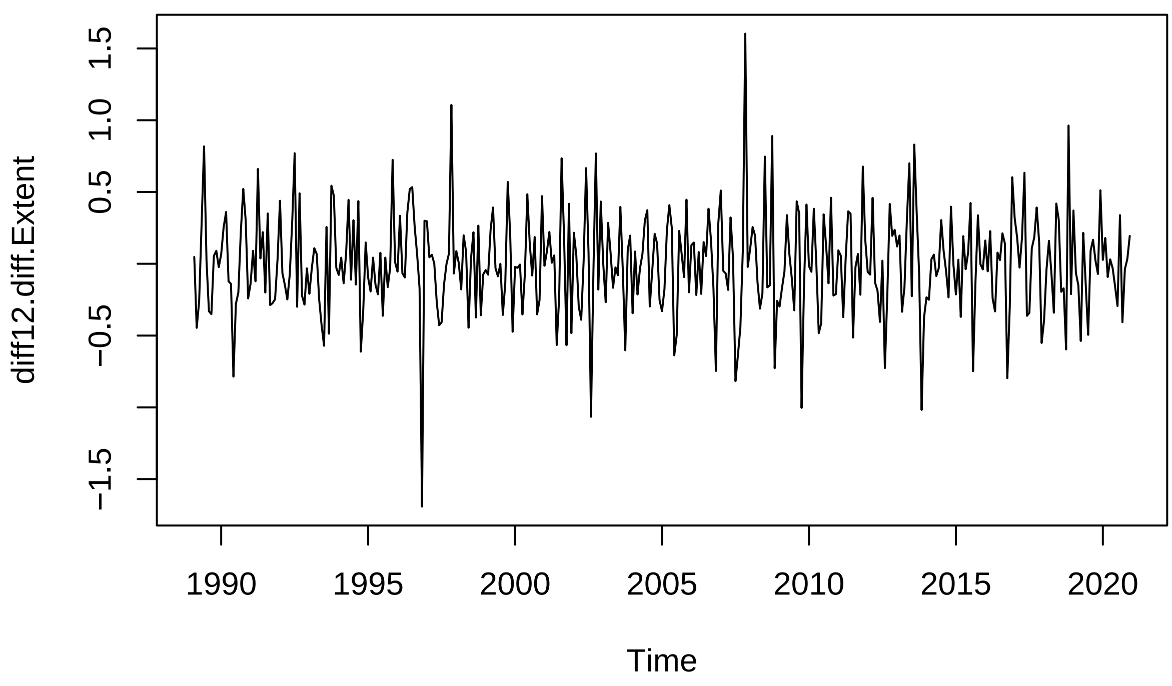

```r
plot(diff.Extent)
```




```r
# seasonal+regular differencing
diff12.diff.Extent=diff(diff12.Extent)
acf(diff12.diff.Extent, lag.max=36)
```

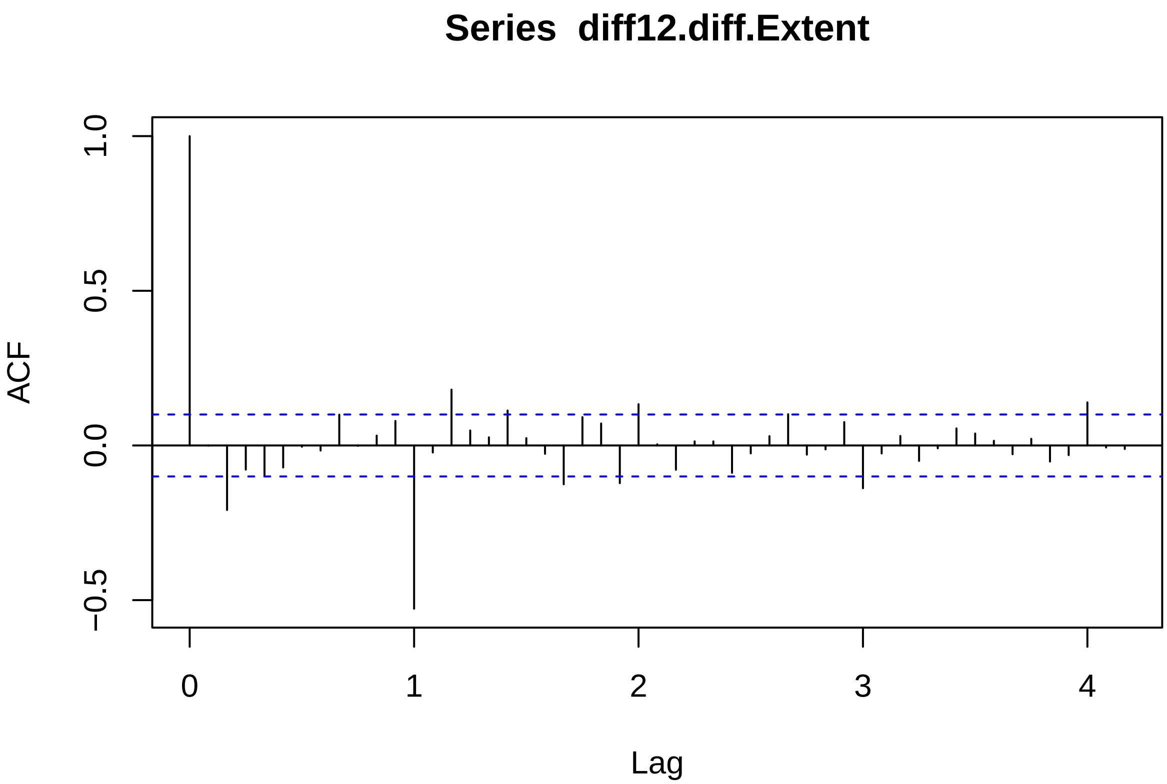



```
plot(diff12.diff.Extent)
```

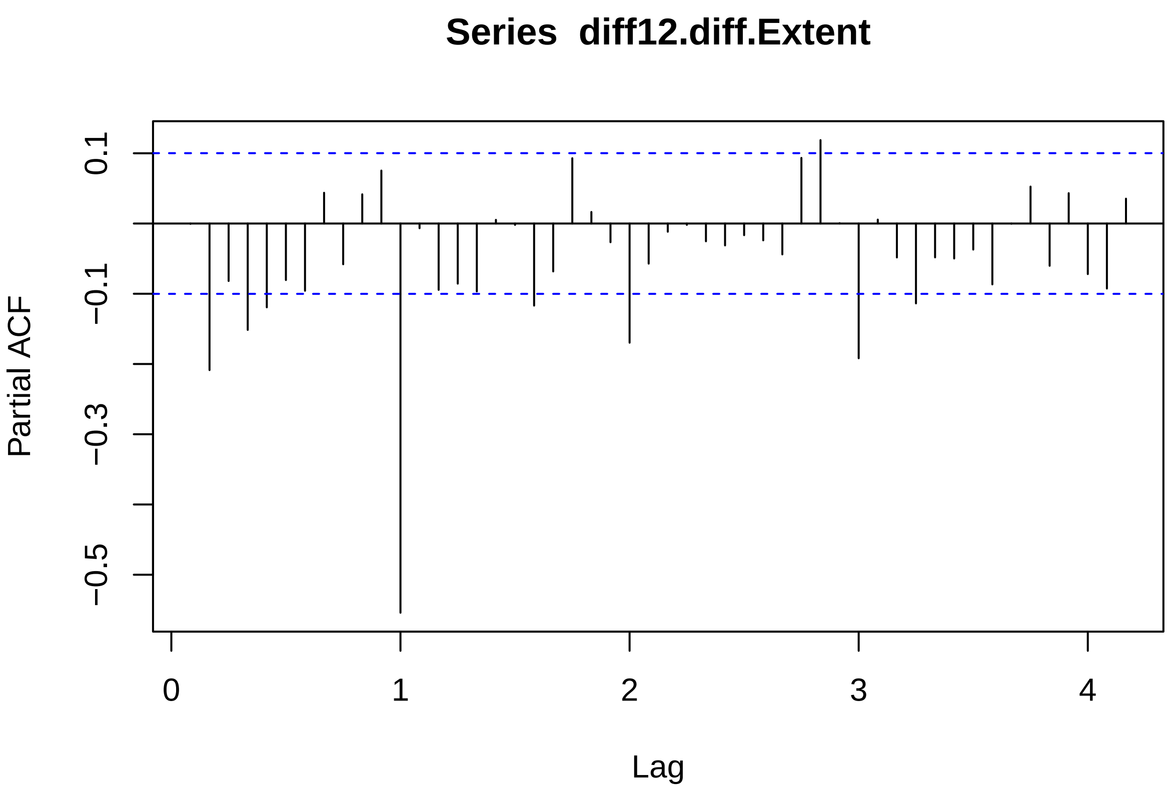


It seems that regression performs poorly in terms of removing non-stationarity compared to HW and Differencing. HW and differencing seem to perform similarly. For simplicity, we proceed with differencing. #TODO reword this

```r
acf(diff12.diff.Extent, lag.max=50)
```




```r
pacf(diff12.diff.Extent, lag.max=50)
```

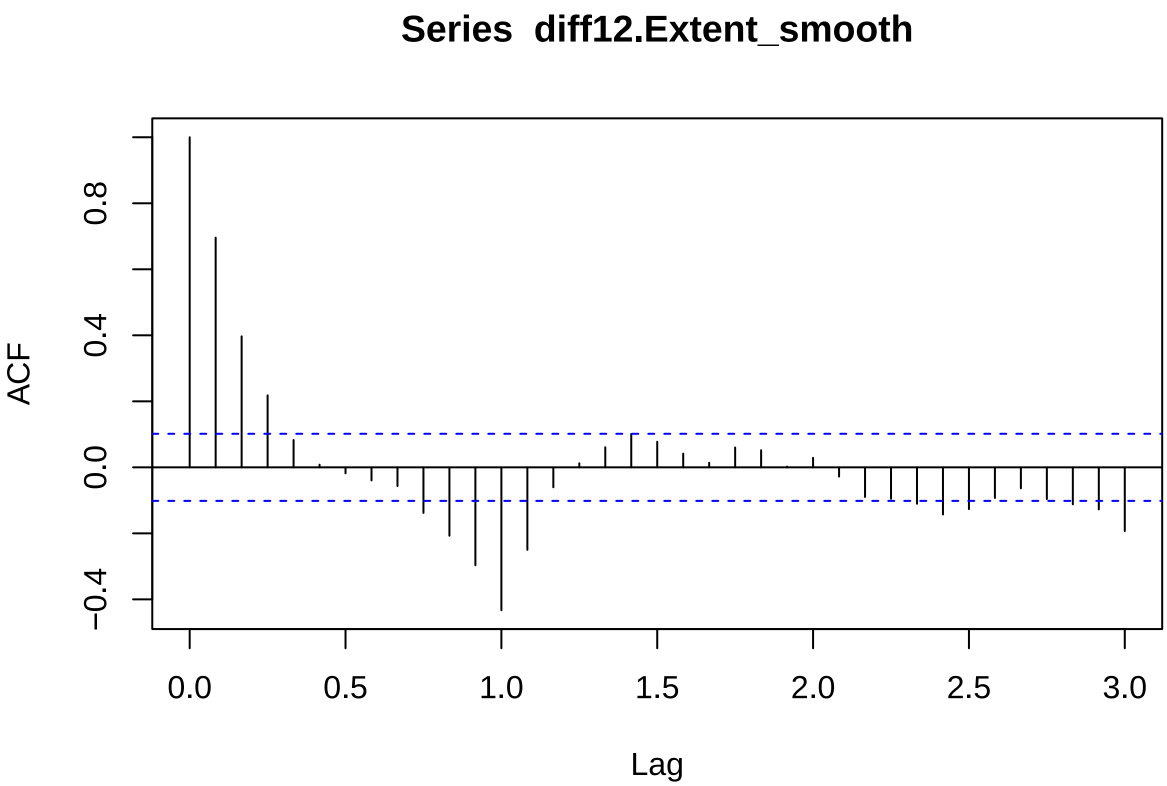


We propose the following models: #TODO

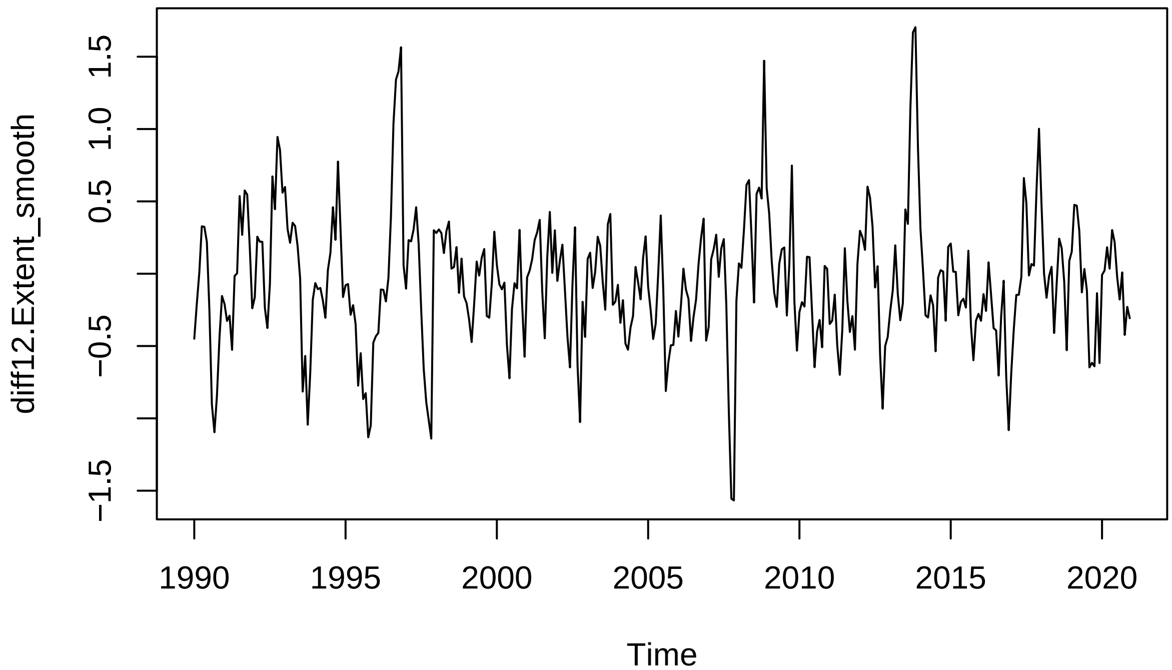
Smoothing, followed by differencing

Try smoothing before differencing to see effect on acf

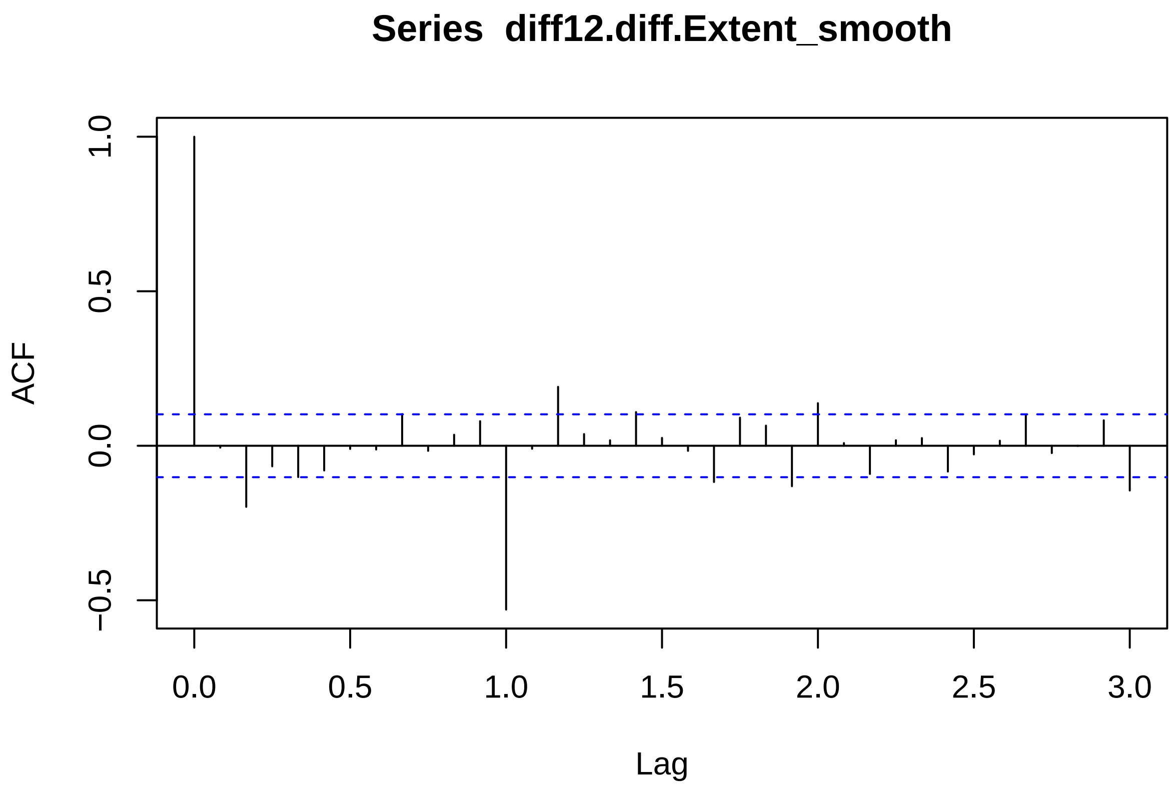
```
smoothing <- HoltWinters(Avg_ExtentTS_Train, season="additive")
smoothed <- smoothing$fitted[,1]
diff12.Extent_smooth=diff(smoothed, lag=12)
acf(diff12.Extent_smooth, lag.max=36)
```



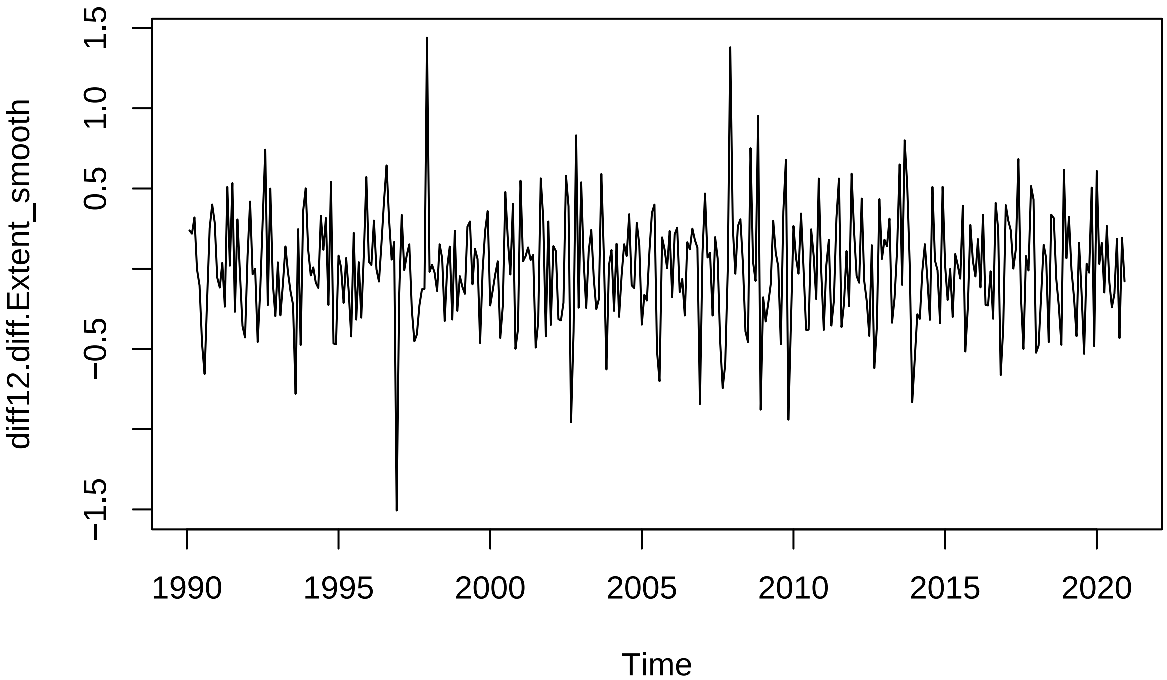

```
plot(diff12.Extent_smooth)
```



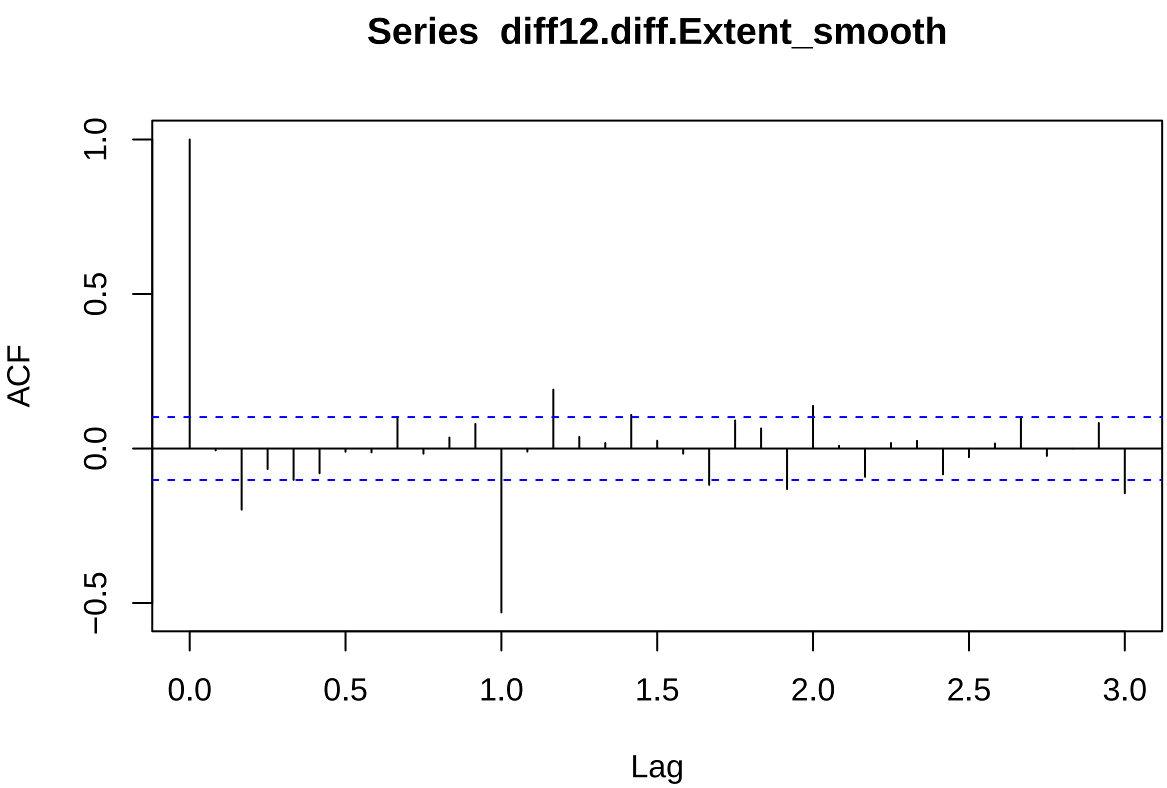

```
diff12.diff.Extent_smooth=diff(diff12.Extent_smooth)
acf(diff12.diff.Extent_smooth, lag.max=36)
```



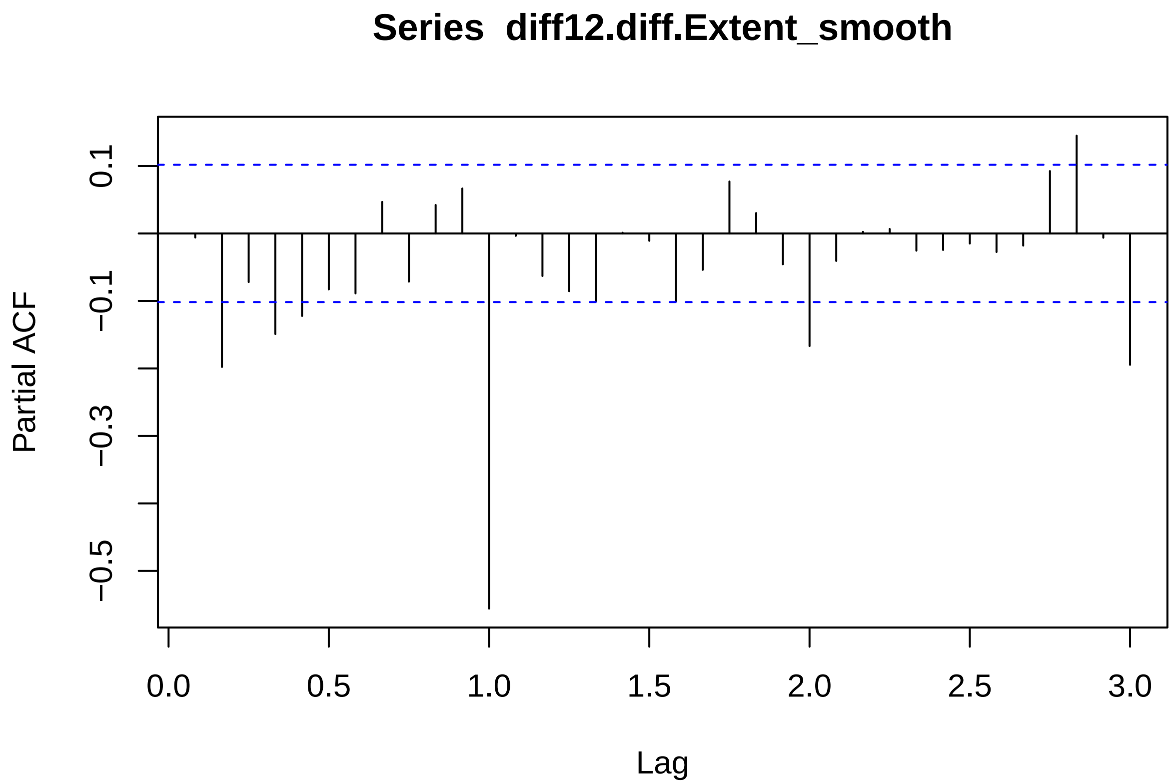

```
plot(diff12.diff.Extent_smooth)
```




```
acf(diff12.diff.Extent_smooth, lag.max=36)
```




```
pacf(diff12.diff.Extent_smooth, lag.max=36)
```

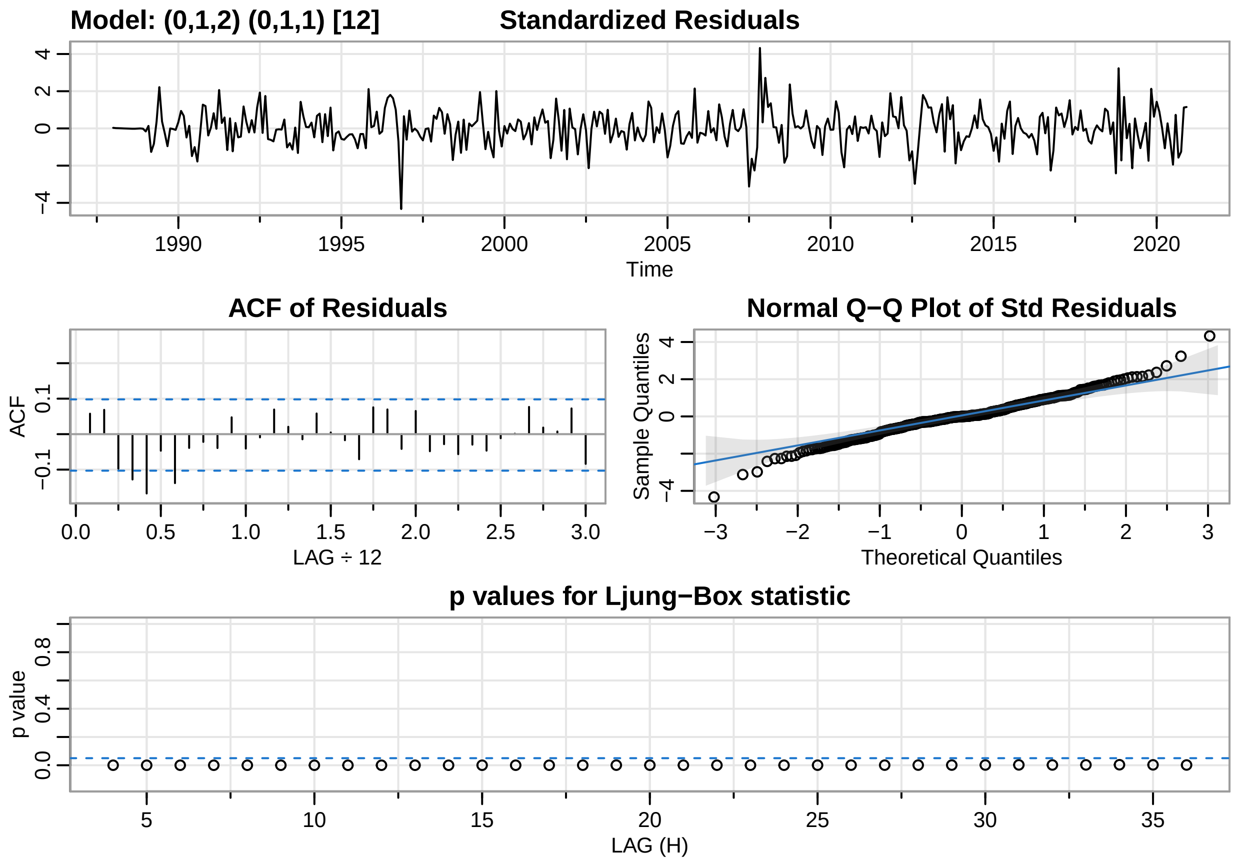


Model Fitting

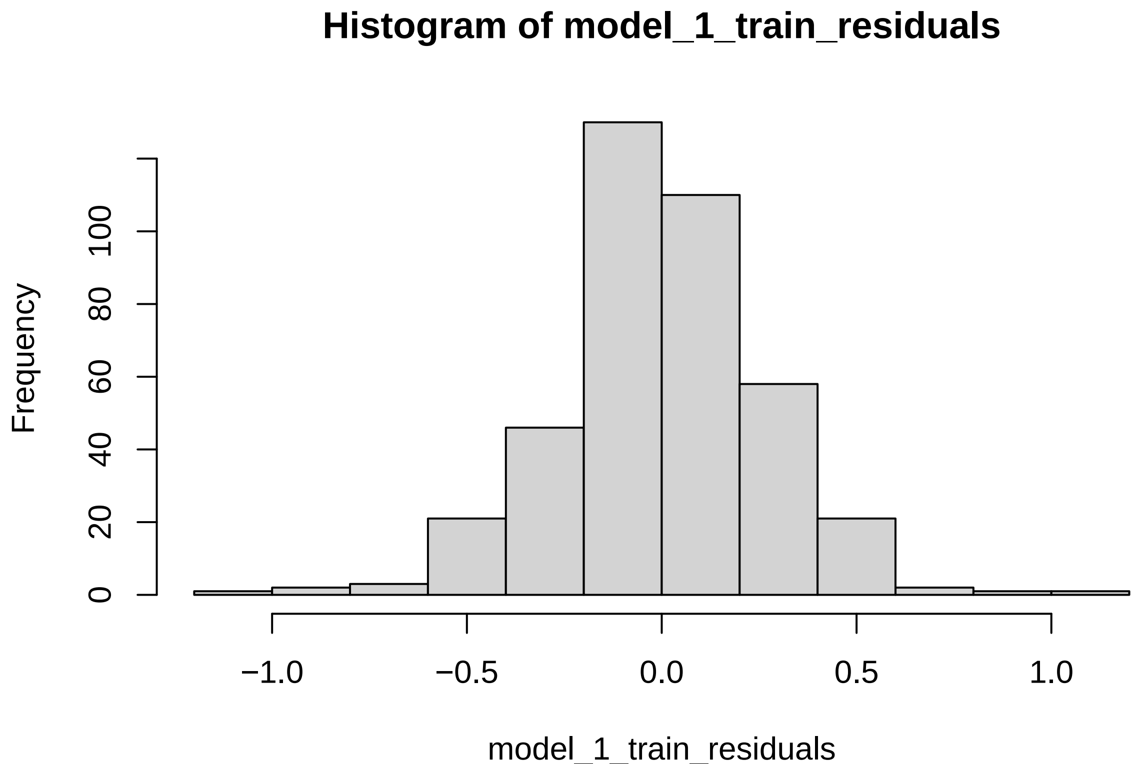
```
library(astsa)

#SARIMA(0,1,2)x(0,1,1)_12
model_1_train <- sarima(Avg_ExtentTS_Train, p=0, d=1, q=2, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -1.046835
## iter   2 value -1.270895
## iter   3 value -1.286671
## iter   4 value -1.288786
## iter   5 value -1.299437
## iter   6 value -1.301999
## iter   7 value -1.302786
## iter   8 value -1.302845
## iter   9 value -1.302850
## iter  10 value -1.302854
## iter  11 value -1.302854
## iter  11 value -1.302854
## iter  11 value -1.302854
## final  value -1.302854
## converged
## initial  value -1.299104
## iter   2 value -1.299174
## iter   3 value -1.299223
## iter   4 value -1.299225
## iter   5 value -1.299225
## iter   5 value -1.299225
```

```
## iter   5 value -1.299225
## final  value -1.299225
## converged
```




```
model_1_train_residuals = resid(model_1_train$fit)
hist(model_1_train_residuals)
```



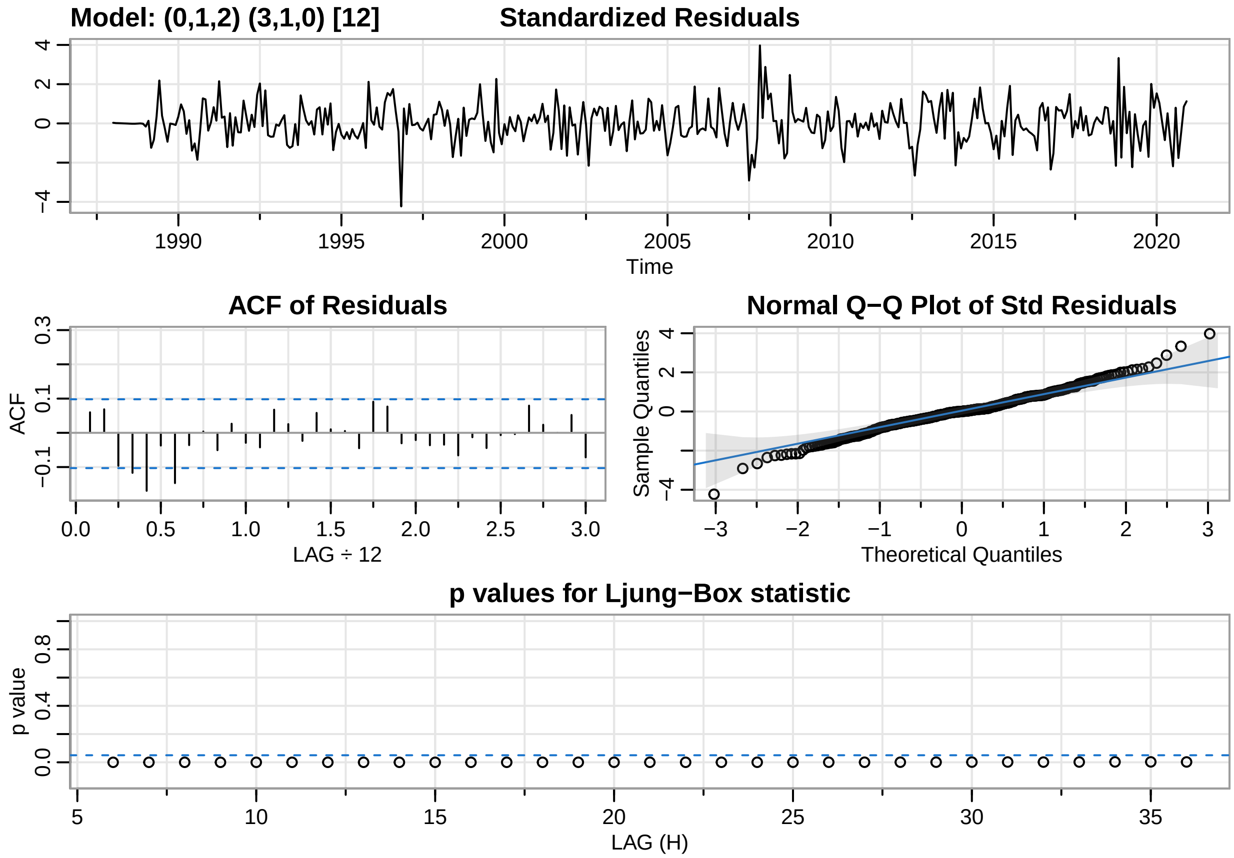

```
shapiro.test(model_1_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_1_train_residuals
## W = 0.98032, p-value = 3.21e-05
```

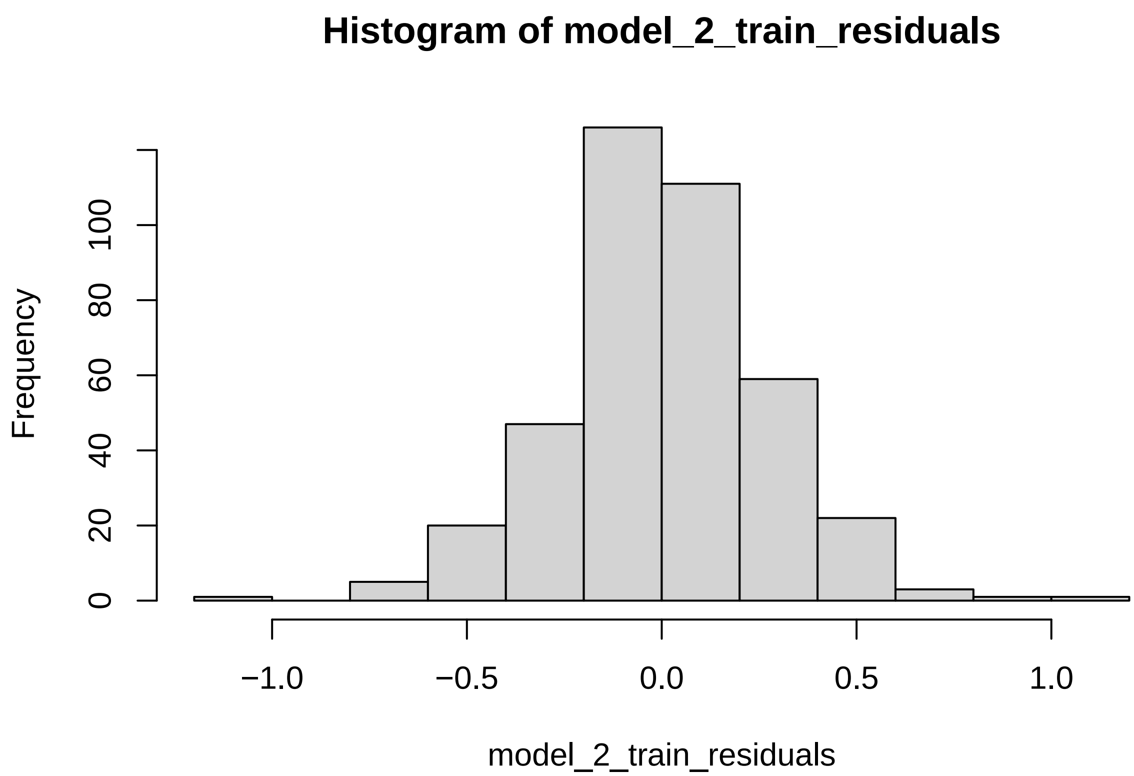
```
#SARIMA(0,1,2)x(3,1,0)_12
model_2_train <- sarima(Avg_ExtentTS_Train, p=0, d=1, q=2, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.039490
## iter   2 value -1.176658
## iter   3 value -1.269525
## iter   4 value -1.294457
## iter   5 value -1.299214
## iter   6 value -1.299421
## iter   7 value -1.299429
## iter   8 value -1.299429
## iter   8 value -1.299429
## iter   8 value -1.299429
## final  value -1.299429
## converged
## initial  value -1.297378
## iter   2 value -1.297439
## iter   3 value -1.297448
## iter   4 value -1.297451
## iter   5 value -1.297451
## iter   6 value -1.297451
## iter   6 value -1.297451
```

```
## iter   6 value -1.297451
## final  value -1.297451
## converged
```




```
model_2_train_residuals = resid(model_2_train$fit)
hist(model_2_train_residuals)
```



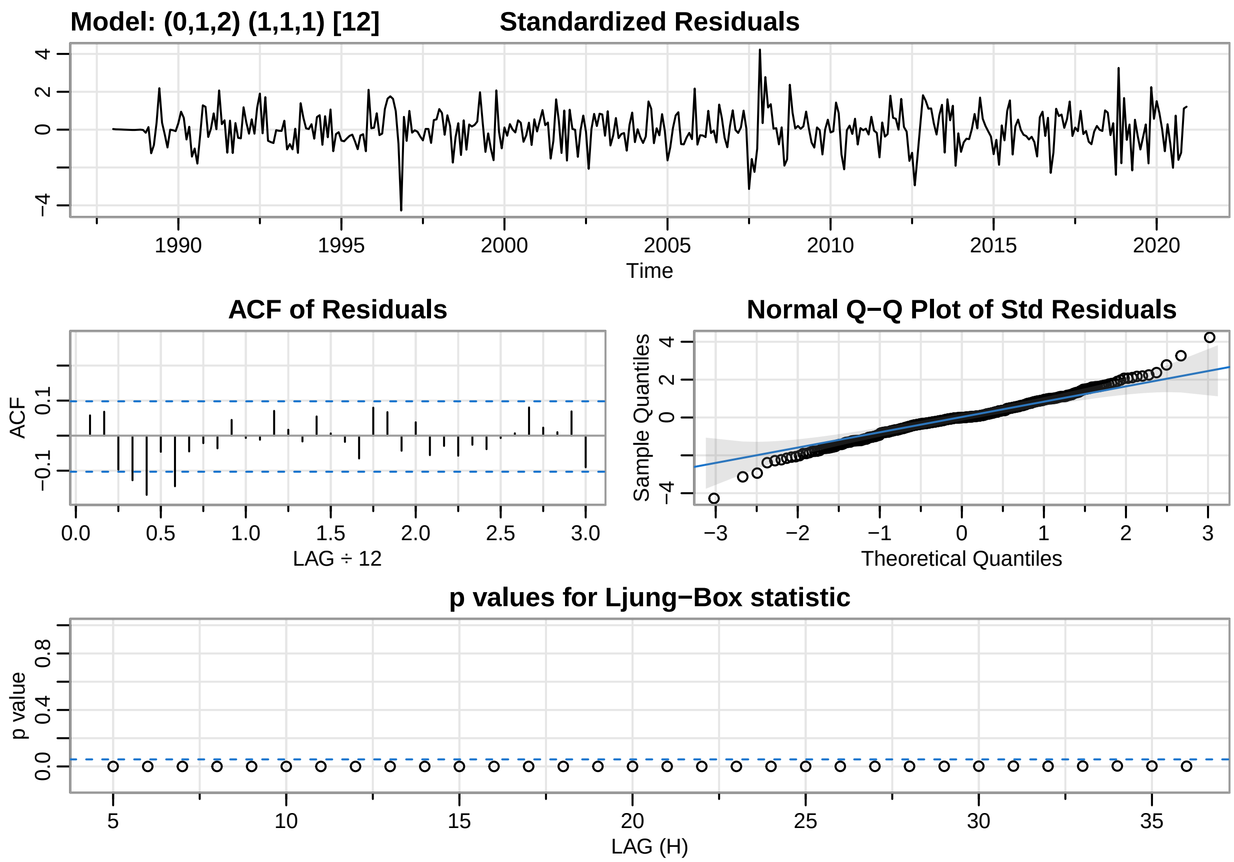

```
shapiro.test(model_2_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_2_train_residuals
## W = 0.98446, p-value = 0.0002963
```

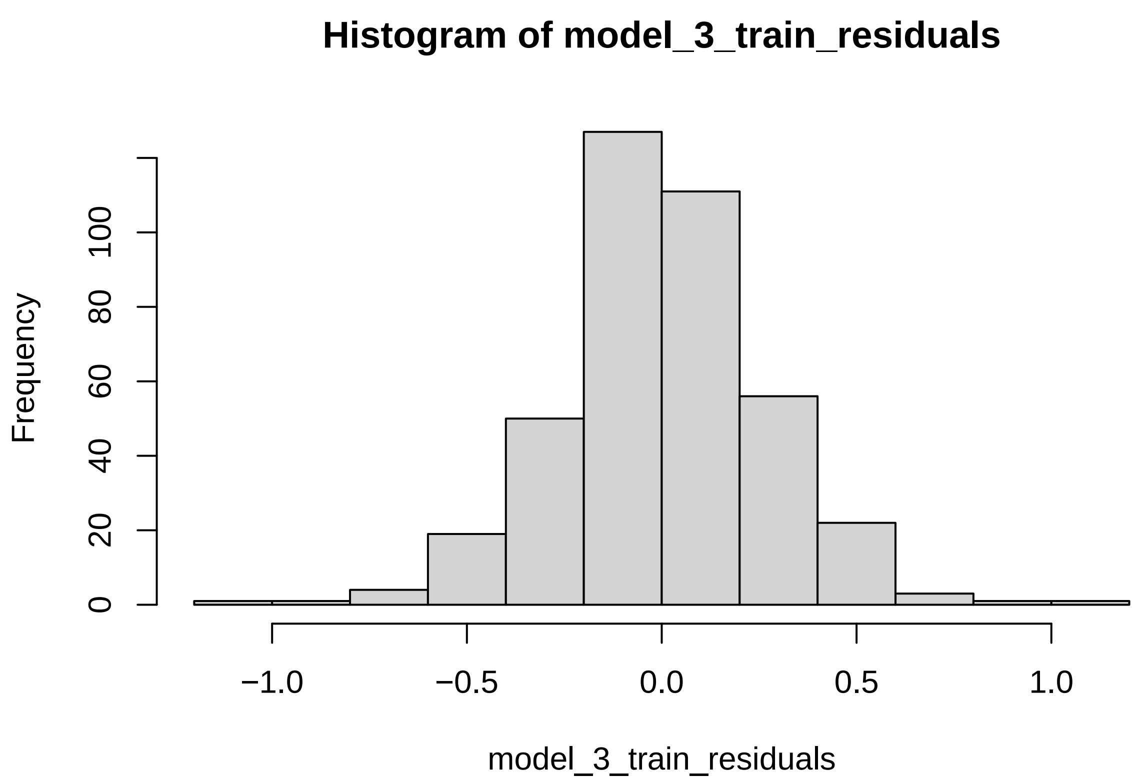
```
#SARIMA(0,1,2)x(1,1,1)_12
model_3_train <- sarima(Avg_ExtentTS_Train, p=0, d=1, q=2, P=1, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -1.044499
## iter   2 value -1.237721
## iter   3 value -1.282334
## iter   4 value -1.287558
## iter   5 value -1.293725
## iter   6 value -1.293955
## iter   7 value -1.294047
## iter   8 value -1.294078
## iter   9 value -1.294083
## iter  10 value -1.294083
## iter  10 value -1.294083
## final  value -1.294083
## converged
## initial  value -1.299017
## iter   2 value -1.299579
## iter   3 value -1.300082
## iter   4 value -1.300197
## iter   5 value -1.300210
## iter   6 value -1.300211
```

```
## iter   7 value -1.300211
## iter   8 value -1.300211
## iter   8 value -1.300211
## iter   8 value -1.300211
## final  value -1.300211
## converged
```




```
model_3_train_residuals = resid(model_3_train$fit)
hist(model_3_train_residuals)
```



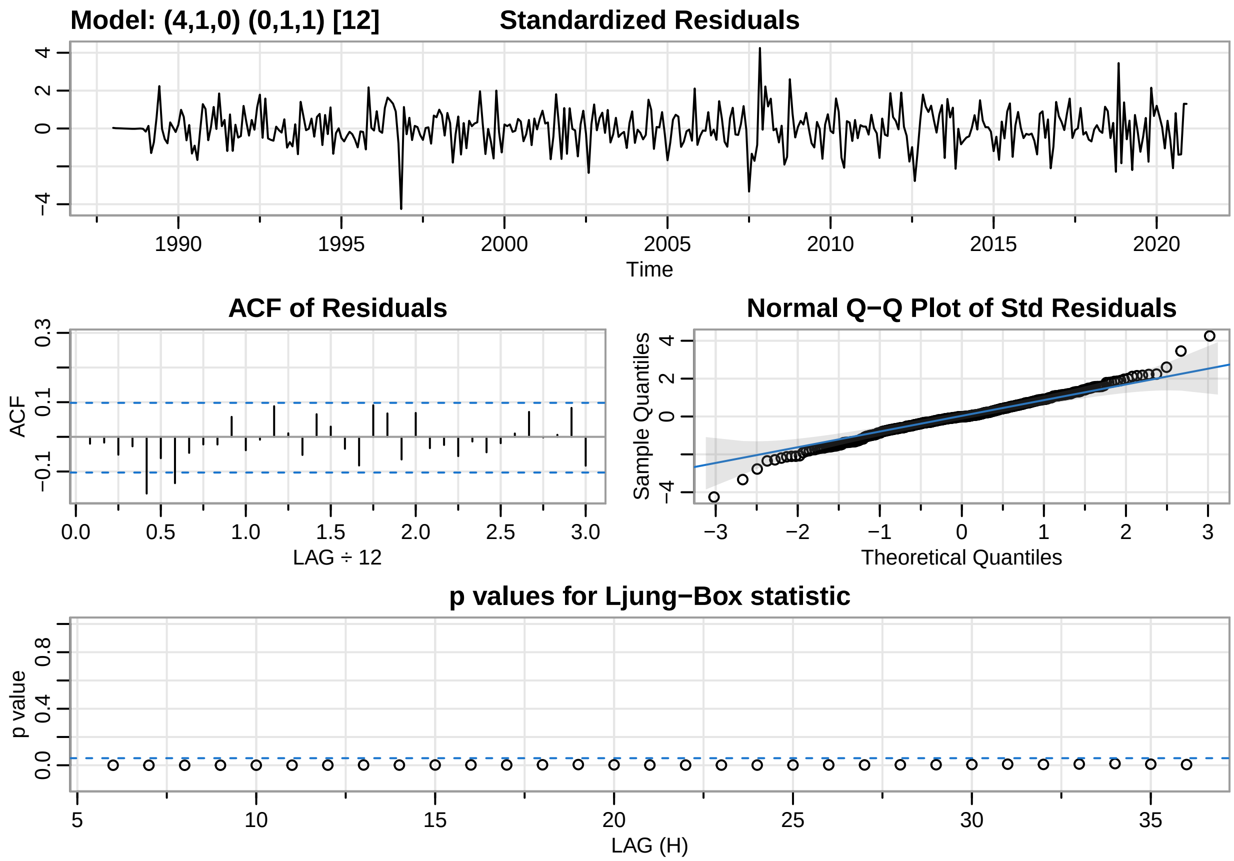

```
shapiro.test(model_3_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_3_train_residuals
## W = 0.98178, p-value = 6.856e-05
```

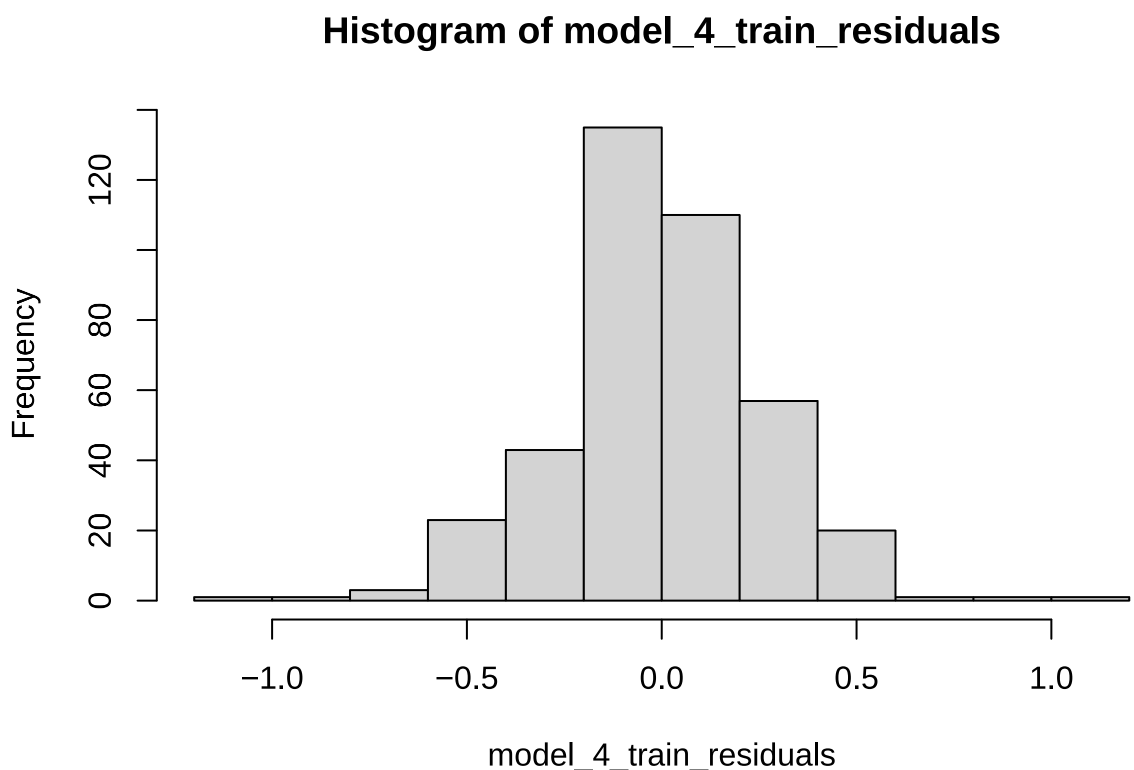
```
#SARIMA(4,1,0)x(0,1,1)_12
model_4_train <- sarima(Avg_ExtentTS_Train, p=4, d=1, q=0, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -1.045273
## iter   2 value -1.275376
## iter   3 value -1.289687
## iter   4 value -1.291649
## iter   5 value -1.300383
## iter   6 value -1.302661
## iter   7 value -1.303220
## iter   8 value -1.303284
## iter   9 value -1.303285
## iter   9 value -1.303285
## iter   9 value -1.303285
## final  value -1.303285
## converged
## initial  value -1.305545
## iter   2 value -1.305718
## iter   3 value -1.305759
## iter   4 value -1.305764
## iter   5 value -1.305764
## iter   5 value -1.305764
```

```
## iter   5 value -1.305764
## final  value -1.305764
## converged
```




```
model_4_train_residuals = resid(model_4_train$fit)
hist(model_4_train_residuals)
```



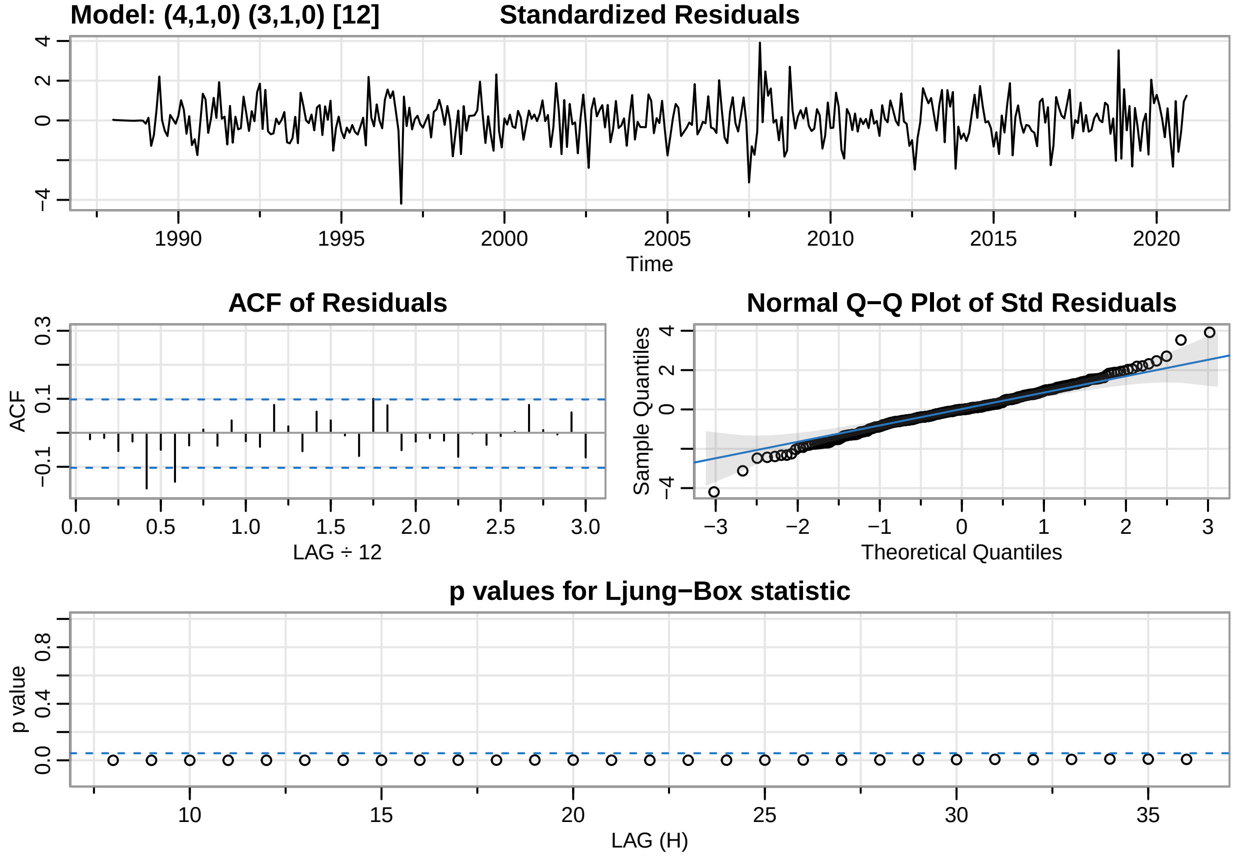

```
shapiro.test(model_4_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_4_train_residuals
## W = 0.98293, p-value = 0.0001267
```

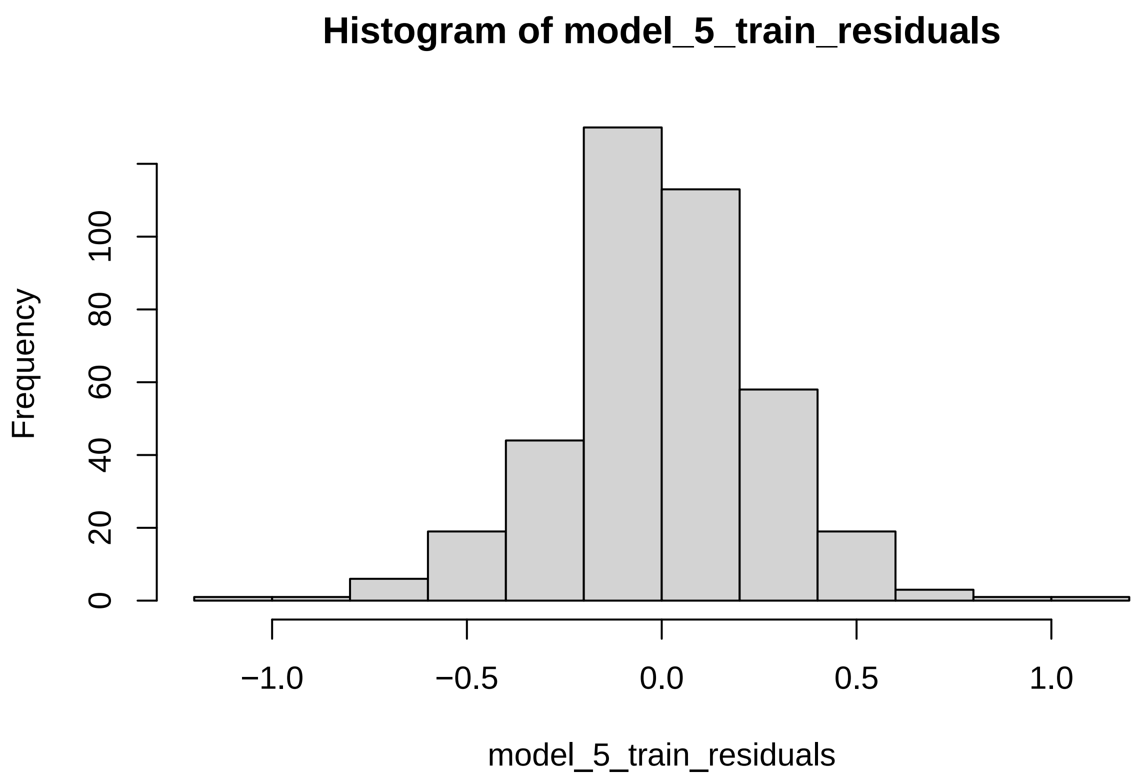
```
#SARIMA(4,1,0)x(3,1,0)_12
model_5_train <- sarima(Avg_ExtentTS_Train, p=4, d=1, q=0, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.034768
## iter   2 value -1.176278
## iter   3 value -1.267070
## iter   4 value -1.292224
## iter   5 value -1.298051
## iter   6 value -1.298311
## iter   7 value -1.298325
## iter   8 value -1.298325
## iter   8 value -1.298325
## iter   8 value -1.298325
## final  value -1.298325
## converged
## initial  value -1.301857
## iter   2 value -1.302141
## iter   3 value -1.302152
## iter   4 value -1.302153
## iter   5 value -1.302153
## iter   5 value -1.302153
## iter   5 value -1.302153
```

```
## final  value -1.302153
## converged
```




```
model_5_train_residuals = resid(model_5_train$fit)
hist(model_5_train_residuals)
```



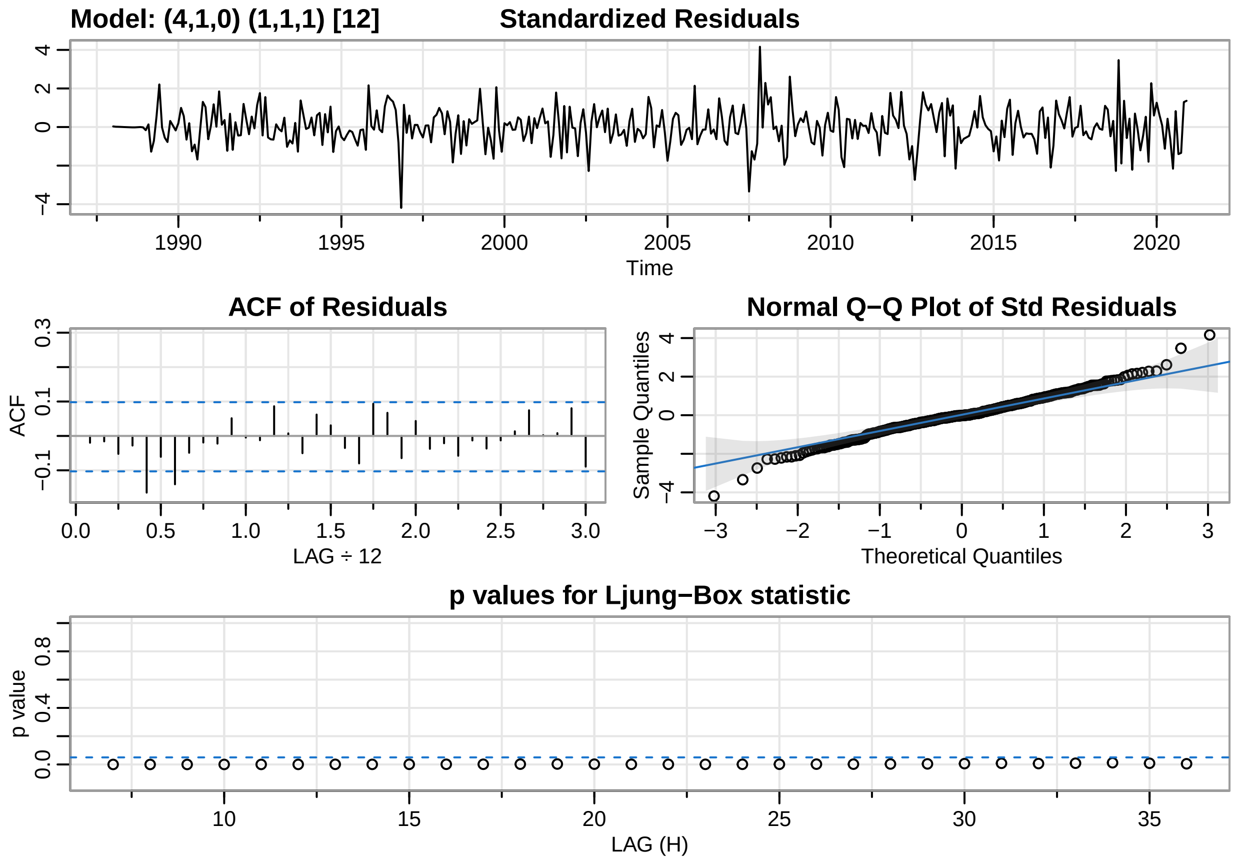

```
shapiro.test(model_5_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_5_train_residuals
## W = 0.98537, p-value = 0.0004979
```

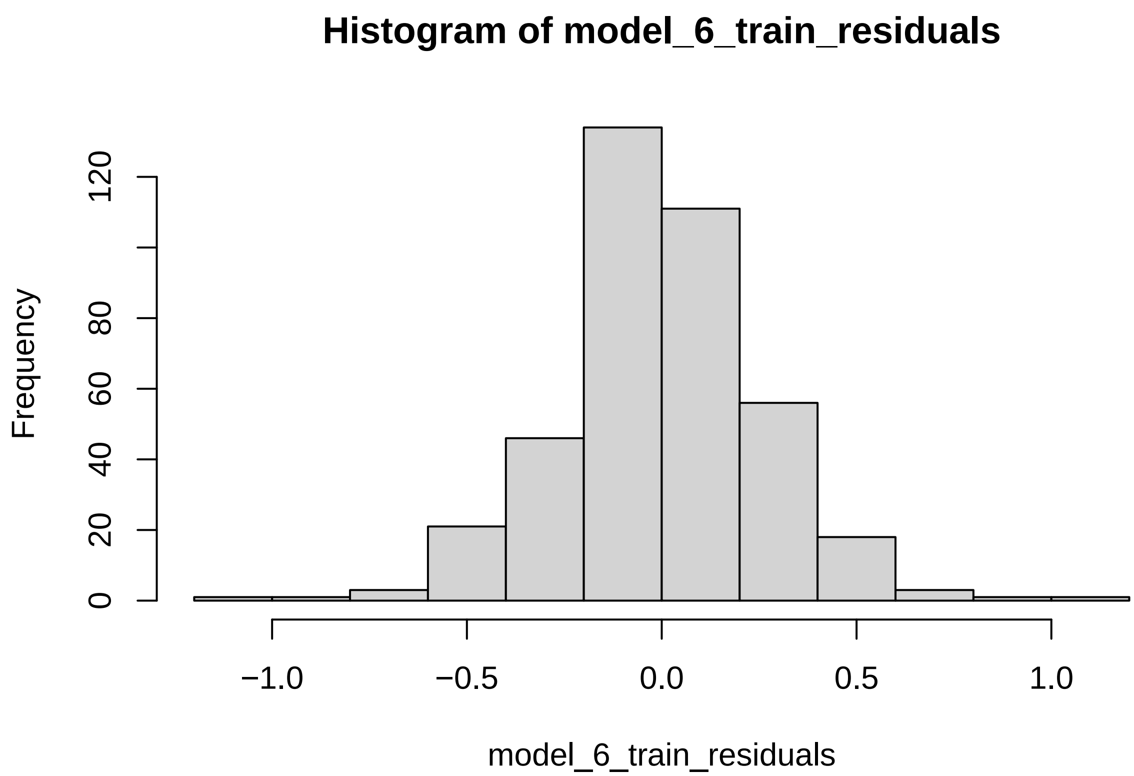
```
#SARIMA(4,1,0)x(1,1,1)_12
model_6_train <- sarima(Avg_ExtentTS_Train, p=4, d=1, q=0, P=1, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -1.041584
## iter   2 value -1.241588
## iter   3 value -1.287381
## iter   4 value -1.292821
## iter   5 value -1.300492
## iter   6 value -1.300777
## iter   7 value -1.300797
## iter   8 value -1.300801
## iter   9 value -1.300802
## iter   9 value -1.300802
## iter   9 value -1.300802
## final  value -1.300802
## converged
## initial  value -1.305672
## iter   2 value -1.306365
## iter   3 value -1.306596
## iter   4 value -1.306681
## iter   5 value -1.306698
## iter   6 value -1.306701
```

```
## iter   7 value -1.306701
## iter   7 value -1.306701
## iter   7 value -1.306701
## final  value -1.306701
## converged
```




```
model_6_train_residuals = resid(model_6_train$fit)
hist(model_6_train_residuals)
```



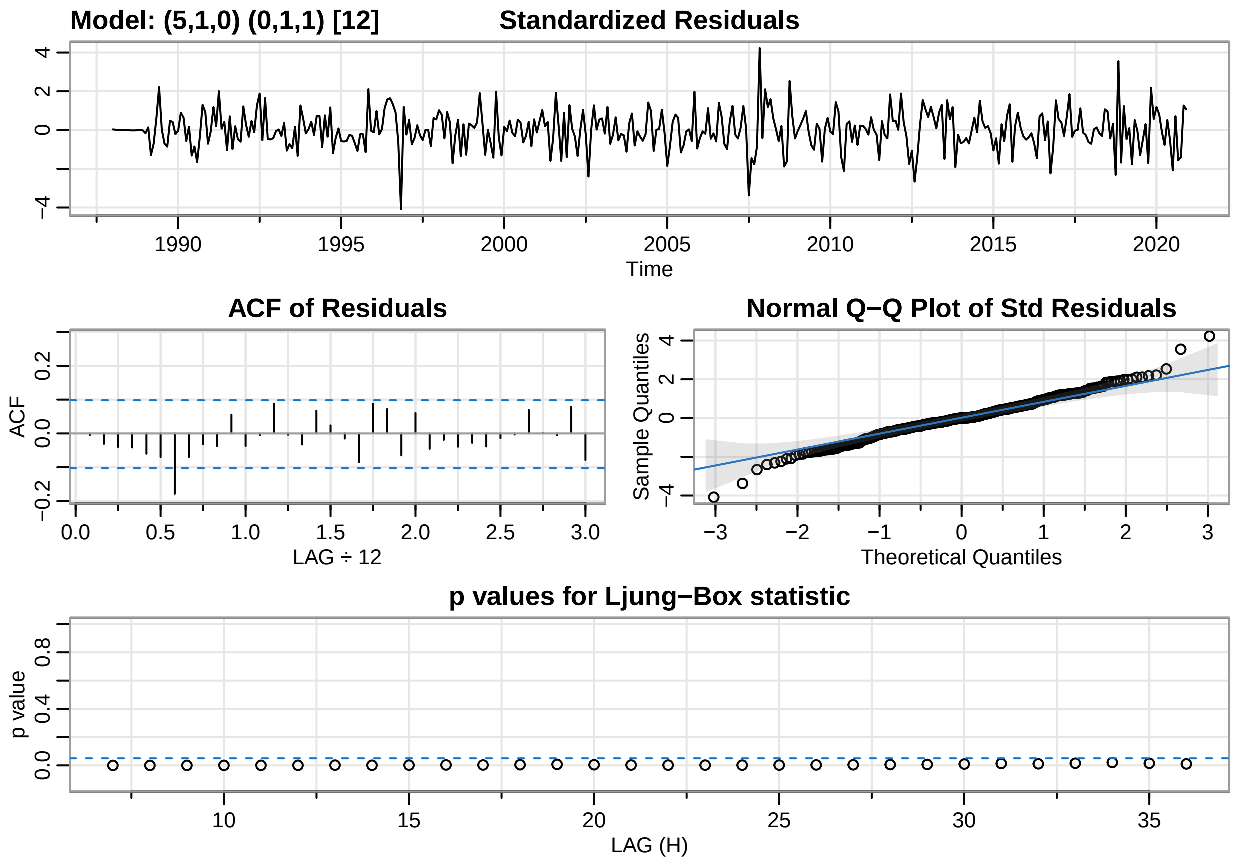

```
shapiro.test(model_6_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_6_train_residuals
## W = 0.98408, p-value = 0.0002389
```

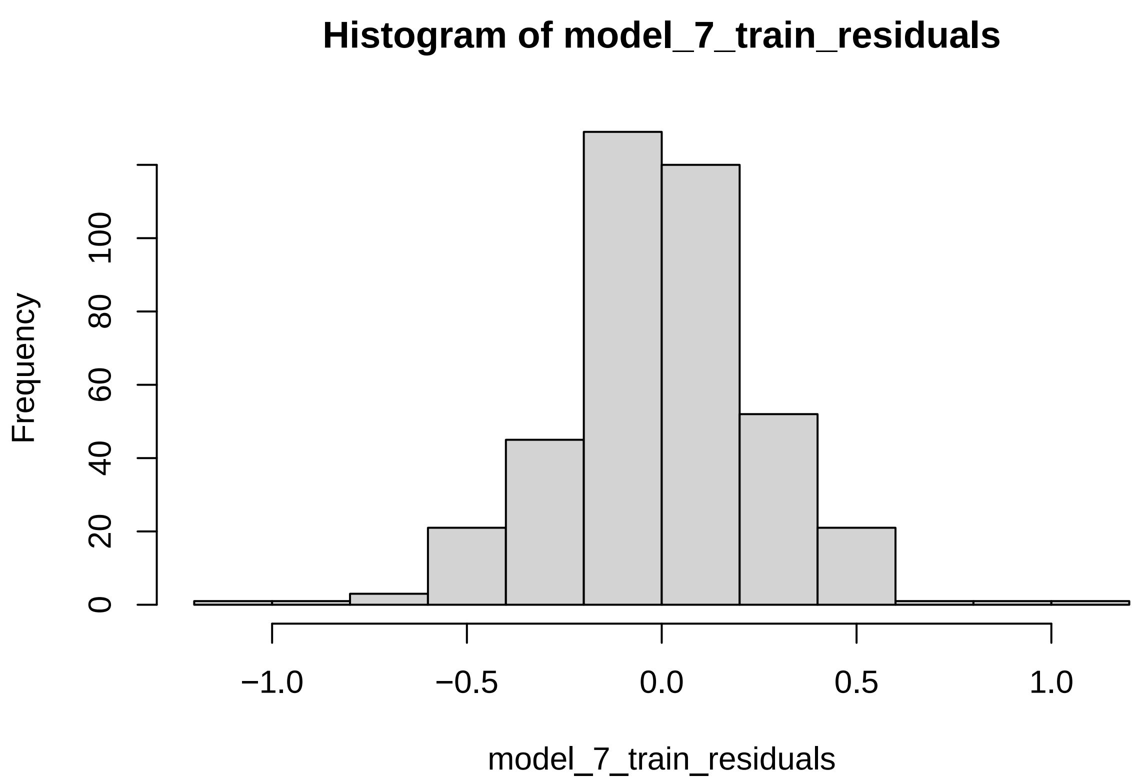
```
#SARIMA(5,1,0)x(0,1,1)_12
model_7_train <- sarima(Avg_ExtentTS_Train, p=5, d=1, q=0, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -1.051141
## iter   2 value -1.280688
## iter   3 value -1.294984
## iter   4 value -1.299314
## iter   5 value -1.306323
## iter   6 value -1.308043
## iter   7 value -1.308352
## iter   8 value -1.308378
## iter   9 value -1.308378
## iter   9 value -1.308378
## iter   9 value -1.308378
## final  value -1.308378
## converged
## initial  value -1.312741
## iter   2 value -1.313183
## iter   3 value -1.313486
## iter   4 value -1.313497
## iter   5 value -1.313500
## iter   6 value -1.313500
```

```
## iter   6 value -1.313500
## iter   6 value -1.313500
## final  value -1.313500
## converged
```




```r
model_7_train_residuals = resid(model_7_train$fit)
hist(model_7_train_residuals)
```



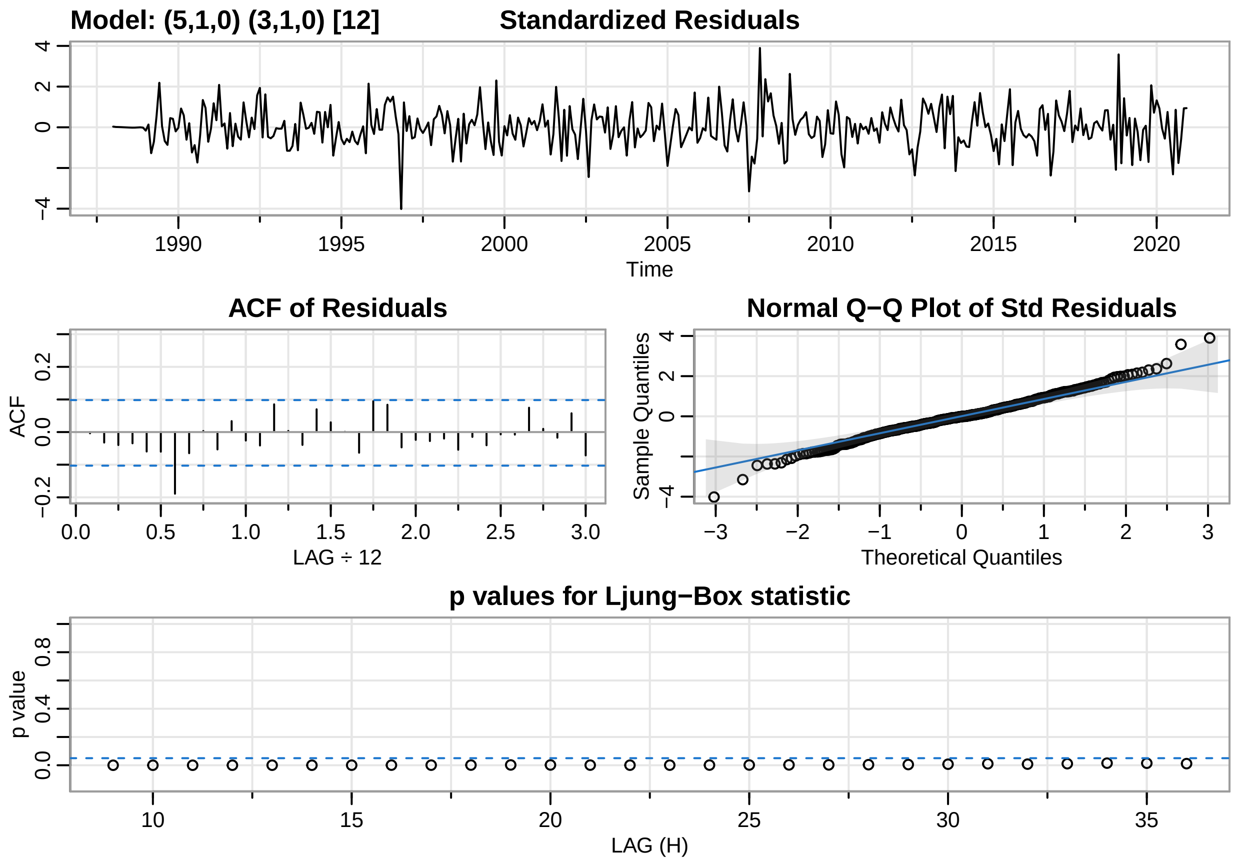
```
shapiro.test(model_7_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_7_train_residuals
## W = 0.98416, p-value = 0.0002503
```

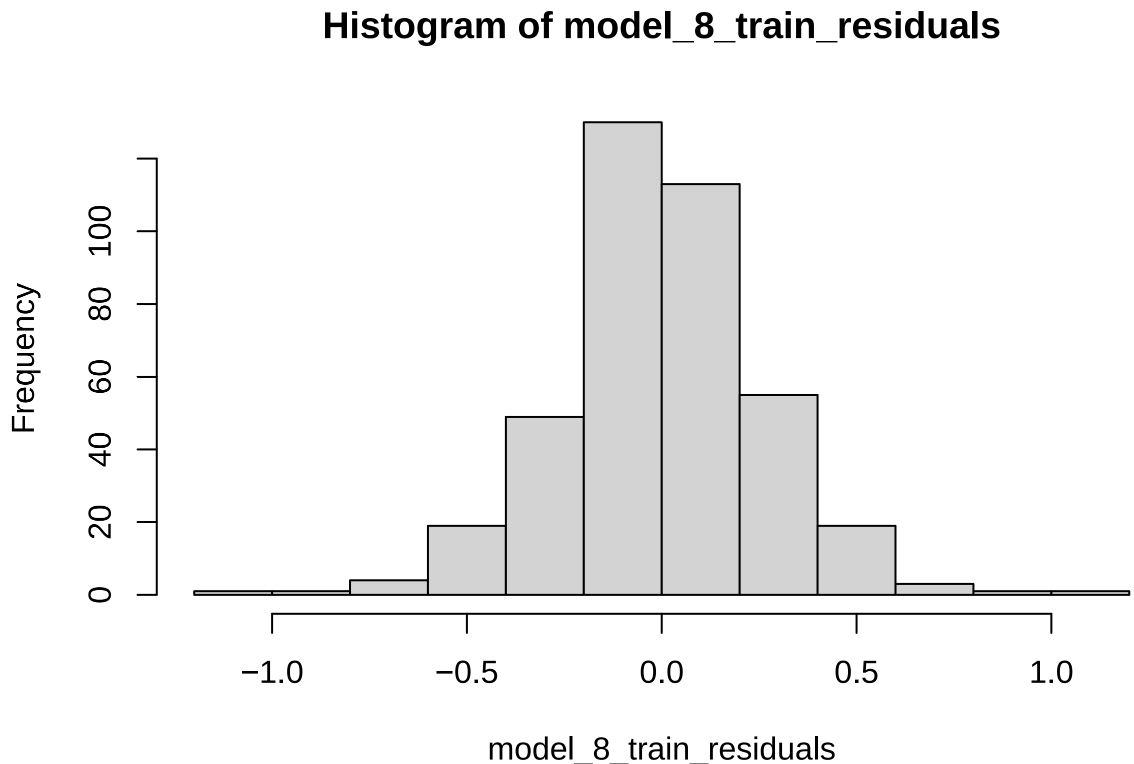
```
#SARIMA(5,1,0)x(3,1,0)_12
model_8_train <- sarima(Avg_ExtentTS_Train, p=5, d=1, q=0, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.034323
## iter   2 value -1.181171
## iter   3 value -1.276196
## iter   4 value -1.303175
## iter   5 value -1.309458
## iter   6 value -1.309745
## iter   7 value -1.309762
## iter   8 value -1.309762
## iter   8 value -1.309762
## iter   8 value -1.309762
## final  value -1.309762
## converged
## initial  value -1.309715
## iter   2 value -1.310249
## iter   3 value -1.310286
## iter   4 value -1.310300
## iter   5 value -1.310300
## iter   6 value -1.310300
## iter   6 value -1.310300
```

```
## iter   6 value -1.310300
## final  value -1.310300
## converged
```




```
model_8_train_residuals = resid(model_8_train$fit)
hist(model_8_train_residuals)
```



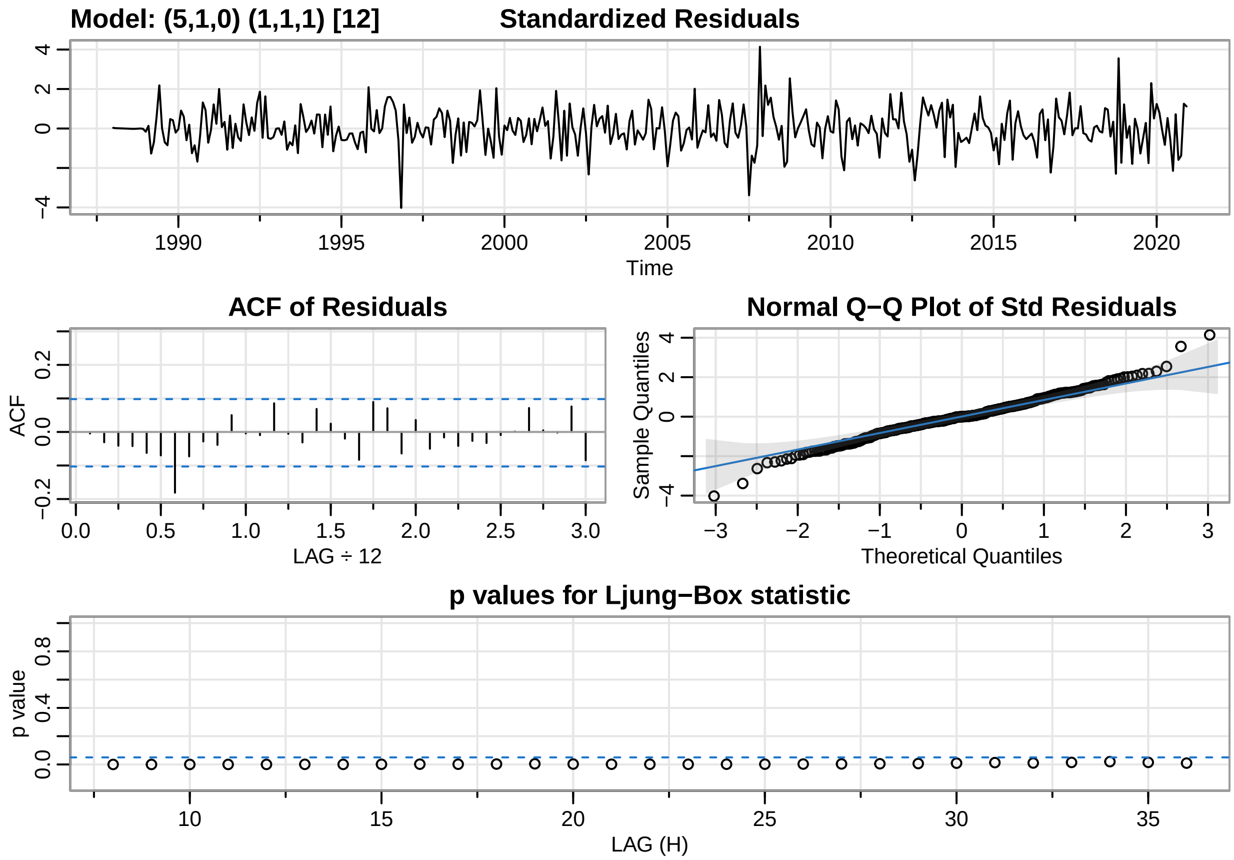

```r
shapiro.test(model_8_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_8_train_residuals
## W = 0.98765, p-value = 0.00193
```

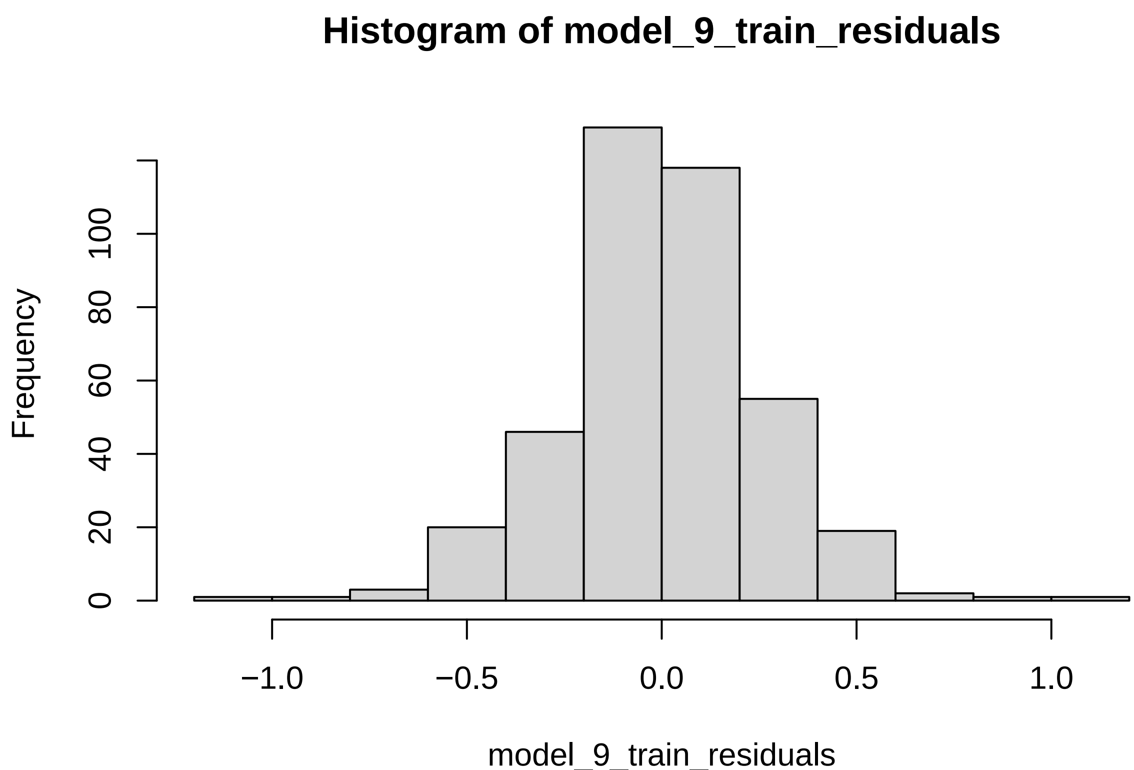
```r
#SARIMA(5,1,0)x(1,1,1)_12
model_9_train <- sarima(Avg_ExtentTS_Train, p=5, d=1, q=0, P=1, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -1.047019
## iter   2 value -1.250977
## iter   3 value -1.298091
## iter   4 value -1.305471
## iter   5 value -1.315008
## iter   6 value -1.315491
## iter   7 value -1.315527
## iter   8 value -1.315540
## iter   9 value -1.315541
## iter   9 value -1.315541
## iter   9 value -1.315541
## final  value -1.315541
## converged
## initial  value -1.313961
## iter   2 value -1.314344
## iter   3 value -1.314392
## iter   4 value -1.314401
## iter   5 value -1.314406
## iter   6 value -1.314406
```

```
## iter   7 value -1.314406
## iter   7 value -1.314406
## iter   7 value -1.314406
## final  value -1.314406
## converged
```




```
model_9_train_residuals = resid(model_9_train$fit)
hist(model_9_train_residuals)
```



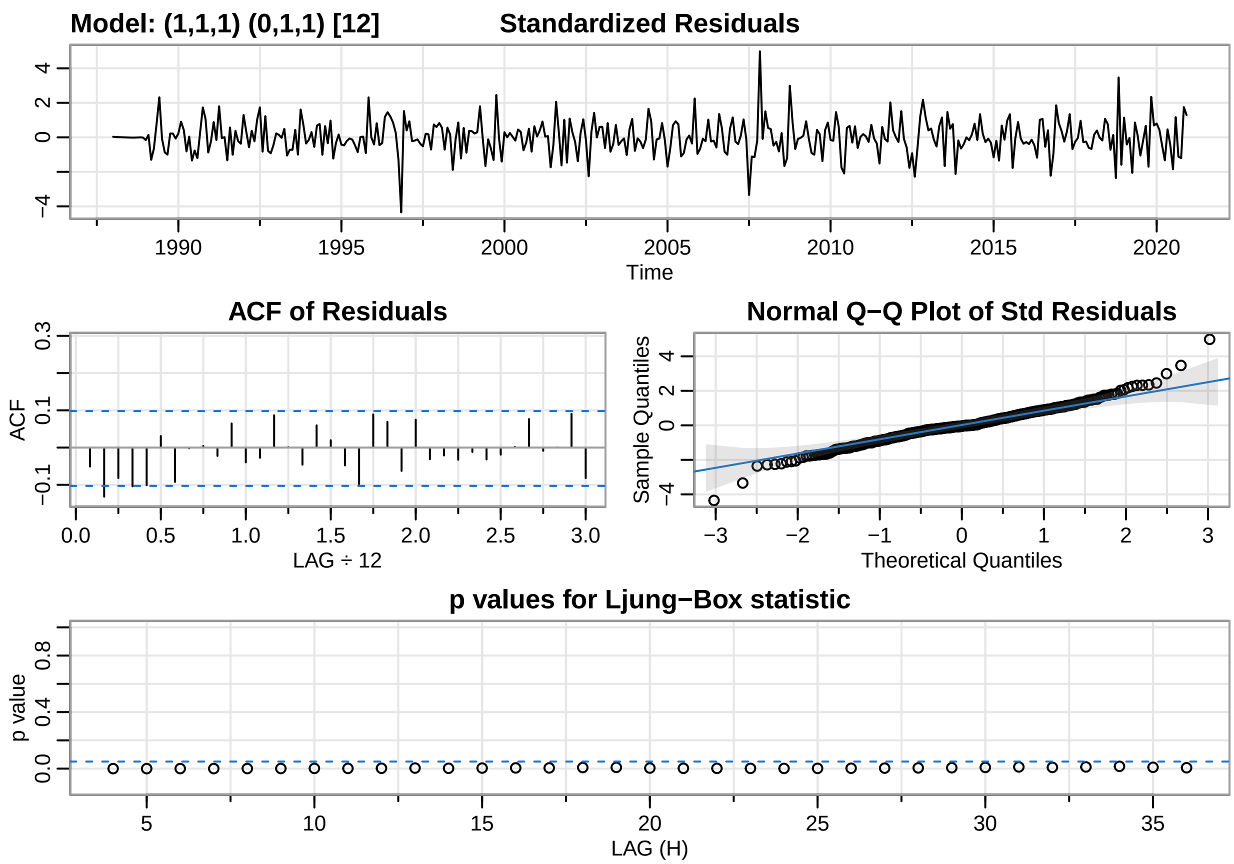

```
shapiro.test(model_9_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_9_train_residuals
## W = 0.98544, p-value = 0.0005193
```

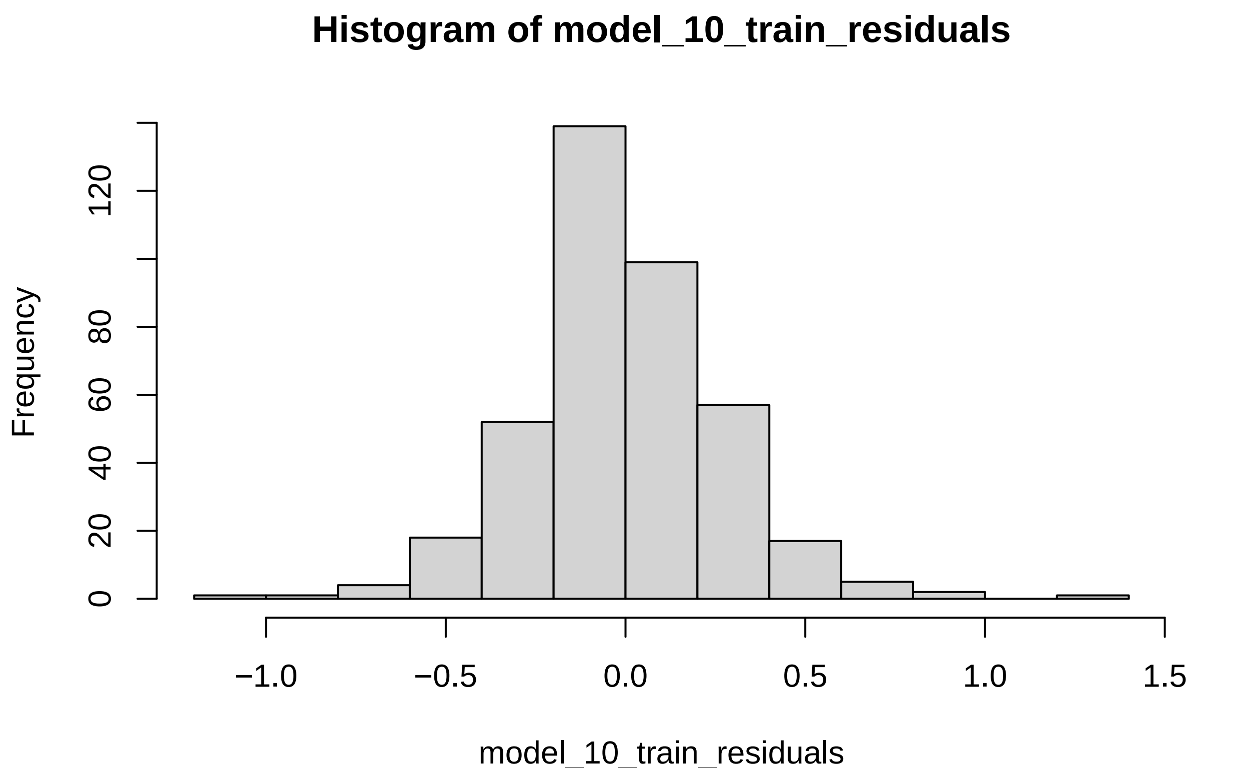
```
#SARIMA(1,1,1)x(0,1,1)_12
model_10_train <- sarima(Avg_ExtentTS_Train, p=1, d=1, q=1, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -1.045552
## iter   2 value -1.242919
## iter   3 value -1.258615
## iter   4 value -1.265606
## iter   5 value -1.269085
## iter   6 value -1.271594
## iter   7 value -1.272018
## iter   8 value -1.272061
## iter   9 value -1.272075
## iter  10 value -1.272103
## iter  11 value -1.273789
## iter  12 value -1.274147
## iter  13 value -1.274999
## iter  14 value -1.275080
## iter  15 value -1.276771
## iter  16 value -1.276973
## iter  17 value -1.277440
## iter  18 value -1.277524
## iter  19 value -1.277528
```

```
## iter  20 value -1.277529
## iter  21 value -1.277529
## iter  22 value -1.277530
## iter  23 value -1.277530
## iter  24 value -1.277530
## iter  25 value -1.277531
## iter  26 value -1.277531
## iter  26 value -1.277531
## iter  26 value -1.277531
## final  value -1.277531
## converged
## initial  value -1.276765
## iter   2 value -1.276787
## iter   3 value -1.276795
## iter   4 value -1.276795
## iter   5 value -1.276795
## iter   6 value -1.276796
## iter   7 value -1.276797
## iter   8 value -1.276797
## iter   9 value -1.276798
## iter  10 value -1.276798
## iter  10 value -1.276798
## iter  10 value -1.276798
## final  value -1.276798
## converged
```



```r
model_10_train_residuals = resid(model_10_train$fit)
hist(model_10_train_residuals)
```



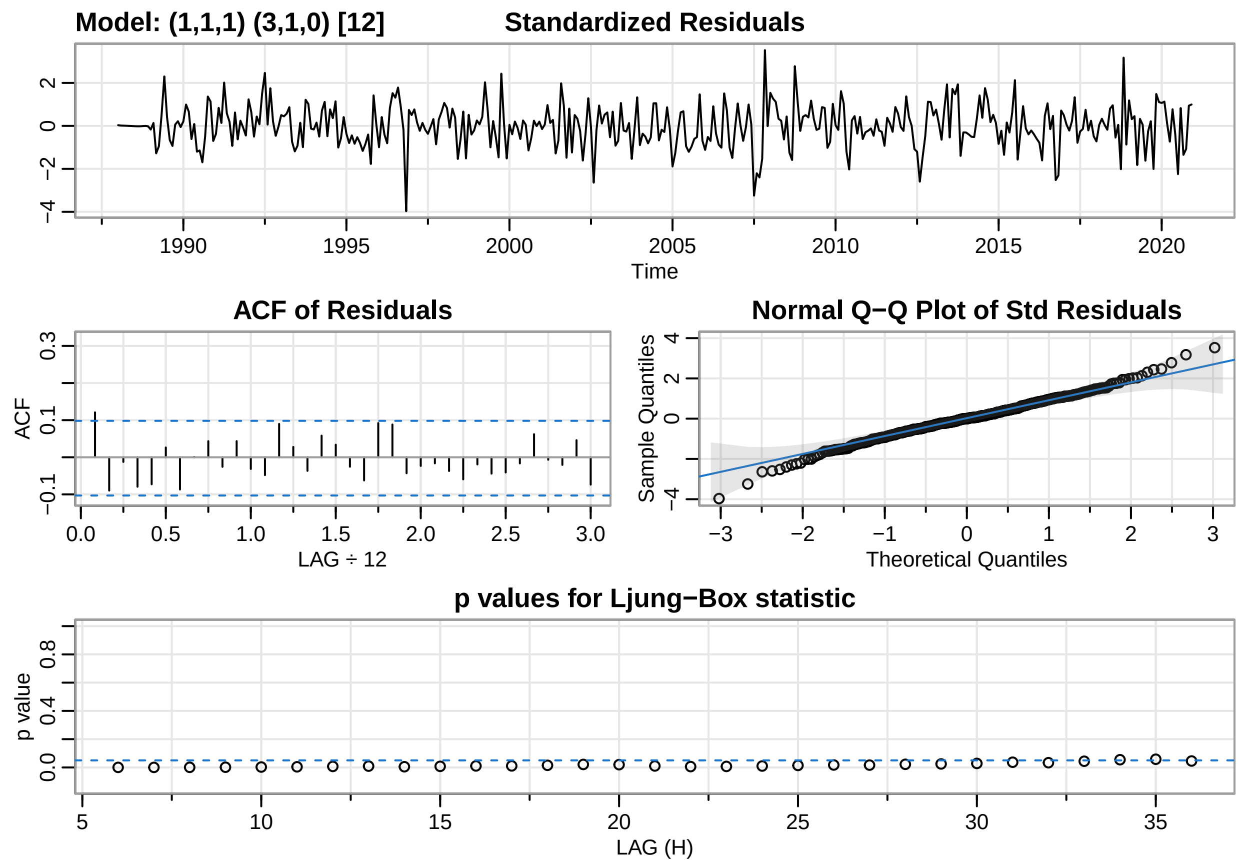

```r
shapiro.test(model_10_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_10_train_residuals
## W = 0.97359, p-value = 1.319e-06
```

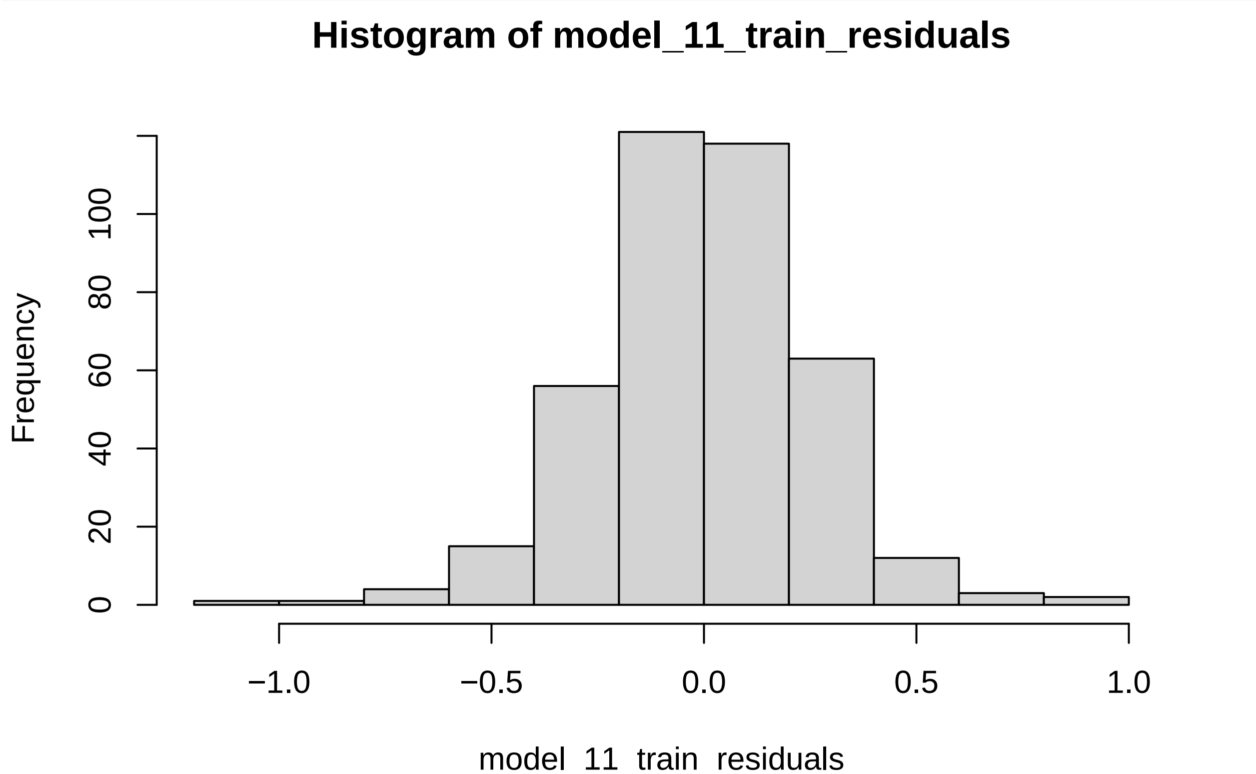
```r
#SARIMA(1,1,1)x(3,1,0)_12
model_11_train <- sarima(Avg_ExtentTS_Train, p=1, d=1, q=1, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.038097
## iter   2 value -1.144641
## iter   3 value -1.239216
## iter   4 value -1.265137
## iter   5 value -1.269506
## iter   6 value -1.269577
## iter   7 value -1.269587
## iter   8 value -1.269588
## iter   9 value -1.269590
## iter  10 value -1.269601
## iter  11 value -1.269726
## iter  12 value -1.269928
## iter  13 value -1.269990
## iter  14 value -1.270074
## iter  15 value -1.270123
## iter  16 value -1.270398
```

```
## iter  17 value -1.274020
## iter  18 value -1.276115
## iter  19 value -1.281384
## iter  20 value -1.292576
## iter  21 value -1.304600
## iter  22 value -1.318963
## iter  23 value -1.331206
## iter  24 value -1.334219
## iter  25 value -1.334457
## iter  26 value -1.334997
## iter  27 value -1.336065
## iter  28 value -1.336081
## iter  29 value -1.336093
## iter  30 value -1.336095
## iter  30 value -1.336095
## iter  30 value -1.336095
## final  value -1.336095
## converged
## initial  value -1.332968
## iter   2 value -1.334981
## iter   3 value -1.336741
## iter   4 value -1.336979
## iter   5 value -1.337236
## iter   6 value -1.337267
## iter   7 value -1.337272
## iter   8 value -1.337273
## iter   9 value -1.337273
## iter   9 value -1.337273
## iter   9 value -1.337273
## final  value -1.337273
## converged
```




```
model_11_train_residuals = resid(model_11_train$fit)
hist(model_11_train_residuals)
```



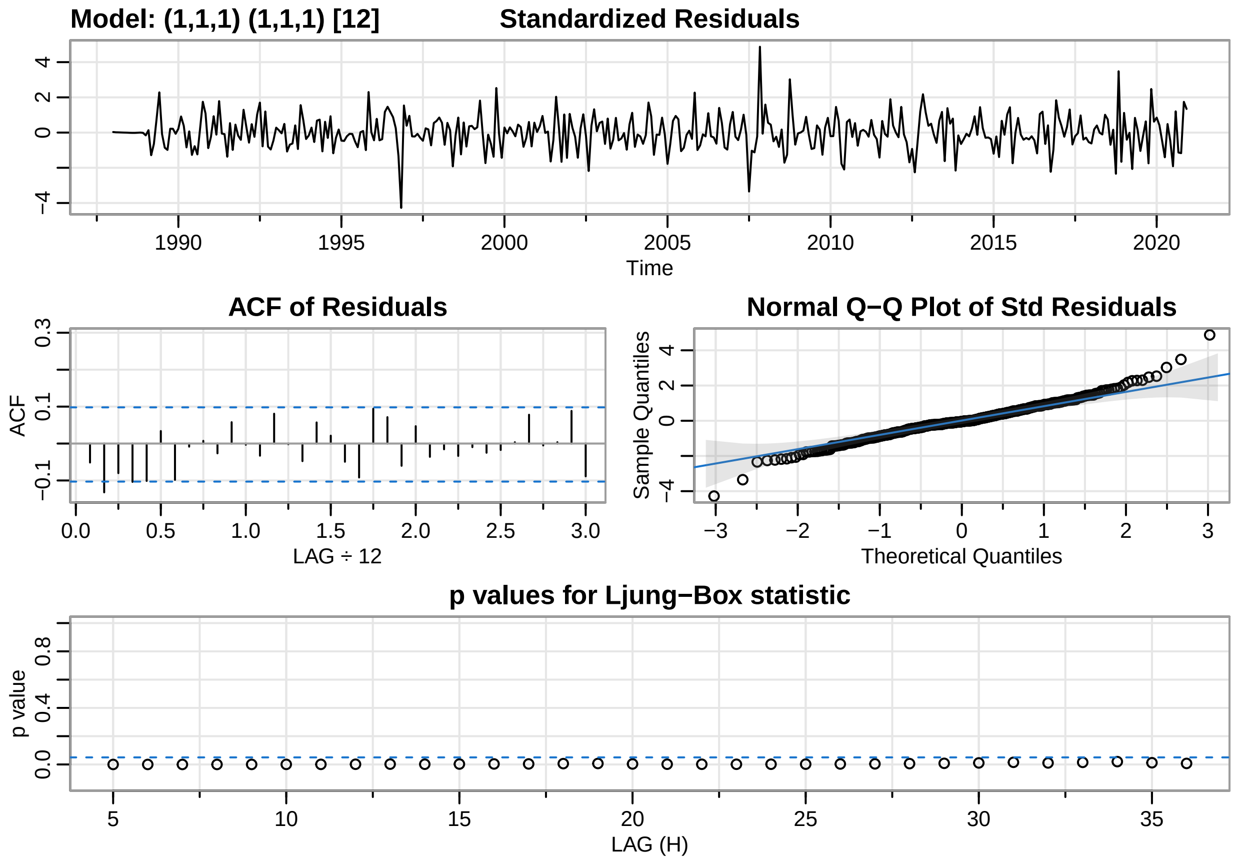
```r
shapiro.test(model_11_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_11_train_residuals
## W = 0.99044, p-value = 0.01131
```

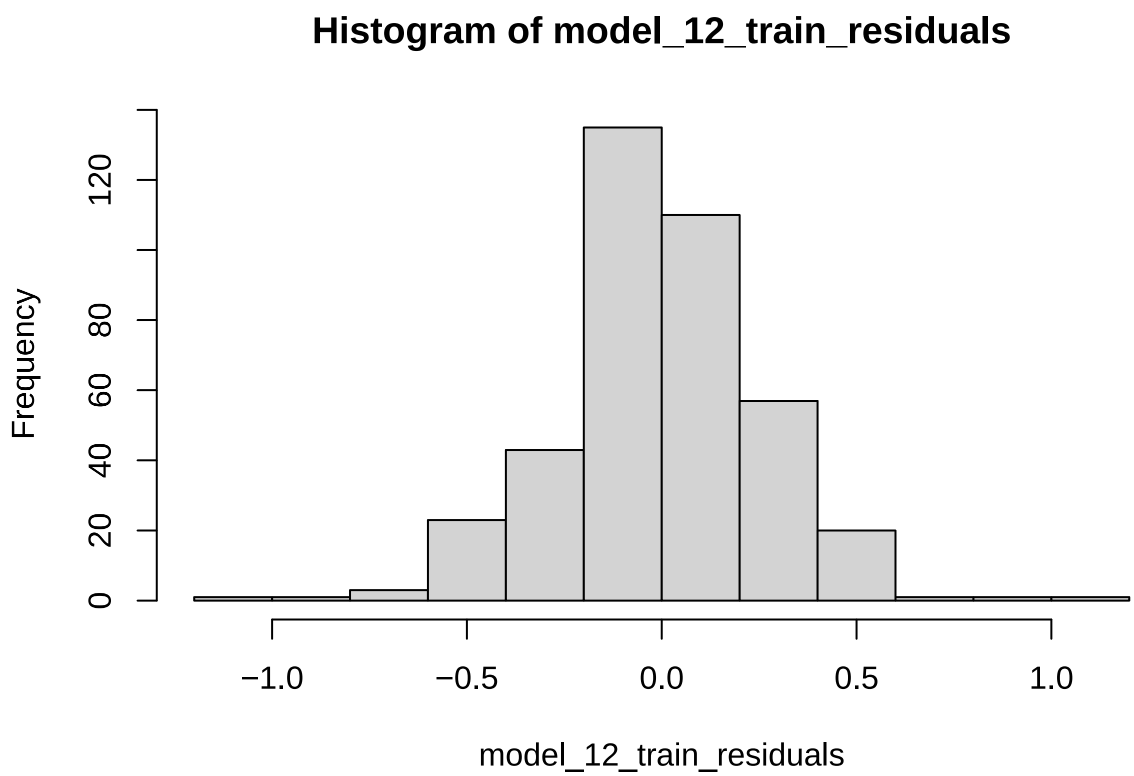
```r
#SARIMA(1,1,1)x(1,1,1)_12
model_12_train <- sarima(Avg_ExtentTS_Train, p=1, d=1, q=1, P=1, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -1.043857
## iter   2 value -1.216764
## iter   3 value -1.259588
## iter   4 value -1.264738
## iter   5 value -1.271016
## iter   6 value -1.271155
## iter   7 value -1.271164
## iter   8 value -1.271165
## iter   9 value -1.271166
## iter  10 value -1.271174
## iter  11 value -1.271177
## iter  12 value -1.271179
## iter  13 value -1.271182
## iter  14 value -1.271192
## iter  15 value -1.271226
## iter  16 value -1.271350
## iter  17 value -1.271513
## iter  18 value -1.271902
## iter  19 value -1.272301
## iter  20 value -1.272695
## iter  21 value -1.272800
## iter  22 value -1.272856
## iter  23 value -1.272922
## iter  24 value -1.273079
## iter  25 value -1.273307
## iter  26 value -1.273488
## iter  27 value -1.273600
## iter  28 value -1.273602
## iter  29 value -1.273604
## iter  30 value -1.273604
## iter  31 value -1.273604
## iter  31 value -1.273604
## iter  31 value -1.273604
## final  value -1.273604
## converged
## initial  value -1.276218
## iter   2 value -1.276807
## iter   3 value -1.277527
## iter   4 value -1.277549
## iter   5 value -1.277577
## iter   6 value -1.277709
## iter   7 value -1.277802
## iter   8 value -1.277869
```

```
## iter   9 value -1.277873
## iter  10 value -1.277874
## iter  10 value -1.277874
## final  value -1.277874
## converged
```




```
model_12_train_residuals = resid(model_4_train$fit)
hist(model_12_train_residuals)
```



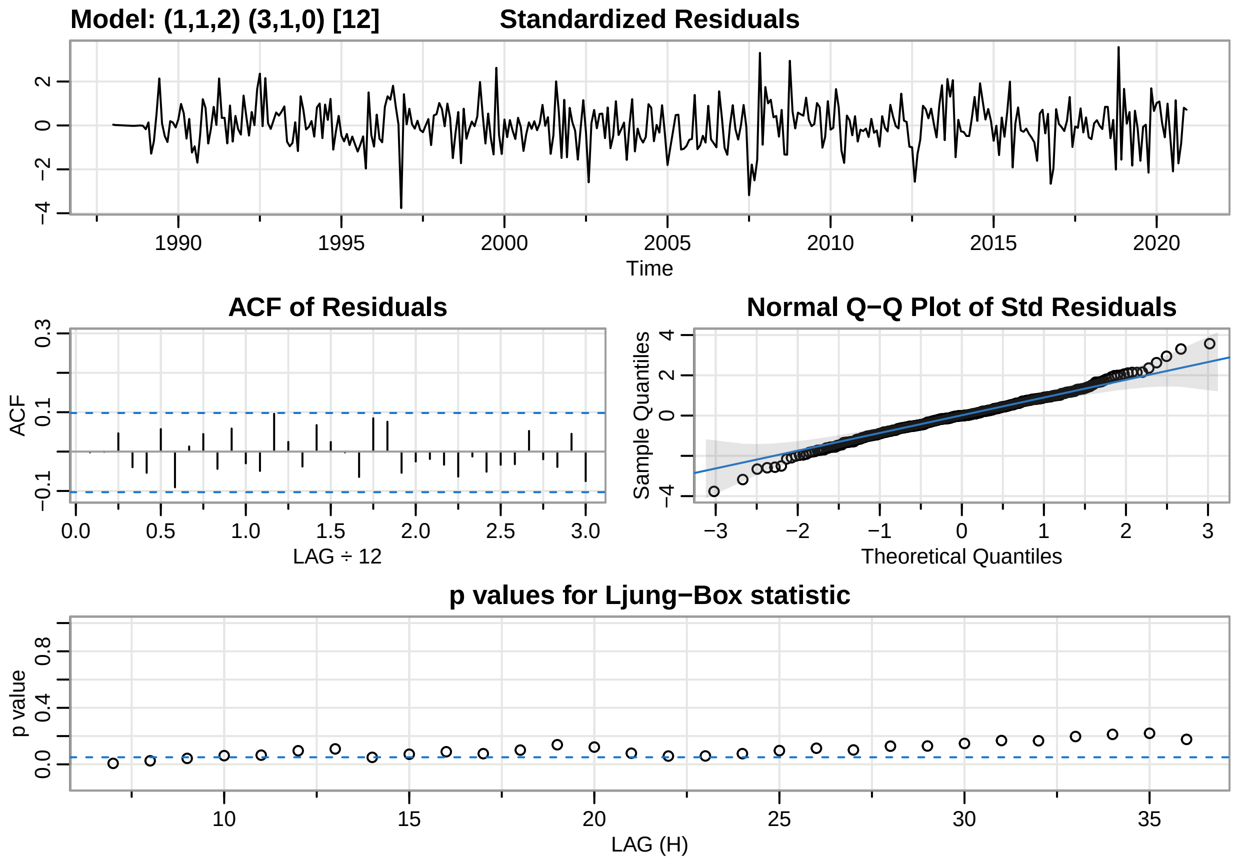

```
shapiro.test(model_12_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_12_train_residuals
## W = 0.98293, p-value = 0.0001267
```

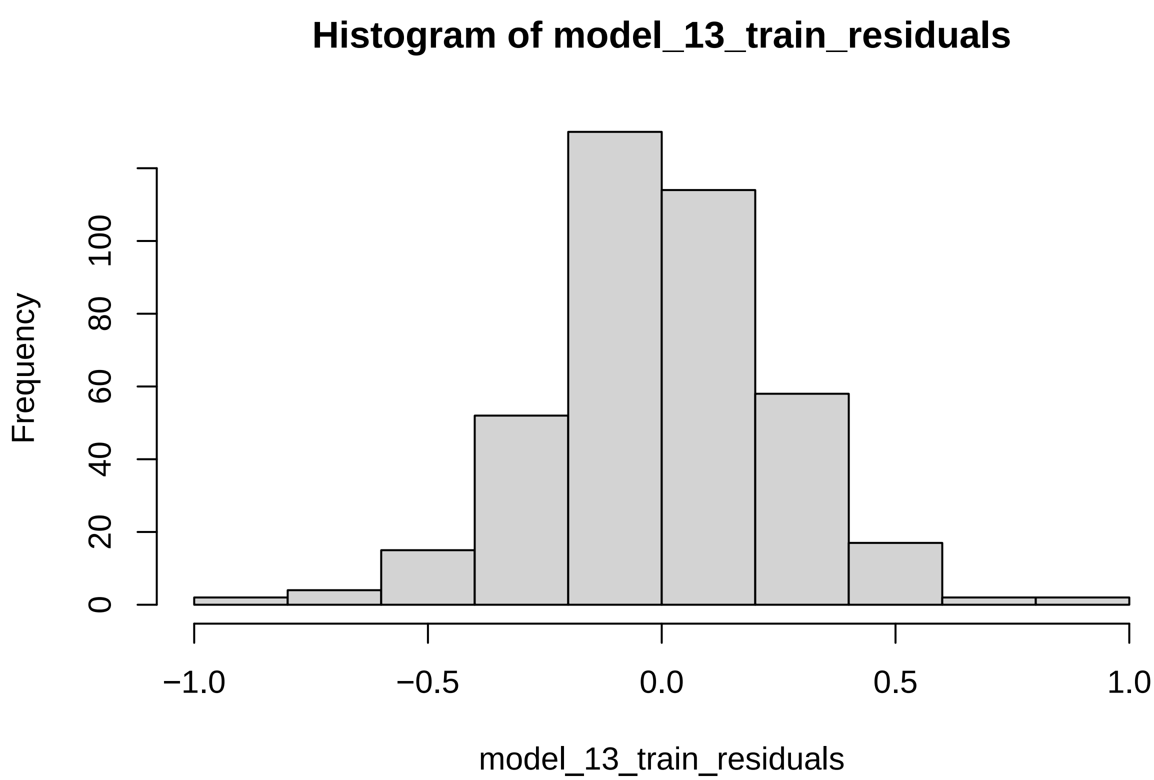
```
#SARIMA(1,1,2)x(3,1,0)_12
model_13_train <- sarima(Avg_ExtentTS_Train, p=1, d=1, q=2, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.038097
## iter   2 value -1.174147
## iter   3 value -1.267295
## iter   4 value -1.292320
## iter   5 value -1.298557
## iter   6 value -1.301947
## iter   7 value -1.306400
## iter   8 value -1.321210
## iter   9 value -1.330951
## iter  10 value -1.335397
## iter  11 value -1.337789
## iter  12 value -1.340572
## iter  13 value -1.347191
## iter  14 value -1.348937
## iter  15 value -1.350213
## iter  16 value -1.351118
## iter  17 value -1.351773
## iter  18 value -1.352098
## iter  19 value -1.352148
```

```
## iter  20 value -1.352151
## iter  21 value -1.352151
## iter  21 value -1.352151
## final  value -1.352151
## converged
## initial  value -1.349885
## iter   2 value -1.351590
## iter   3 value -1.352561
## iter   4 value -1.352788
## iter   5 value -1.352968
## iter   6 value -1.352987
## iter   7 value -1.353011
## iter   8 value -1.353019
## iter   9 value -1.353020
## iter  10 value -1.353021
## iter  10 value -1.353021
## iter  10 value -1.353021
## final  value -1.353021
## converged
```




```
model_13_train_residuals = resid(model_13_train$fit)
hist(model_13_train_residuals)
```



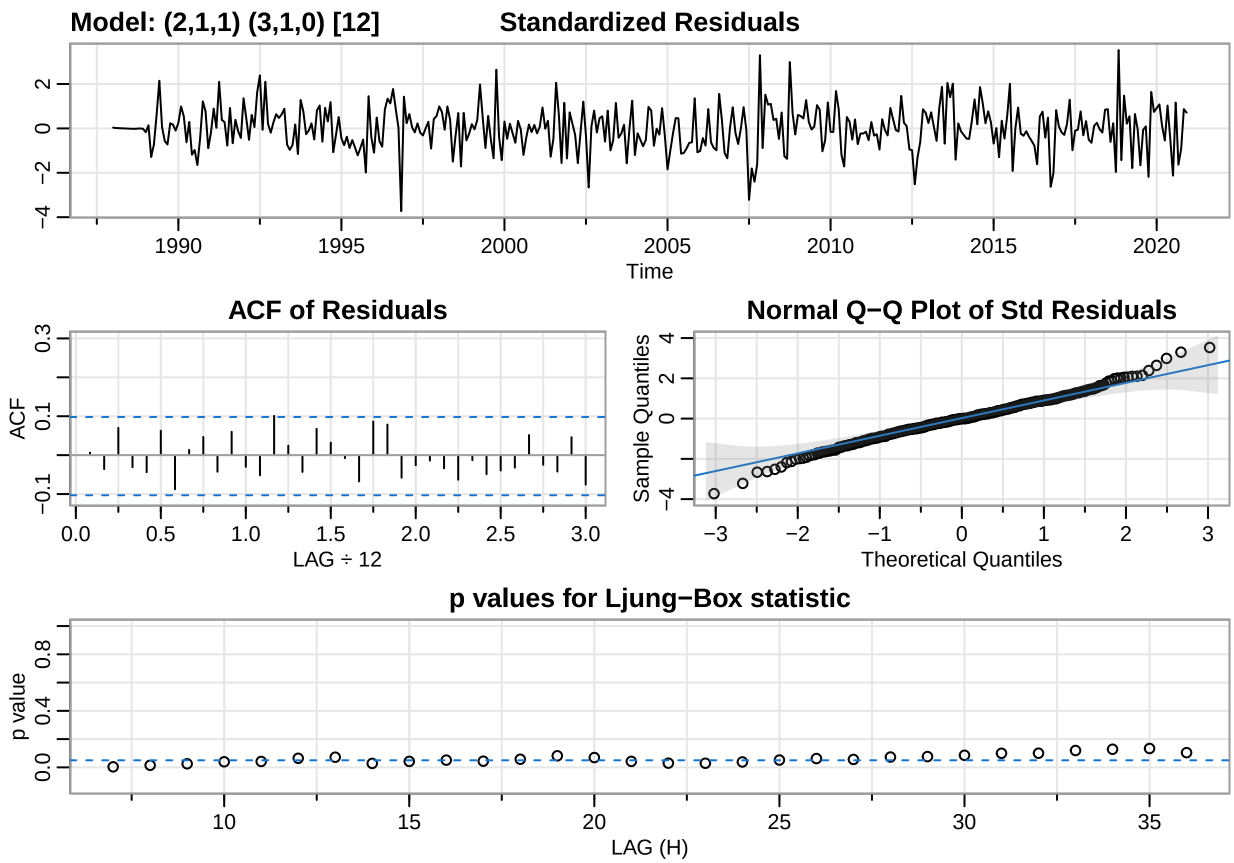

```
shapiro.test(model_13_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_13_train_residuals
## W = 0.98949, p-value = 0.006128
```

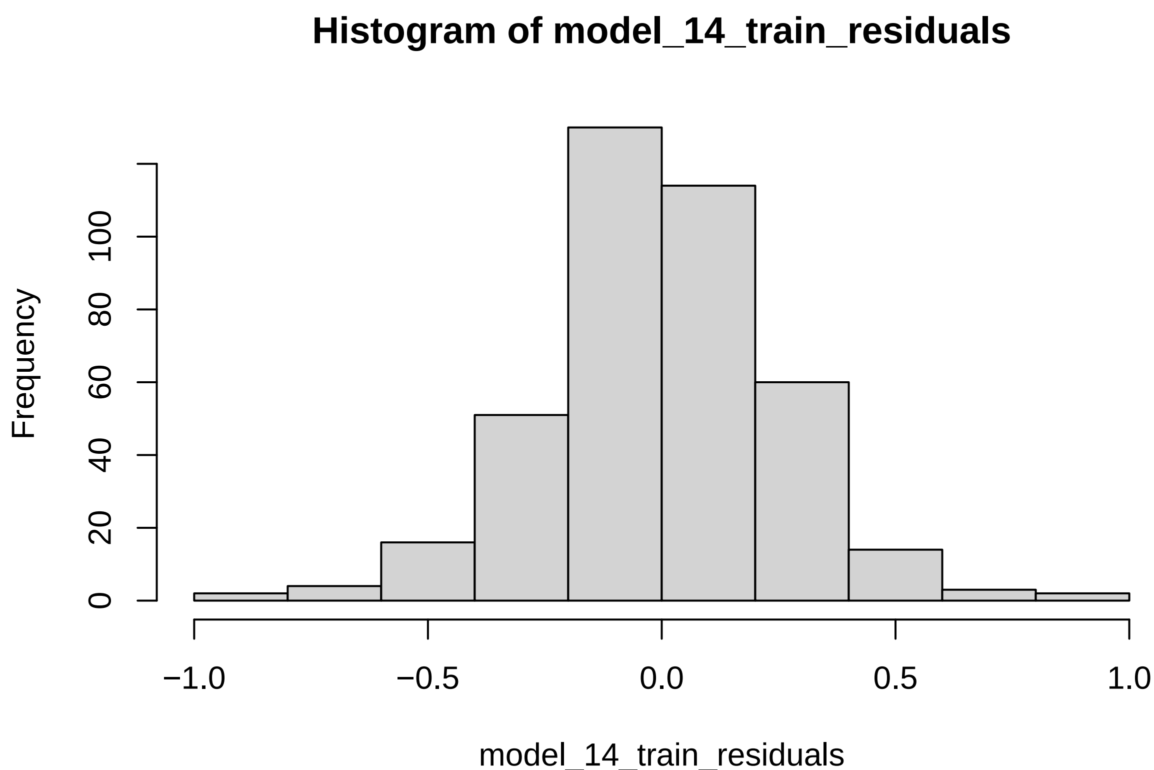
```
#SARIMA(2,1,1)x(3,1,0)_12
model_14_train <- sarima(Avg_ExtentTS_Train, p=2, d=1, q=1, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.036901
## iter   2 value -1.166690
## iter   3 value -1.255867
## iter   4 value -1.280740
## iter   5 value -1.285485
## iter   6 value -1.286803
## iter   7 value -1.286902
## iter   8 value -1.286964
## iter   9 value -1.287912
## iter  10 value -1.294229
## iter  11 value -1.298570
## iter  12 value -1.305005
## iter  13 value -1.310441
## iter  14 value -1.314638
## iter  15 value -1.315596
## iter  16 value -1.329652
## iter  17 value -1.330212
## iter  18 value -1.330363
## iter  19 value -1.330578
```

```
## iter  20 value -1.330666
## iter  21 value -1.330705
## iter  22 value -1.330707
## iter  23 value -1.330707
## iter  24 value -1.330707
## iter  24 value -1.330707
## iter  24 value -1.330707
## final  value -1.330707
## converged
## initial  value -1.343351
## iter   2 value -1.348834
## iter   3 value -1.350278
## iter   4 value -1.351392
## iter   5 value -1.351635
## iter   6 value -1.351709
## iter   7 value -1.351762
## iter   8 value -1.351781
## iter   9 value -1.351785
## iter  10 value -1.351785
## iter  10 value -1.351785
## iter  10 value -1.351785
## final  value -1.351785
## converged
```




```
model_14_train_residuals = resid(model_14_train$fit)
hist(model_14_train_residuals)
```




```
shapiro.test(model_14_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_14_train_residuals
## W = 0.99044, p-value = 0.01131
```

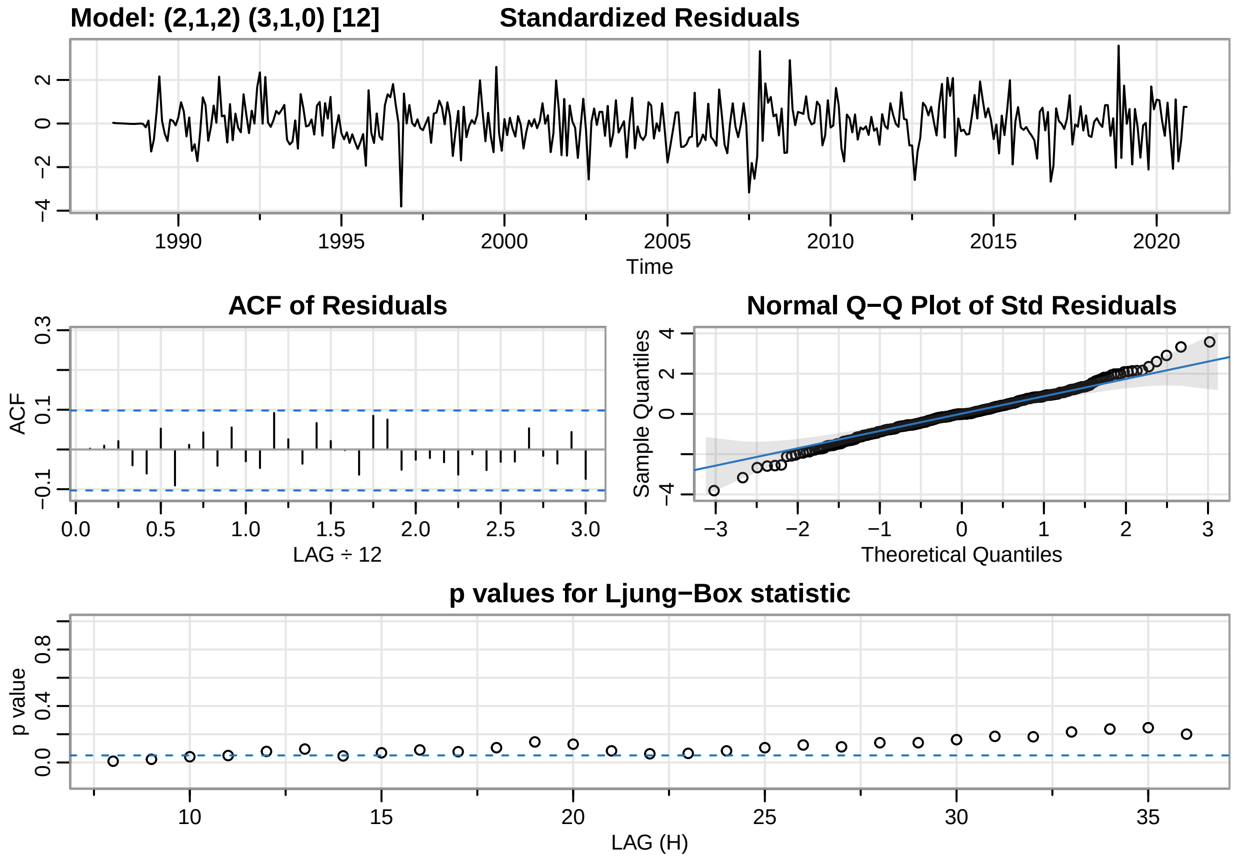
```
#SARIMA(2,1,2)x(3,1,0)_12
model_15_train <- sarima(Avg_ExtentTS_Train, p=2, d=1, q=2, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.036901
## iter   2 value -1.157349
## iter   3 value -1.257467
## iter   4 value -1.285105
## iter   5 value -1.293434
## iter   6 value -1.295560
## iter   7 value -1.300881
## iter   8 value -1.302705
## iter   9 value -1.311610
## iter  10 value -1.325367
## iter  11 value -1.333384
## iter  12 value -1.335509
## iter  13 value -1.335519
## iter  14 value -1.337033
## iter  15 value -1.338102
## iter  16 value -1.339632
## iter  17 value -1.340943
## iter  18 value -1.341232
## iter  19 value -1.343535
```

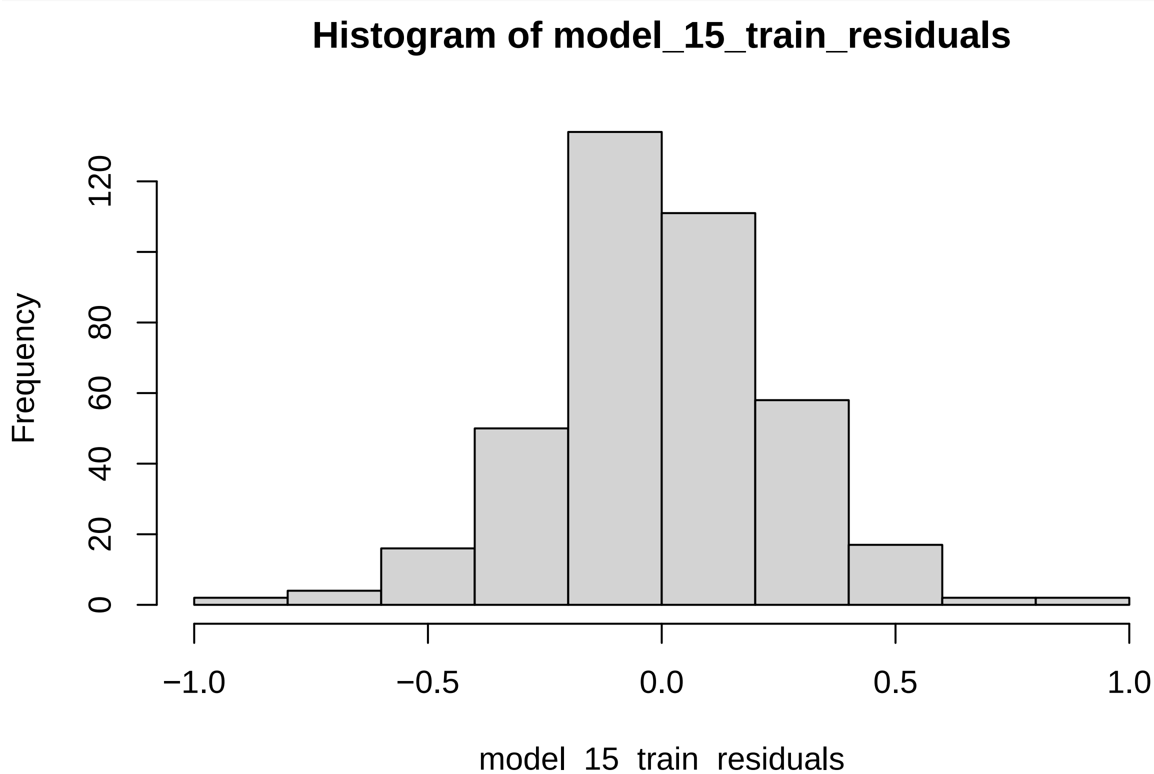
```
## iter  20 value -1.345330
## iter  21 value -1.346789
## iter  22 value -1.346934
## iter  23 value -1.347039
## iter  24 value -1.347063
## iter  25 value -1.347078
## iter  26 value -1.347086
## iter  27 value -1.347112
## iter  28 value -1.347139
## iter  29 value -1.347147
## iter  30 value -1.347149
## iter  31 value -1.347149
## iter  32 value -1.347149
## iter  32 value -1.347149
## iter  32 value -1.347149
## final  value -1.347149
## converged
## initial  value -1.348512
## iter   2 value -1.349569
## iter   3 value -1.349586
## iter   4 value -1.350848
## iter   5 value -1.350982
## iter   6 value -1.351193
## iter   7 value -1.352121
## iter   8 value -1.352601
## iter   9 value -1.353193
## iter  10 value -1.353272
## iter  11 value -1.353333
## iter  12 value -1.353339
## iter  13 value -1.353341
## iter  14 value -1.353341
## iter  15 value -1.353341
## iter  16 value -1.353342
## iter  16 value -1.353341
## iter  16 value -1.353341
## final  value -1.353342
## converged
```

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

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




```
model_15_train_residuals = resid(model_15_train$fit)
hist(model_15_train_residuals)
```

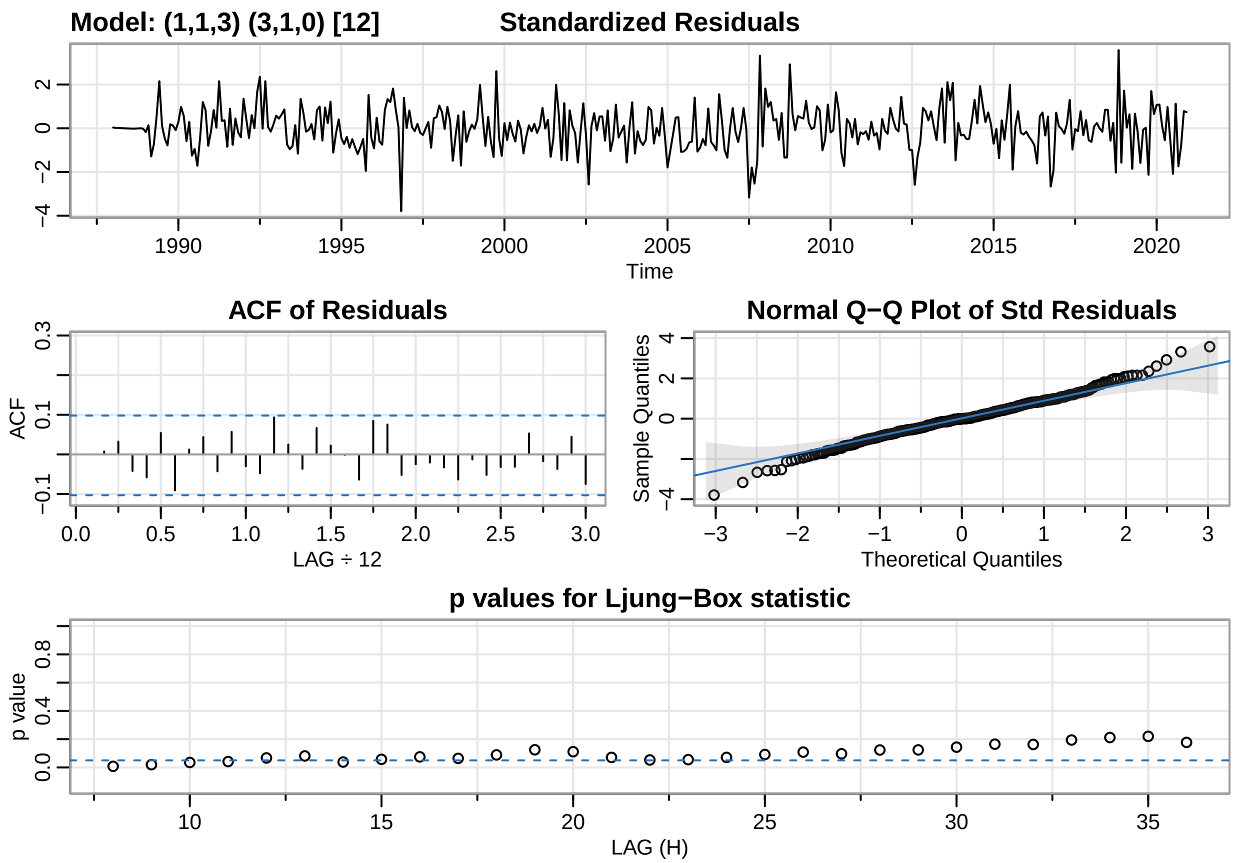


```
shapiro.test(model_15_train_residuals)
```

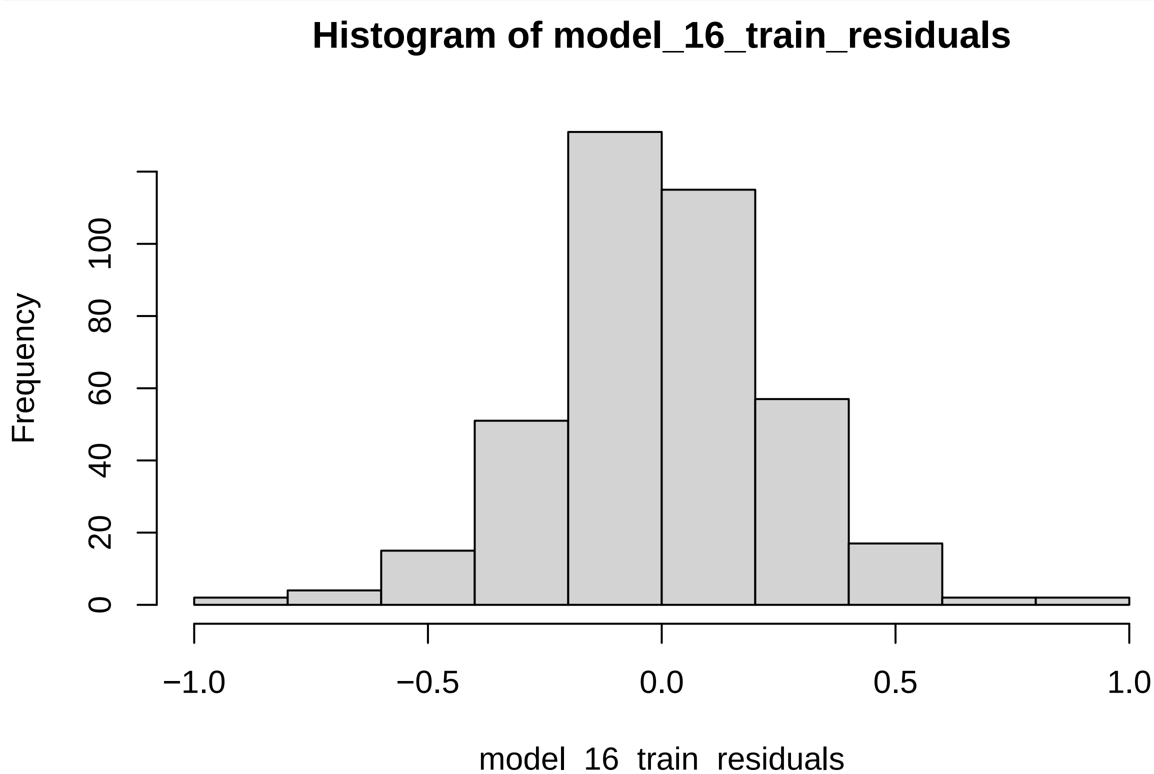
```
##
##  Shapiro-Wilk normality test
##
## data:  model_15_train_residuals
## W = 0.98876, p-value = 0.003849
```

```
#SARIMA(1,1,3)x(3,1,0)_12
model_16_train <- sarima(Avg_ExtentTS_Train, p=1, d=1, q=3, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.038097
## iter   2 value -1.179338
## iter   3 value -1.282734
## iter   4 value -1.316080
## iter   5 value -1.323583
## iter   6 value -1.325829
## iter   7 value -1.327176
## iter   8 value -1.330400
## iter   9 value -1.336051
## iter  10 value -1.343009
## iter  11 value -1.345708
## iter  12 value -1.349993
## iter  13 value -1.351352
## iter  14 value -1.351698
## iter  15 value -1.352012
## iter  16 value -1.352157
## iter  17 value -1.352161
## iter  18 value -1.352172
## iter  19 value -1.352179
## iter  20 value -1.352186
## iter  21 value -1.352188
## iter  22 value -1.352189
## iter  23 value -1.352190
## iter  24 value -1.352190
## iter  24 value -1.352190
## iter  24 value -1.352190
## final  value -1.352190
## converged
## initial  value -1.349943
## iter   2 value -1.352612
## iter   3 value -1.352925
## iter   4 value -1.353025
## iter   5 value -1.353075
## iter   6 value -1.353123
## iter   7 value -1.353167
## iter   8 value -1.353192
## iter   9 value -1.353198
## iter  10 value -1.353199
## iter  11 value -1.353199
## iter  11 value -1.353199
## iter  11 value -1.353199
## final  value -1.353199
## converged
```




```
model_16_train_residuals = resid(model_16_train$fit)
hist(model_16_train_residuals)
```



```r
shapiro.test(model_16_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_16_train_residuals
## W = 0.98903, p-value = 0.004551
```

```r
#SARIMA(3,1,1)x(3,1,0)_12
#model_17_train <- sarima(Avg_ExtentTS_Train, p=3, d=1, q=1, P=3, D=1, Q=0, S=12, details = TRUE)
#model_17_train_residuals = resid(model_17_train$fit)
#hist(model_17_train_residuals)
#shapiro.test(model_17_train_residuals)
#gives an error when run
```

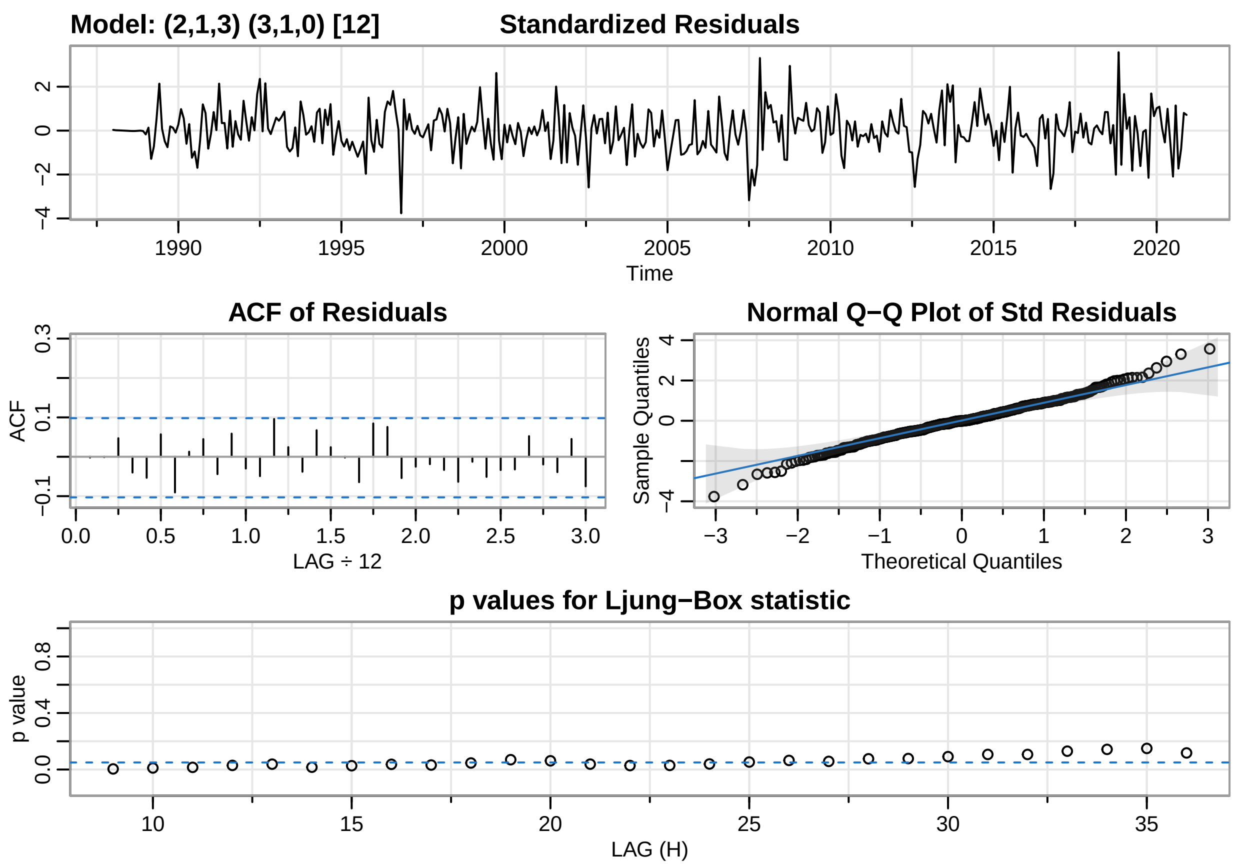
```r
#SARIMA(2,1,3)x(3,1,0)_12
model_18_train <- sarima(Avg_ExtentTS_Train, p=2, d=1, q=3, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.036901
## iter   2 value -1.163257
## iter   3 value -1.266869
## iter   4 value -1.296361
## iter   5 value -1.310632
## iter   6 value -1.321697
## iter   7 value -1.325417
## iter   8 value -1.332240
## iter   9 value -1.336089
## iter  10 value -1.340935
## iter  11 value -1.350055
## iter  12 value -1.350337
## iter  13 value -1.350570
## iter  14 value -1.350839
## iter  15 value -1.350916
## iter  16 value -1.351018
## iter  17 value -1.351217
## iter  18 value -1.351605
## iter  19 value -1.351656
## iter  20 value -1.351764
## iter  21 value -1.351810
## iter  22 value -1.351859
## iter  23 value -1.351932
## iter  24 value -1.352073
## iter  25 value -1.352252
## iter  26 value -1.352391
## iter  27 value -1.352416
## iter  28 value -1.352463
## iter  29 value -1.352465
## iter  30 value -1.352465
## iter  31 value -1.352466
## iter  32 value -1.352466
## iter  32 value -1.352466
## iter  33 value -1.352466
## iter  33 value -1.352466
## iter  33 value -1.352466
## final  value -1.352466
```

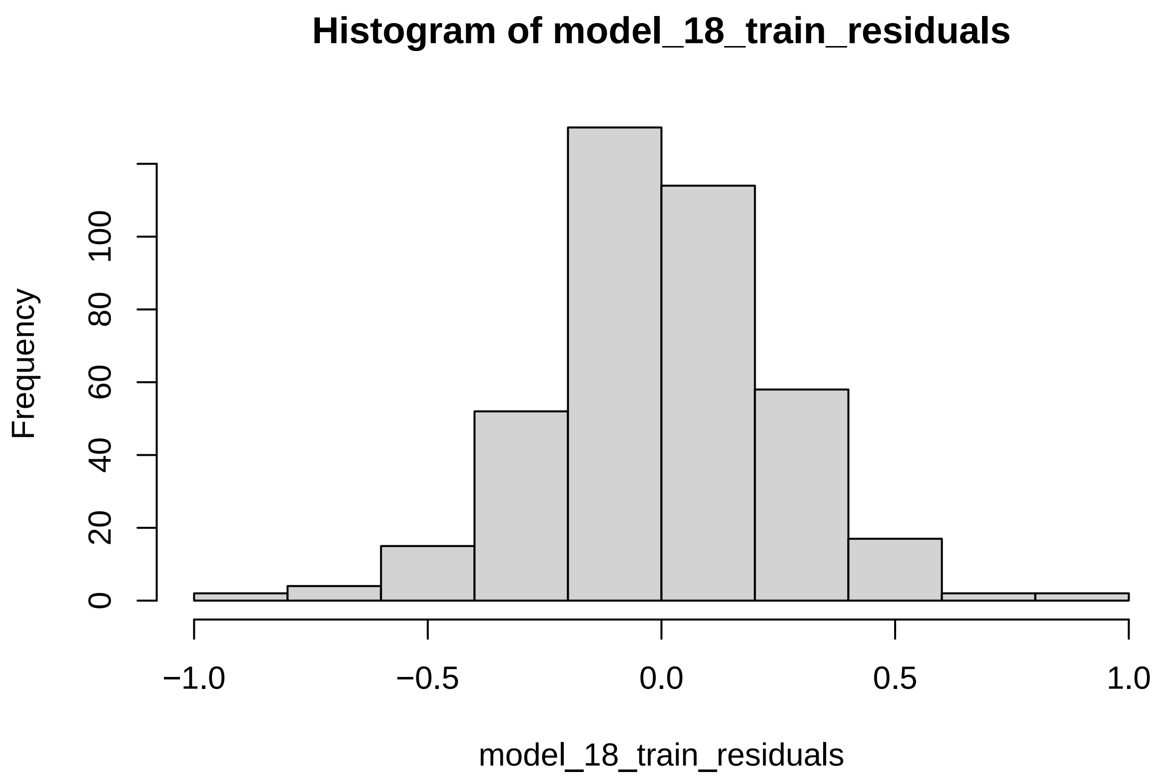
```
## converged
## initial  value -1.352034
## iter   2 value -1.352248
## iter   3 value -1.352661
## iter   4 value -1.352988
## iter   5 value -1.353009
## iter   6 value -1.353020
## iter   7 value -1.353021
## iter   8 value -1.353021
## iter   9 value -1.353021
## iter   9 value -1.353021
## iter   9 value -1.353021
## final  value -1.353021
## converged
```

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

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




```
model_18_train_residuals = resid(model_18_train$fit)
hist(model_18_train_residuals)
```



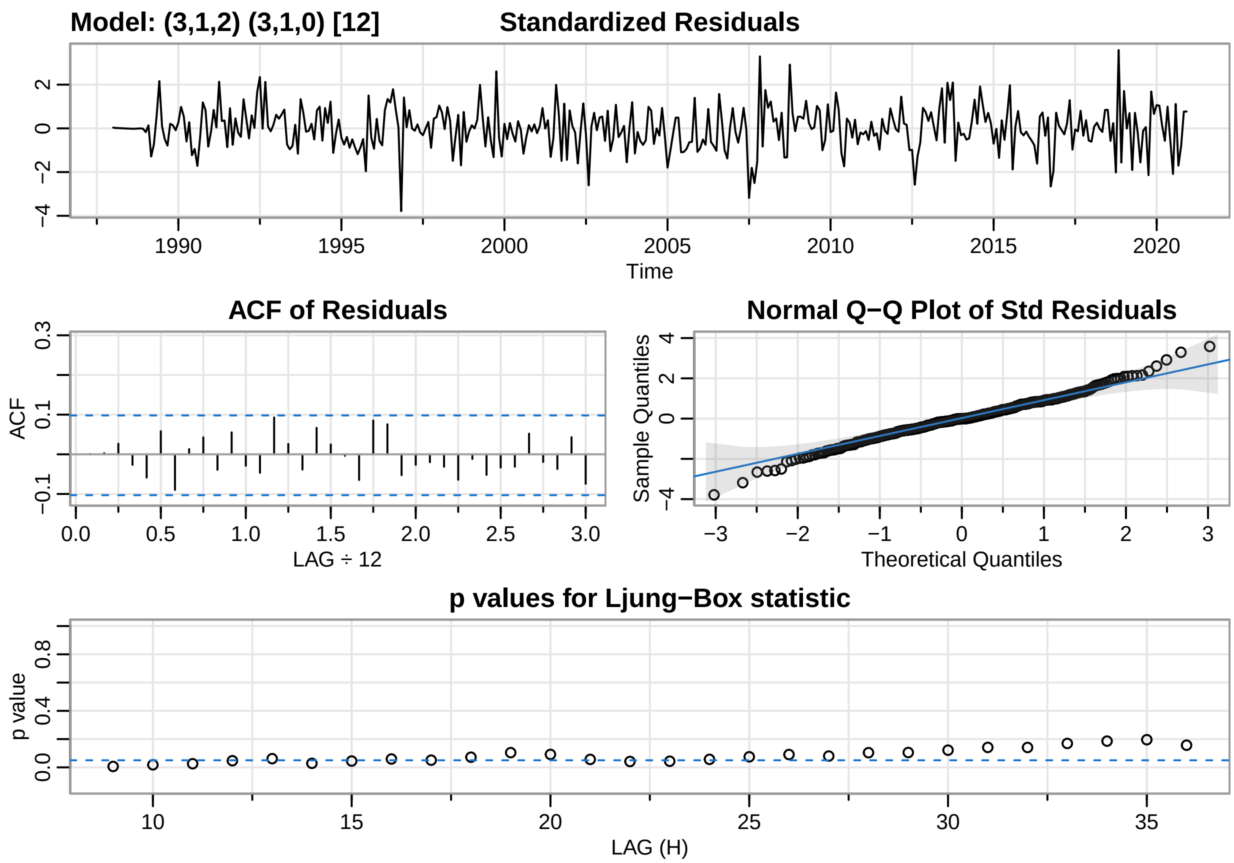

```
shapiro.test(model_18_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_18_train_residuals
## W = 0.9895, p-value = 0.00616
```

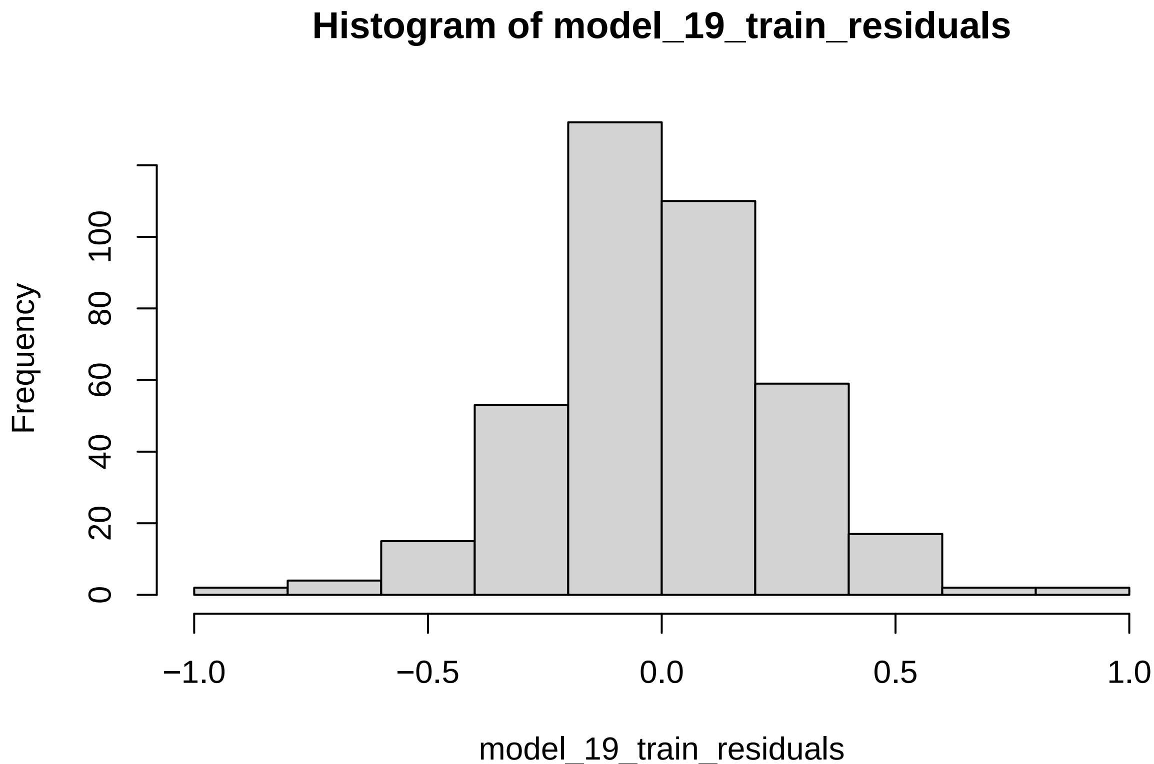
```
#SARIMA(3,1,2)x(3,1,0)_12
model_19_train <- sarima(Avg_ExtentTS_Train, p=3, d=1, q=2, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -1.036159
## iter   2 value -1.160007
## iter   3 value -1.259848
## iter   4 value -1.286934
## iter   5 value -1.296300
## iter   6 value -1.299039
## iter   7 value -1.331452
## iter   8 value -1.341032
## iter   9 value -1.349800
## iter  10 value -1.351699
## iter  11 value -1.352268
## iter  12 value -1.352842
## iter  13 value -1.354250
## iter  14 value -1.357163
## iter  15 value -1.359003
## iter  16 value -1.359630
## iter  17 value -1.360347
## iter  18 value -1.361172
## iter  18 value -1.361172
```

```
## iter  18 value -1.361172
## final  value -1.361172
## converged
## initial  value -1.349476
## iter   2 value -1.350545
## iter   3 value -1.352126
## iter   4 value -1.352892
## iter   5 value -1.353195
## iter   6 value -1.353227
## iter   7 value -1.353242
## iter   8 value -1.353248
## iter   9 value -1.353255
## iter  10 value -1.353260
## iter  11 value -1.353313
## iter  12 value -1.353376
## iter  13 value -1.353392
## iter  14 value -1.353425
## iter  15 value -1.353444
## iter  16 value -1.353444
## iter  17 value -1.353446
## iter  17 value -1.353445
## final  value -1.353446
## converged
```




```
model_19_train_residuals = resid(model_19_train$fit)
hist(model_19_train_residuals)
```




```
shapiro.test(model_19_train_residuals)
```

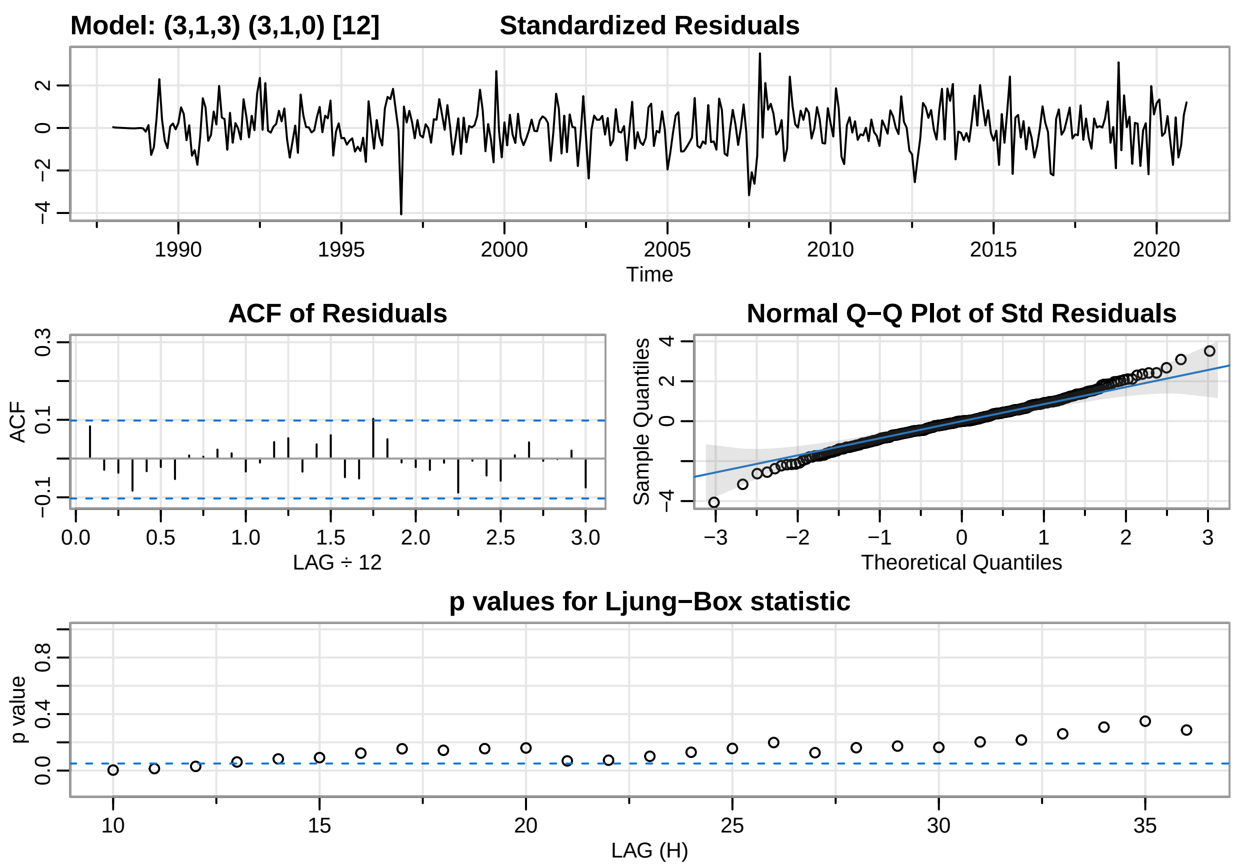
```
##
##  Shapiro-Wilk normality test
##
## data:  model_19_train_residuals
## W = 0.98931, p-value = 0.005438
```

```
#SARIMA(3,1,3)x(3,1,0)_12
model_20_train <- sarima(Avg_ExtentTS_Train, p=3, d=1, q=3, P=3, D=1, Q=0, S=12 , details = TRUE)
```

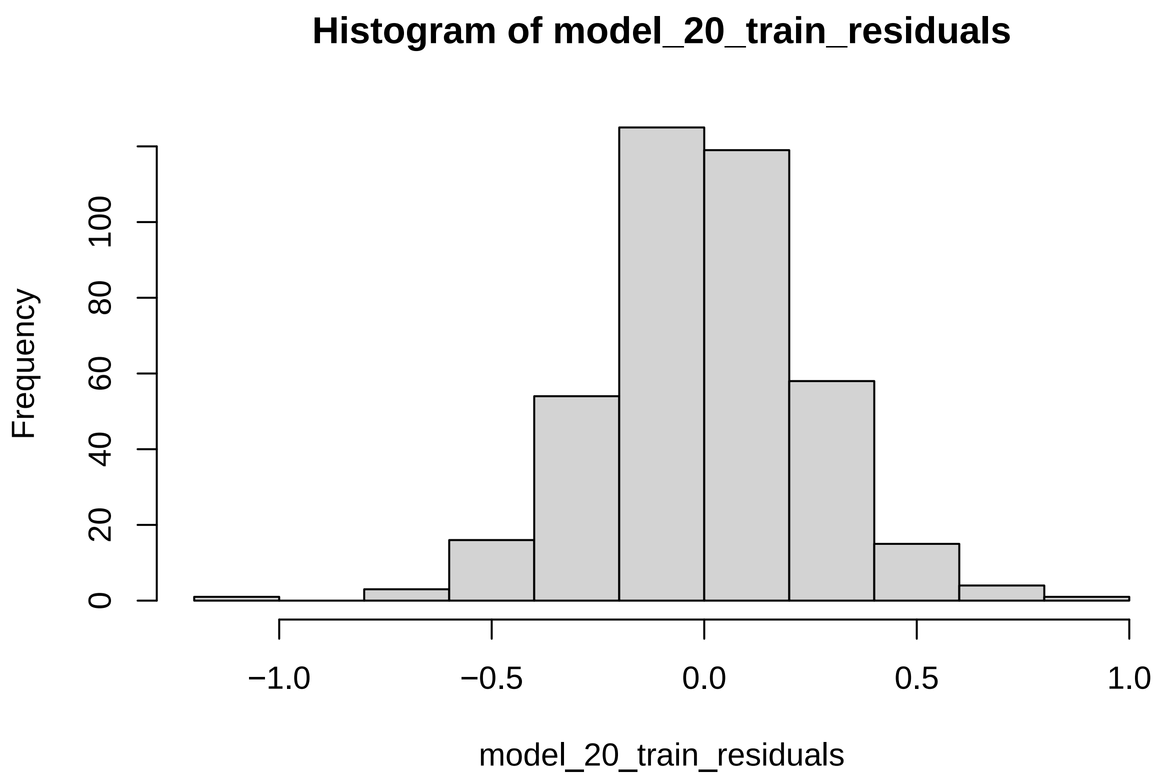
```
## initial  value -1.036159
## iter   2 value -1.164260
## iter   3 value -1.261457
## iter   4 value -1.294471
## iter   5 value -1.310799
## iter   6 value -1.331362
## iter   7 value -1.332748
## iter   8 value -1.340710
## iter   9 value -1.342401
## iter  10 value -1.344768
## iter  11 value -1.356444
## iter  12 value -1.358054
## iter  13 value -1.361687
## iter  14 value -1.361893
## iter  15 value -1.361951
## iter  16 value -1.362041
## iter  16 value -1.362041
## iter  17 value -1.362048
## iter  17 value -1.362048
```

```
## iter  18 value -1.362055
## iter  18 value -1.362055
## iter  19 value -1.362057
## iter  19 value -1.362057
## iter  19 value -1.362057
## final  value -1.362057
## converged
## initial  value -1.349245
## iter   2 value -1.350423
## iter   3 value -1.351430
## iter   4 value -1.352904
## iter   5 value -1.352954
## iter   6 value -1.353039
## iter   7 value -1.353175
## iter   8 value -1.353367
## iter   9 value -1.353555
## iter  10 value -1.353680
## iter  11 value -1.353702
## iter  12 value -1.353712
## iter  13 value -1.353734
## iter  14 value -1.353767
## iter  15 value -1.353868
## iter  16 value -1.353965
## iter  17 value -1.354067
## iter  18 value -1.354333
## iter  19 value -1.354582
## iter  20 value -1.354746
## iter  21 value -1.354794
## iter  22 value -1.354799
## iter  23 value -1.354846
## iter  24 value -1.354906
## iter  25 value -1.355108
## iter  26 value -1.356058
## iter  27 value -1.356158
## iter  28 value -1.356195
## iter  29 value -1.357452
## iter  30 value -1.358237
## iter  31 value -1.358849
## iter  32 value -1.359578
## iter  33 value -1.360331
## iter  34 value -1.361291
## iter  35 value -1.361471
## iter  36 value -1.362078
## iter  37 value -1.362381
## iter  38 value -1.362544
## iter  39 value -1.362731
## iter  40 value -1.362773
## iter  41 value -1.362816
## iter  42 value -1.362843
## iter  43 value -1.362863
## iter  44 value -1.362906
## iter  45 value -1.362946
## iter  46 value -1.363173
## iter  47 value -1.363342
```

```
## iter  48 value -1.363621
## iter  49 value -1.363728
## iter  50 value -1.363844
## iter  51 value -1.363959
## iter  52 value -1.364065
## iter  53 value -1.364278
## iter  54 value -1.364305
## iter  55 value -1.364463
## iter  56 value -1.364505
## iter  57 value -1.364535
## iter  58 value -1.364554
## iter  59 value -1.364565
## iter  60 value -1.364576
## iter  61 value -1.364584
## iter  62 value -1.364587
## iter  63 value -1.364587
## iter  64 value -1.364587
## iter  64 value -1.364587
## iter  64 value -1.364587
## final  value -1.364587
## converged
```




```
model_20_train_residuals = resid(model_20_train$fit)
hist(model_20_train_residuals)
```




```
shapiro.test(model_20_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_20_train_residuals
## W = 0.99038, p-value = 0.01088
```

```
library(huxtable)
```

```
##
## Attaching package: 'huxtable'

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

```
goodness_of_fit <- hux(
        Model = c('SARIMA(0,1,2)x(0,1,1)_12', 'SARIMA(0,1,2)x(3,1,0)_12', 'SARIMA(0,1,2)x(1,1,1)_12',
                  'SARIMA(4,1,0)x(0,1,1)_12', 'SARIMA(4,1,0)x(3,1,0)_12', 'SARIMA(4,1,0)x(1,1,1)_12',
                  'SARIMA(5,1,0)x(0,1,1)_12', 'SARIMA(5,1,0)x(3,1,0)_12', 'SARIMA(5,1,0)x(1,1,1)_12',
                  'SARIMA(1,1,1)x(0,1,1)_12', 'SARIMA(1,1,1)x(3,1,0)_12', 'SARIMA(1,1,1)x(1,1,1)_12',
                  'SARIMA(1,1,2)x(3,1,0)_12', 'SARIMA(2,1,1)x(3,1,0)_12', 'SARIMA(2,1,2)x(3,1,0)_12',
                  'SARIMA(1,1,3)x(3,1,0)_12', 'SARIMA(2,1,3)x(3,1,0)_12', 'SARIMA(3,1,2)x(3,1,0)_12',
                  'SARIMA(1,1,3)x(3,1,0)_12'),
        AIC = c(model_1_train$AIC, model_2_train$AIC, model_3_train$AIC,
                model_4_train$AIC, model_5_train$AIC, model_6_train$AIC,
                model_7_train$AIC, model_8_train$AIC, model_9_train$AIC,
                model_10_train$AIC, model_11_train$AIC, model_12_train$AIC,
                model_13_train$AIC, model_14_train$AIC, model_15_train$AIC,
                model_16_train$AIC, model_18_train$AIC, model_19_train$AIC,
```

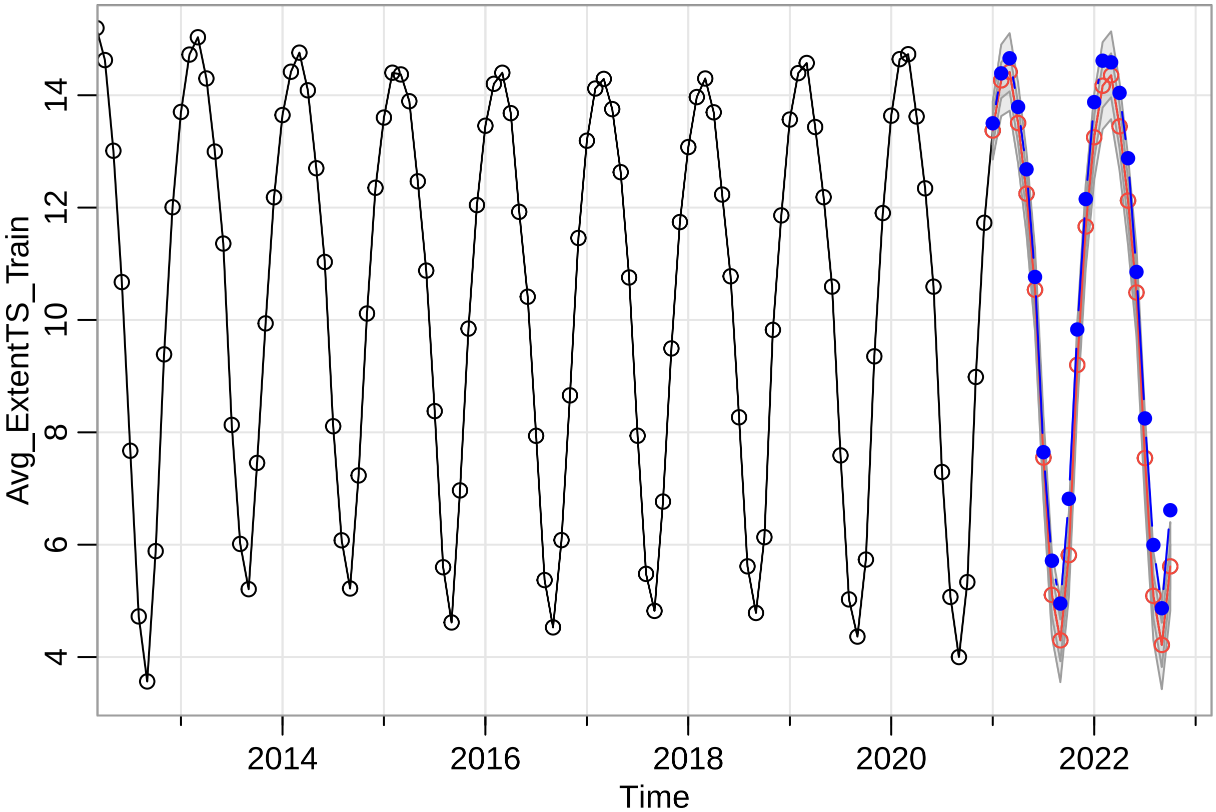
```
               model_20_train$AIC),
       AICc = c(model_1_train$AICc, model_2_train$AICc, model_3_train$AICc,
               model_4_train$AICc, model_5_train$AICc, model_6_train$AICc,
               model_7_train$AICc, model_8_train$AICc, model_9_train$AICc,
               model_10_train$AICc, model_11_train$AICc, model_12_train$AICc,
               model_13_train$AICc, model_14_train$AICc, model_15_train$AICc,
               model_16_train$AICc, model_18_train$AICc, model_19_train$AICc,
               model_20_train$AICc),
       BIC = c(model_1_train$BIC, model_2_train$BIC, model_3_train$BIC,
               model_4_train$BIC, model_5_train$BIC, model_6_train$BIC,
               model_7_train$BIC, model_8_train$BIC, model_9_train$BIC,
               model_10_train$BIC, model_11_train$BIC, model_12_train$BIC,
               model_13_train$BIC, model_14_train$BIC, model_15_train$BIC,
               model_16_train$BIC, model_18_train$BIC, model_19_train$BIC,
               model_20_train$BIC),
       MSE = c(mean(model_1_train_residuals^2), mean(model_2_train_residuals^2), mean(model_3_train_re
               mean(model_4_train_residuals^2), mean(model_5_train_residuals^2), mean(model_6_train_re
               mean(model_7_train_residuals^2), mean(model_8_train_residuals^2), mean(model_9_train_re
               mean(model_10_train_residuals^2), mean(model_11_train_residuals^2), mean(model_12_train
               mean(model_13_train_residuals^2), mean(model_14_train_residuals^2), mean(model_15_train
               mean(model_16_train_residuals^2), mean(model_18_train_residuals^2), mean(model_19_train
               mean(model_20_train_residuals^2))
     )

goodness_of_fit %>%
  set_number_format(col=c(2,3,4,5), value=3) %>%
  set_bottom_border(c(1,13,16), everywhere) %>%
  set_bold(c(12,14,15,16,17,18,19,20), everywhere) %>%
  set_background_color(c(12,14,15,16,17,18,19,20), everywhere, "grey95")
```

Model	AIC	AICc	BIC	MSE
SARIMA(0,1,2)x(0,1,1)_12	0.260	0.260	0.302	0.070
SARIMA(0,1,2)x(3,1,0)_12	0.274	0.275	0.336	0.071
SARIMA(0,1,2)x(1,1,1)_12	0.264	0.264	0.315	0.070
SARIMA(4,1,0)x(0,1,1)_12	0.258	0.258	0.320	0.069
SARIMA(4,1,0)x(3,1,0)_12	0.275	0.276	0.358	0.070
SARIMA(4,1,0)x(1,1,1)_12	0.261	0.262	0.333	0.069
SARIMA(5,1,0)x(0,1,1)_12	0.247	0.248	0.320	0.068
SARIMA(5,1,0)x(3,1,0)_12	0.264	0.265	0.357	0.069
SARIMA(5,1,0)x(1,1,1)_12	0.251	0.252	0.333	0.068
SARIMA(1,1,1)x(0,1,1)_12	0.305	0.305	0.346	0.073
SARIMA(1,1,1)x(3,1,0)_12	**0.195**	**0.195**	**0.257**	**0.064**
SARIMA(1,1,1)x(1,1,1)_12	0.308	0.309	0.360	0.069
SARIMA(1,1,2)x(3,1,0)_12	**0.168**	**0.169**	**0.241**	**0.062**
SARIMA(2,1,1)x(3,1,0)_12	**0.171**	**0.171**	**0.243**	**0.062**
SARIMA(2,1,2)x(3,1,0)_12	**0.173**	**0.174**	**0.255**	**0.062**
SARIMA(1,1,3)x(3,1,0)_12	**0.173**	**0.174**	**0.256**	**0.062**
SARIMA(2,1,3)x(3,1,0)_12	**0.179**	**0.180**	**0.272**	**0.062**
SARIMA(3,1,2)x(3,1,0)_12	**0.178**	**0.179**	**0.271**	**0.062**
SARIMA(1,1,3)x(3,1,0)_12	**0.161**	**0.162**	**0.264**	**0.060**

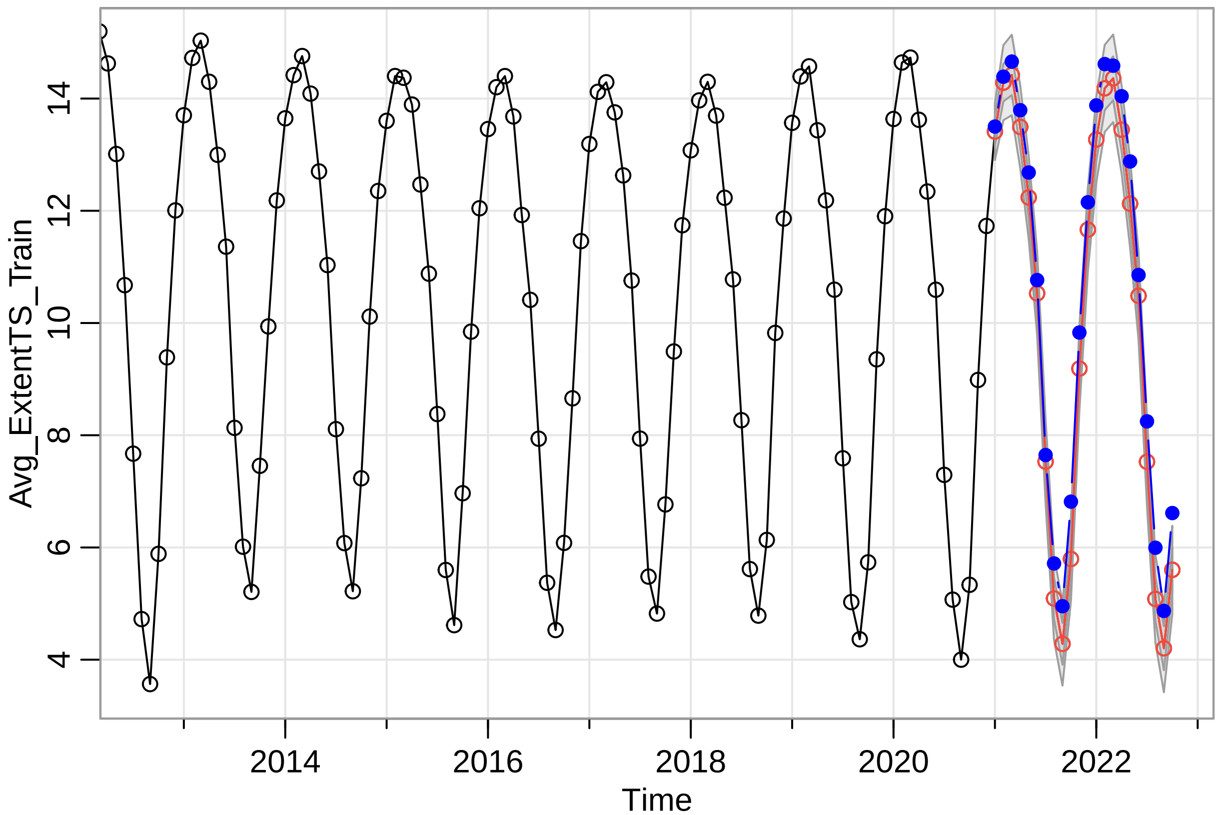
.

Model Selection

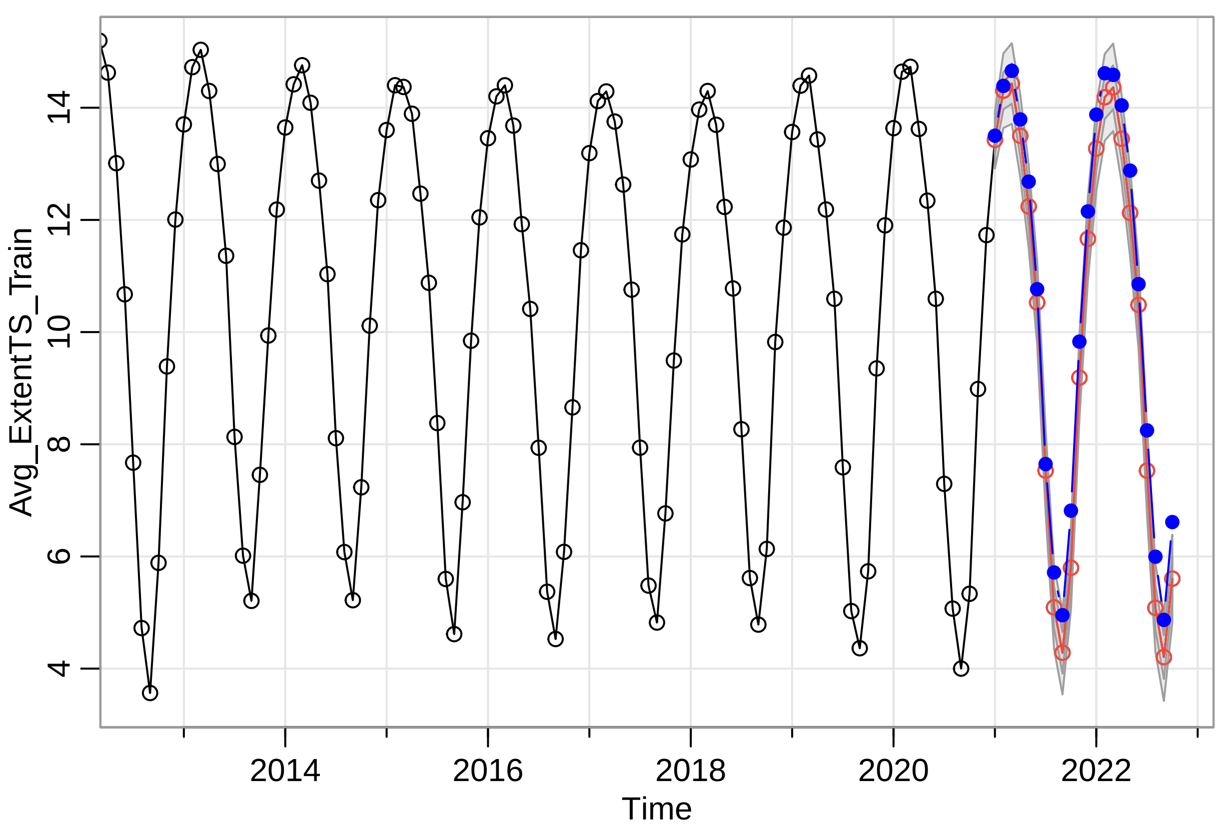
```
model_8_train_forecast<- sarima.for(Avg_ExtentTS_Train, n.ahead=22, p=1,d=1,q=1,P=3,D=1,Q=0,S=12)
lines(Avg_ExtentTS_Test,col='blue',type='b',pch=16)
```



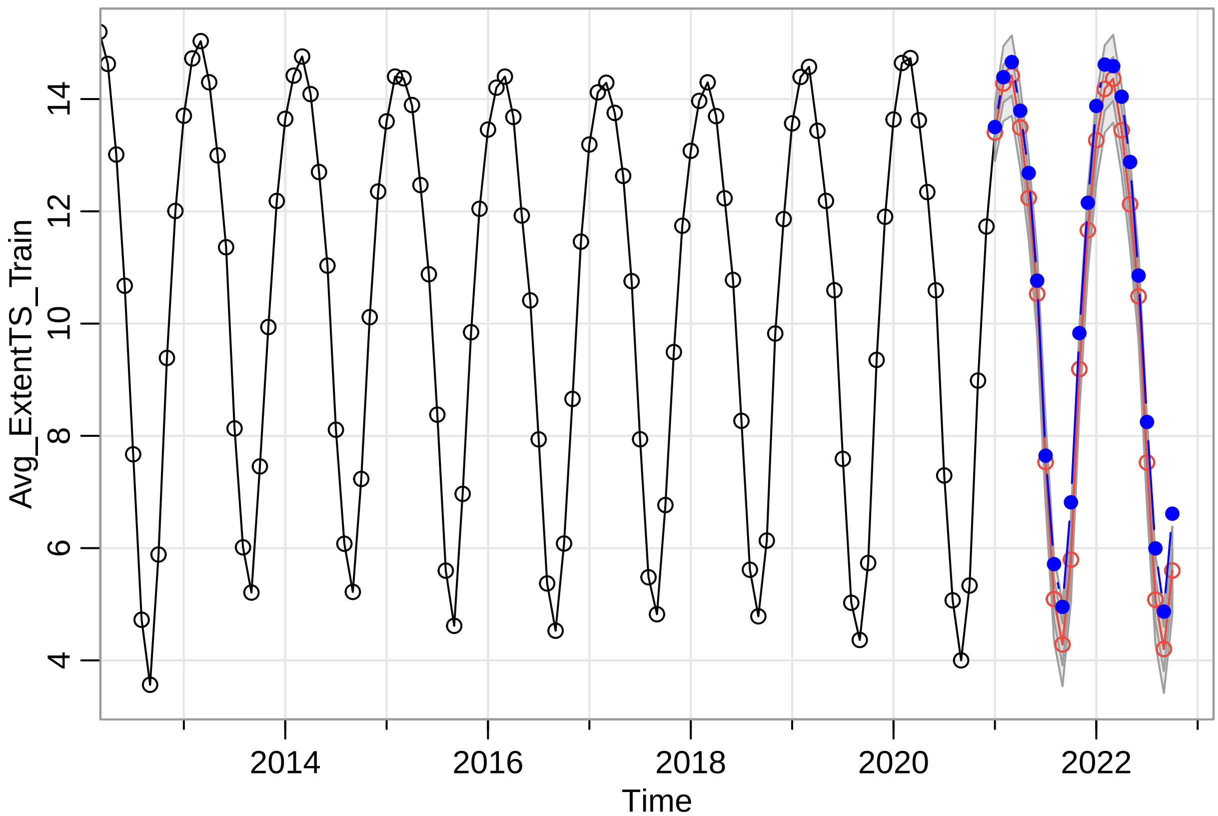

```
model_10_train_forecast<- sarima.for(Avg_ExtentTS_Train, n.ahead=22, p=1,d=1,q=2,P=3,D=1,Q=0,S=12)
lines(Avg_ExtentTS_Test,col='blue',type='b',pch=16)
```




```
model_11_train_forecast<- sarima.for(Avg_ExtentTS_Train, n.ahead=22, p=2,d=1,q=1,P=3,D=1,Q=0,S=12)
lines(Avg_ExtentTS_Test,col='blue',type='b',pch=16)
```



```
model_12_train_forecast<- sarima.for(Avg_ExtentTS_Train, n.ahead=22, p=2,d=1,q=2,P=3,D=1,Q=0,S=12)
lines(Avg_ExtentTS_Test,col='blue',type='b',pch=16)
```




```
mean((model_8_train_forecast$pred-Avg_ExtentTS_Test)^2)
```

```
## [1] 0.3321616
```

```
mean((model_10_train_forecast$pred-Avg_ExtentTS_Test)^2)
```

```
## [1] 0.3385571
```

```
mean((model_11_train_forecast$pred-Avg_ExtentTS_Test)^2)
```

```
## [1] 0.3363344
```

```
mean((model_12_train_forecast$pred-Avg_ExtentTS_Test)^2)
```

```
## [1] 0.3388543
```

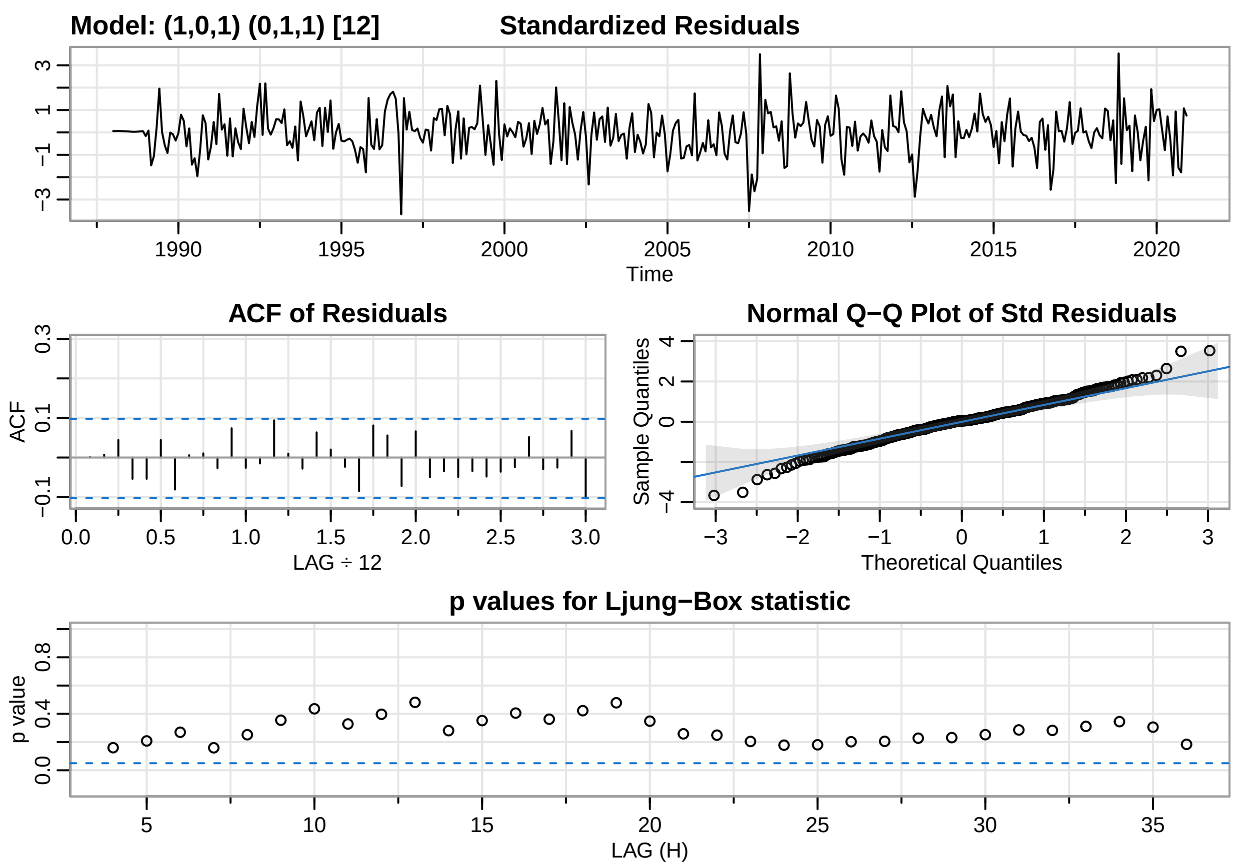
#TODO could test more combinations of ARMA models, athough not necessary #TODO decide on a final model, and make predictions

Model Fitting on Seasonally differenced data

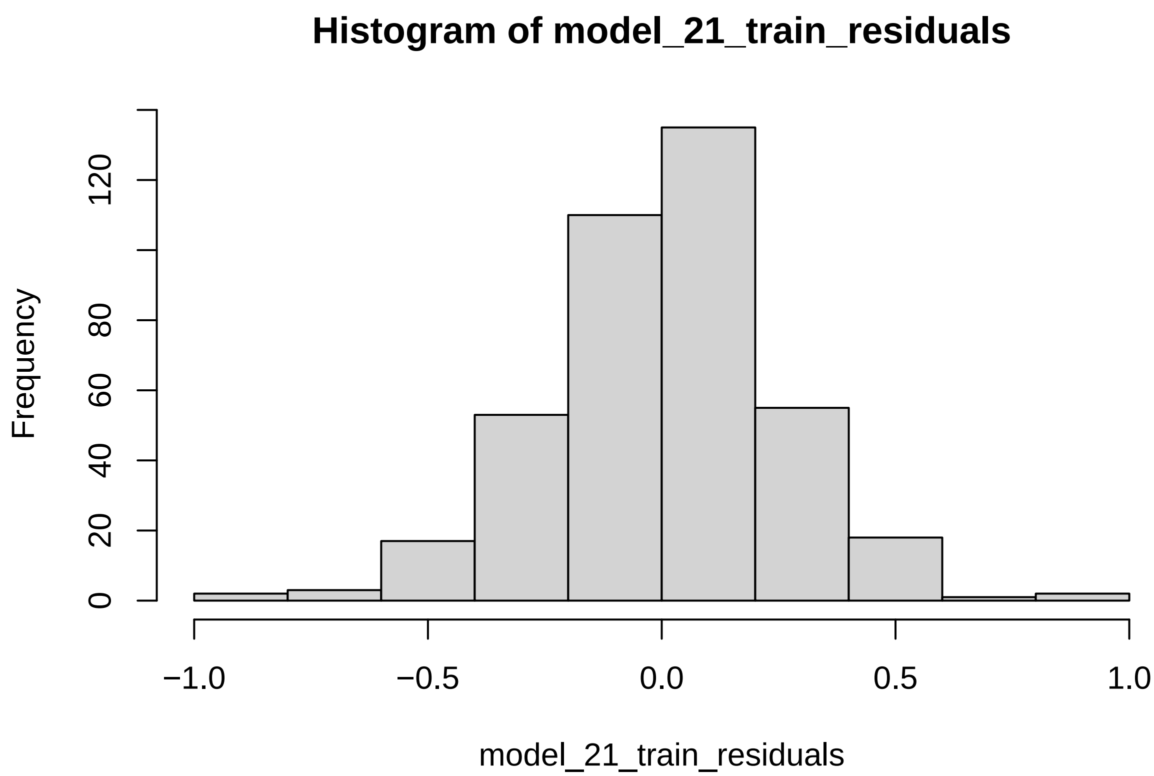
```
#SARIMA(1,0,1)x(0,1,1)_12
model_21_train <- sarima(Avg_ExtentTS_Train, p=1, d=0, q=1, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -0.801086
## iter   2 value -1.133503
## iter   3 value -1.361046
```

```
## iter   4 value -1.364659
## iter   5 value -1.367122
## iter   6 value -1.367636
## iter   7 value -1.367710
## iter   8 value -1.367744
## iter   9 value -1.367751
## iter  10 value -1.367751
## iter  10 value -1.367751
## final  value -1.367751
## converged
## initial  value -1.365232
## iter   2 value -1.365266
## iter   3 value -1.365293
## iter   4 value -1.365301
## iter   5 value -1.365301
## iter   6 value -1.365301
## iter   6 value -1.365301
## iter   6 value -1.365301
## final  value -1.365301
## converged
```




```
model_21_train_residuals = resid(model_21_train$fit)
hist(model_21_train_residuals)
```



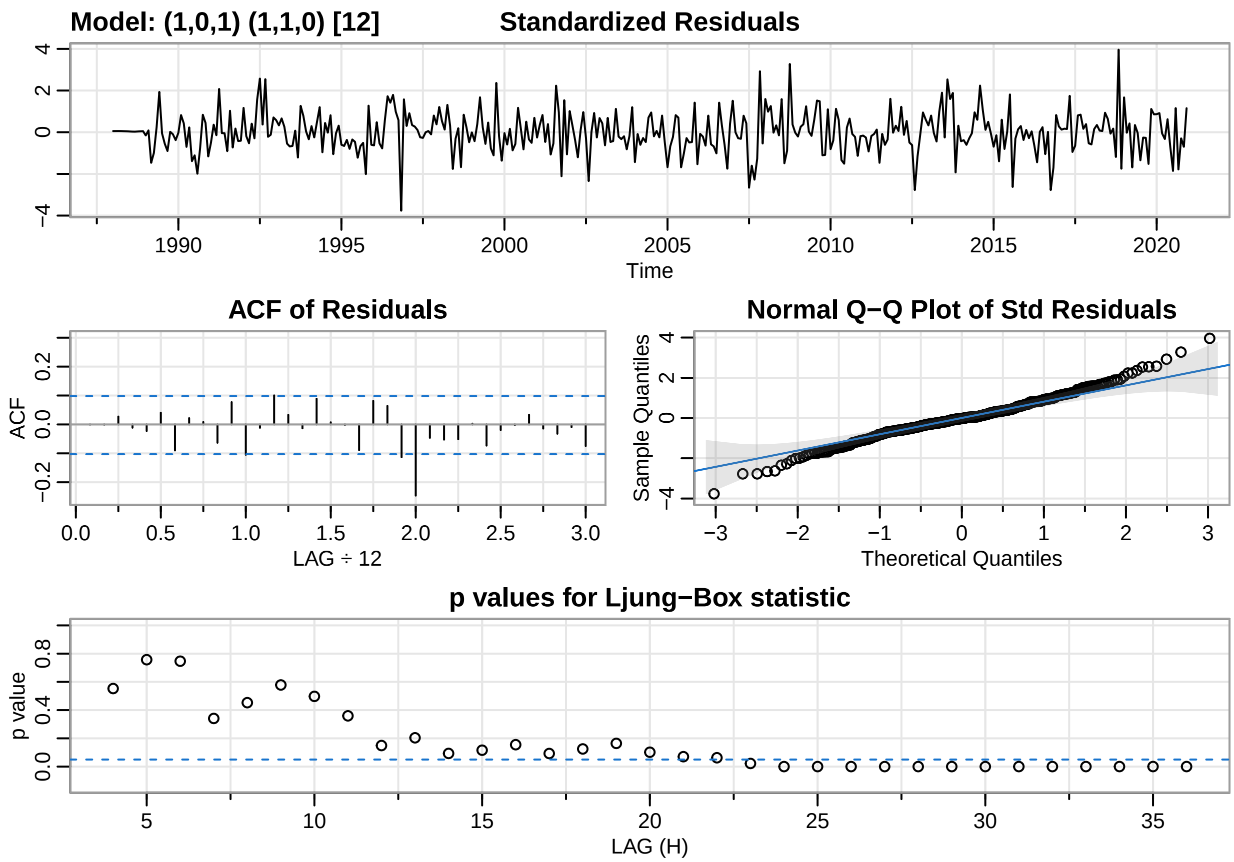

```
shapiro.test(model_21_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_21_train_residuals
## W = 0.98802, p-value = 0.002432
```

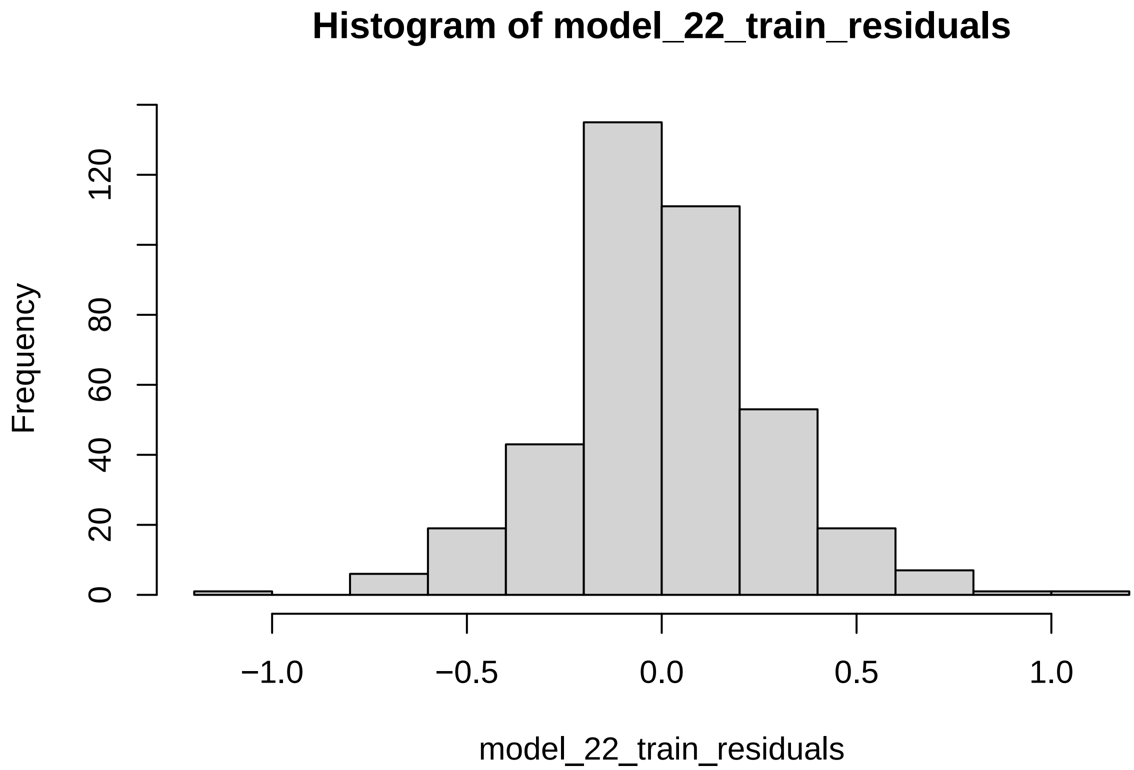
```
#SARIMA(1,0,1)x(1,1,0)_12
model_22_train <- sarima(Avg_ExtentTS_Train, p=1, d=0, q=1, P=1, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -0.794710
## iter   2 value -1.125638
## iter   3 value -1.297146
## iter   4 value -1.306046
## iter   5 value -1.306798
## iter   6 value -1.306962
## iter   7 value -1.307040
## iter   8 value -1.307043
## iter   9 value -1.307044
## iter  10 value -1.307044
## iter  11 value -1.307044
## iter  12 value -1.307044
## iter  12 value -1.307044
## iter  12 value -1.307044
## final  value -1.307044
## converged
## initial  value -1.307068
## iter   2 value -1.307118
## iter   3 value -1.307150
```

```
## iter   4 value -1.307158
## iter   5 value -1.307161
## iter   6 value -1.307161
## iter   7 value -1.307161
## iter   8 value -1.307161
## iter   8 value -1.307161
## iter   8 value -1.307161
## final  value -1.307161
## converged
```




```
model_22_train_residuals = resid(model_22_train$fit)
hist(model_22_train_residuals)
```



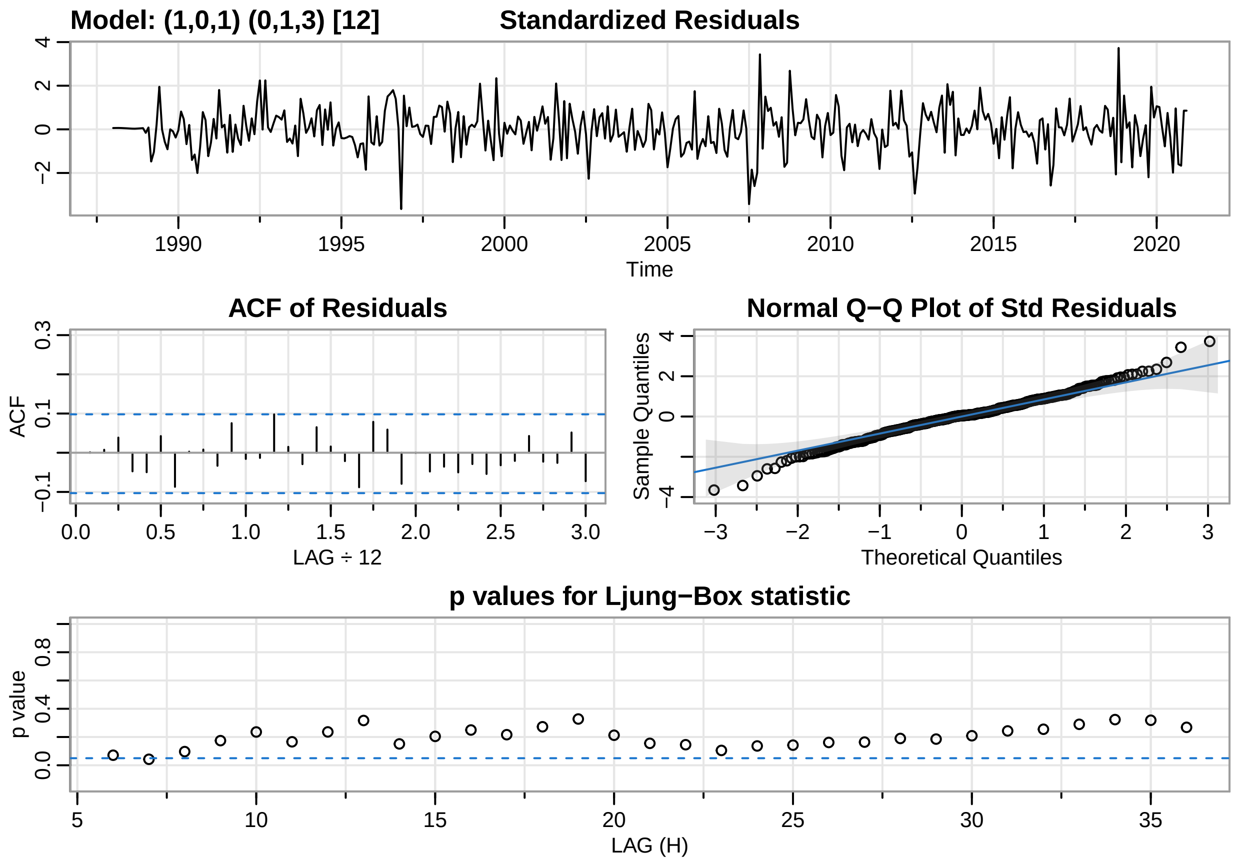

```
shapiro.test(model_22_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_22_train_residuals
## W = 0.98428, p-value = 0.0002677
```

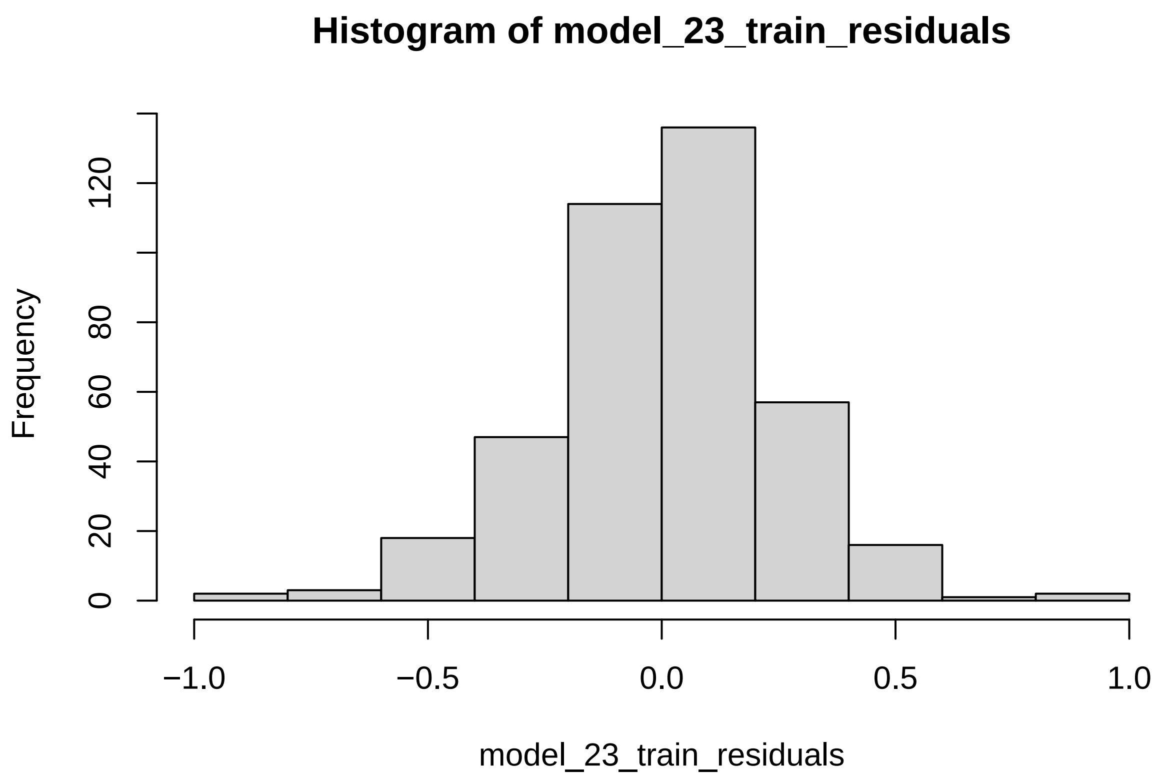
```
#SARIMA(1,0,1)x(0,1,3)_12
model_23_train <- sarima(Avg_ExtentTS_Train, p=1, d=0, q=1, P=0, D=1, Q=3, S=12 , details = TRUE)
```

```
## initial  value -0.801086
## iter   2 value -1.144412
## iter   3 value -1.344981
## iter   4 value -1.357046
## iter   5 value -1.368996
## iter   6 value -1.371115
## iter   7 value -1.371624
## iter   8 value -1.371650
## iter   9 value -1.371662
## iter  10 value -1.371662
## iter  11 value -1.371662
## iter  11 value -1.371662
## iter  11 value -1.371662
## final  value -1.371662
## converged
## initial  value -1.368723
## iter   2 value -1.368752
## iter   3 value -1.368772
## iter   4 value -1.368806
```

```
## iter   5 value -1.368811
## iter   6 value -1.368813
## iter   7 value -1.368813
## iter   7 value -1.368813
## iter   7 value -1.368813
## final  value -1.368813
## converged
```




```
model_23_train_residuals = resid(model_23_train$fit)
hist(model_23_train_residuals)
```



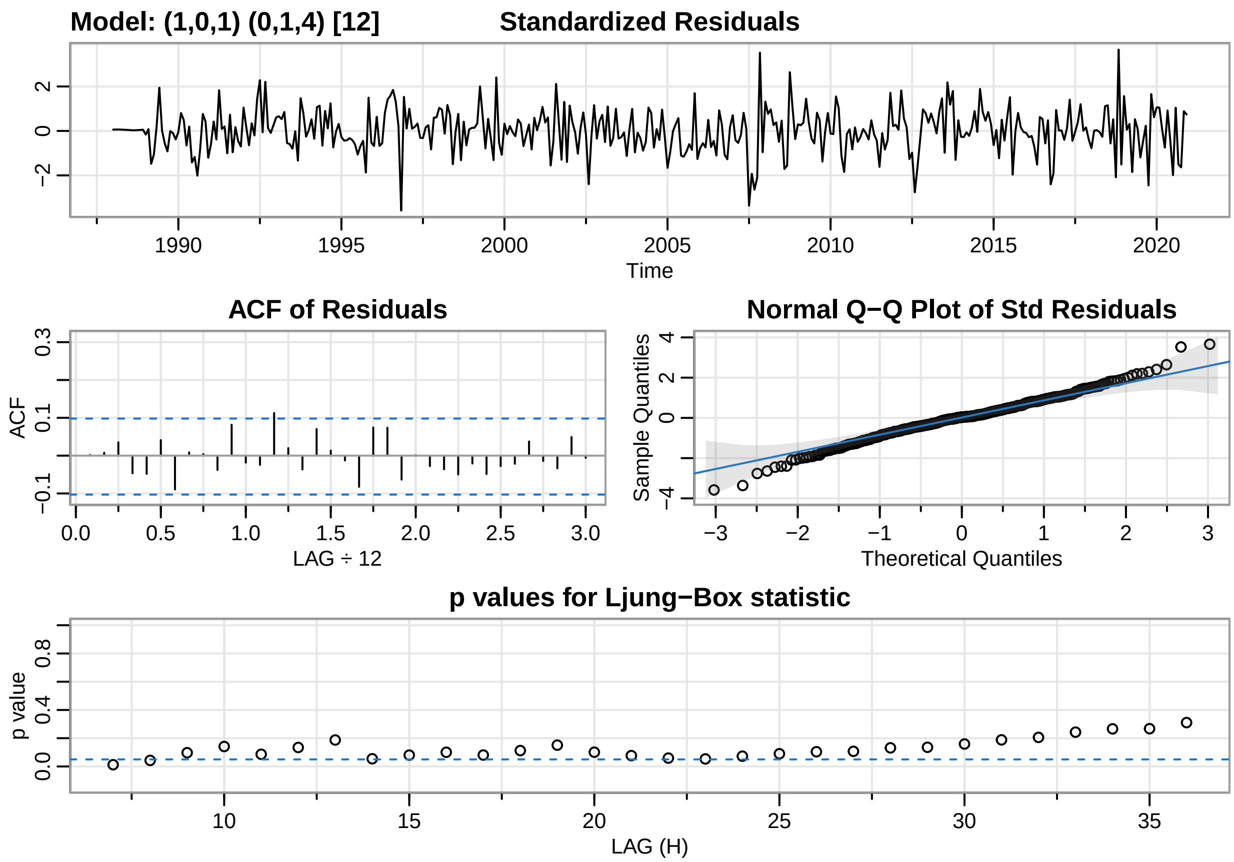

```
shapiro.test(model_23_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_23_train_residuals
## W = 0.98706, p-value = 0.001346
```

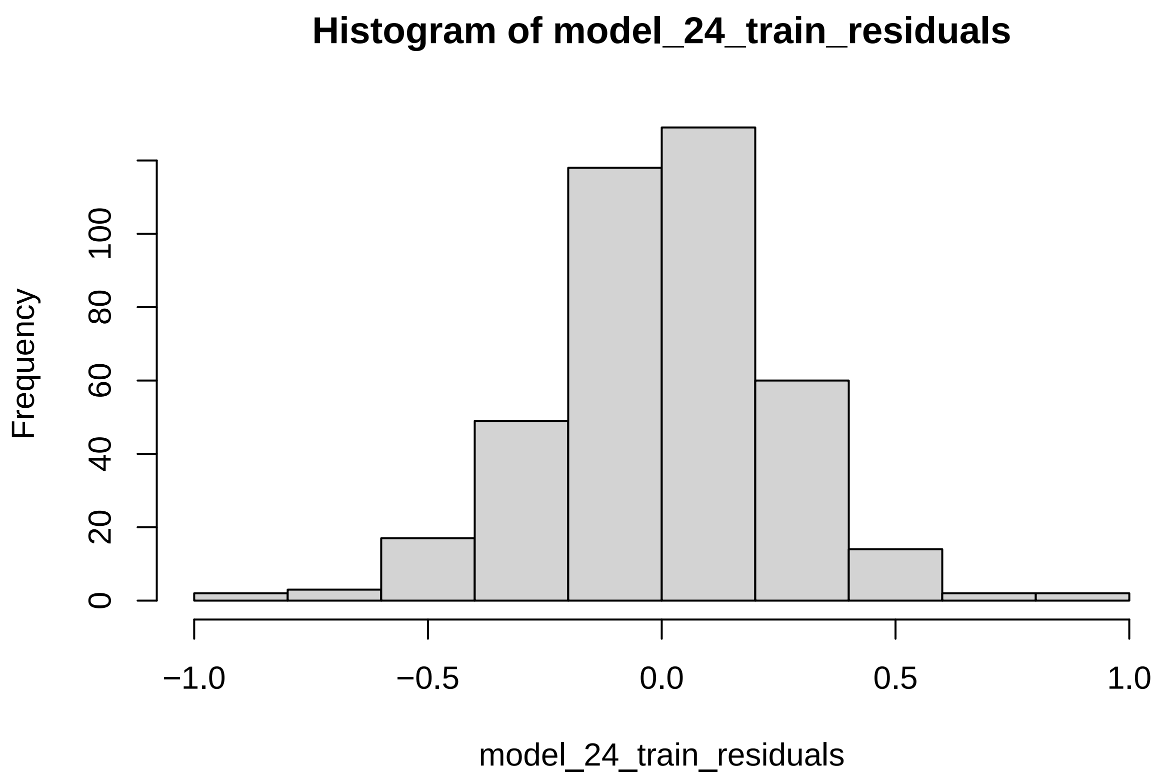
```
#SARIMA(1,0,1)x(0,1,4)_12
model_24_train <- sarima(Avg_ExtentTS_Train, p=1, d=0, q=1, P=0, D=1, Q=4, S=12 , details = TRUE)
```

```
## initial  value -0.801086
## iter   2 value -1.140892
## iter   3 value -1.368903
## iter   4 value -1.371731
## iter   5 value -1.373468
## iter   6 value -1.375135
## iter   7 value -1.375394
## iter   8 value -1.375462
## iter   9 value -1.375466
## iter  10 value -1.375469
## iter  11 value -1.375469
## iter  11 value -1.375469
## iter  11 value -1.375469
## final  value -1.375469
## converged
## initial  value -1.372802
## iter   2 value -1.372833
## iter   3 value -1.372871
## iter   4 value -1.372893
```

```
## iter   5 value -1.372905
## iter   6 value -1.372905
## iter   6 value -1.372905
## iter   6 value -1.372905
## final  value -1.372905
## converged
```




```
model_24_train_residuals = resid(model_24_train$fit)
hist(model_24_train_residuals)
```



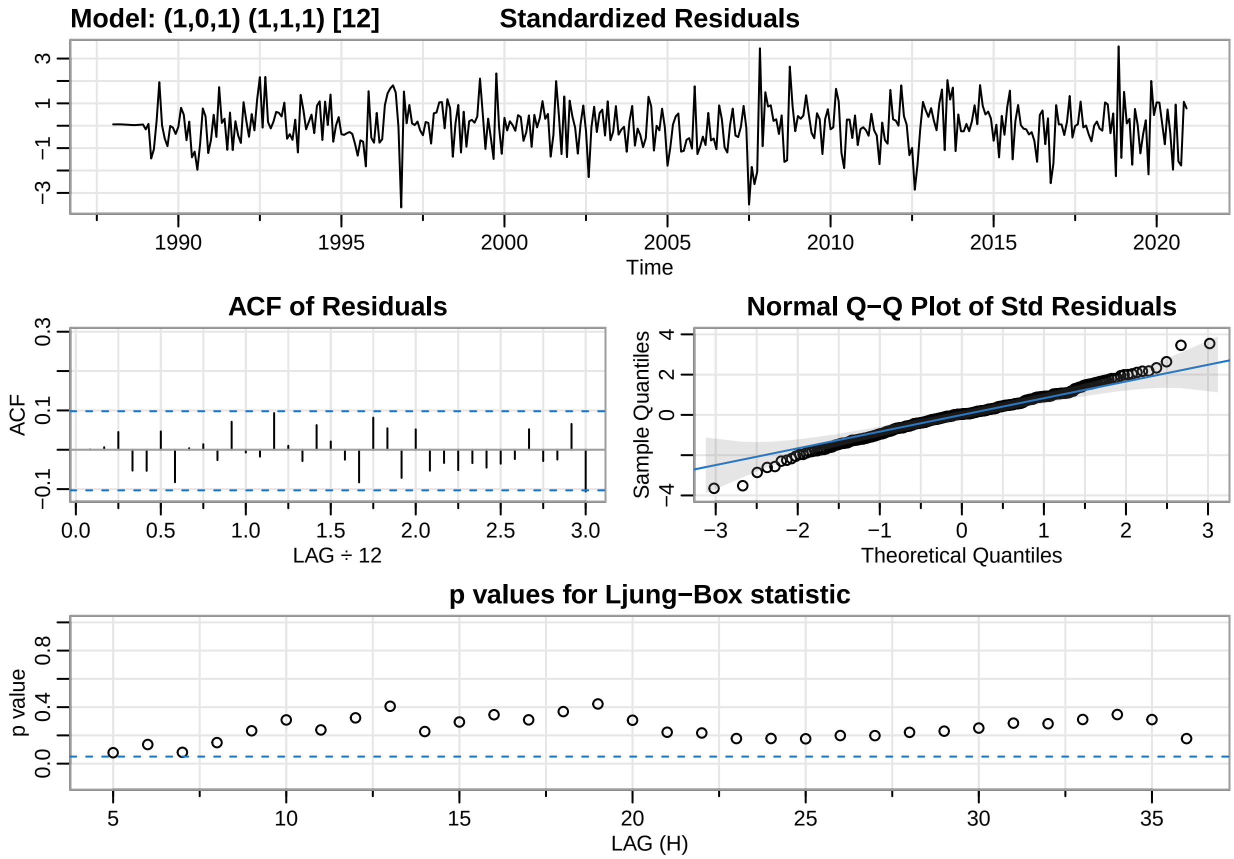

```r
shapiro.test(model_24_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_24_train_residuals
## W = 0.98849, p-value = 0.003246
```

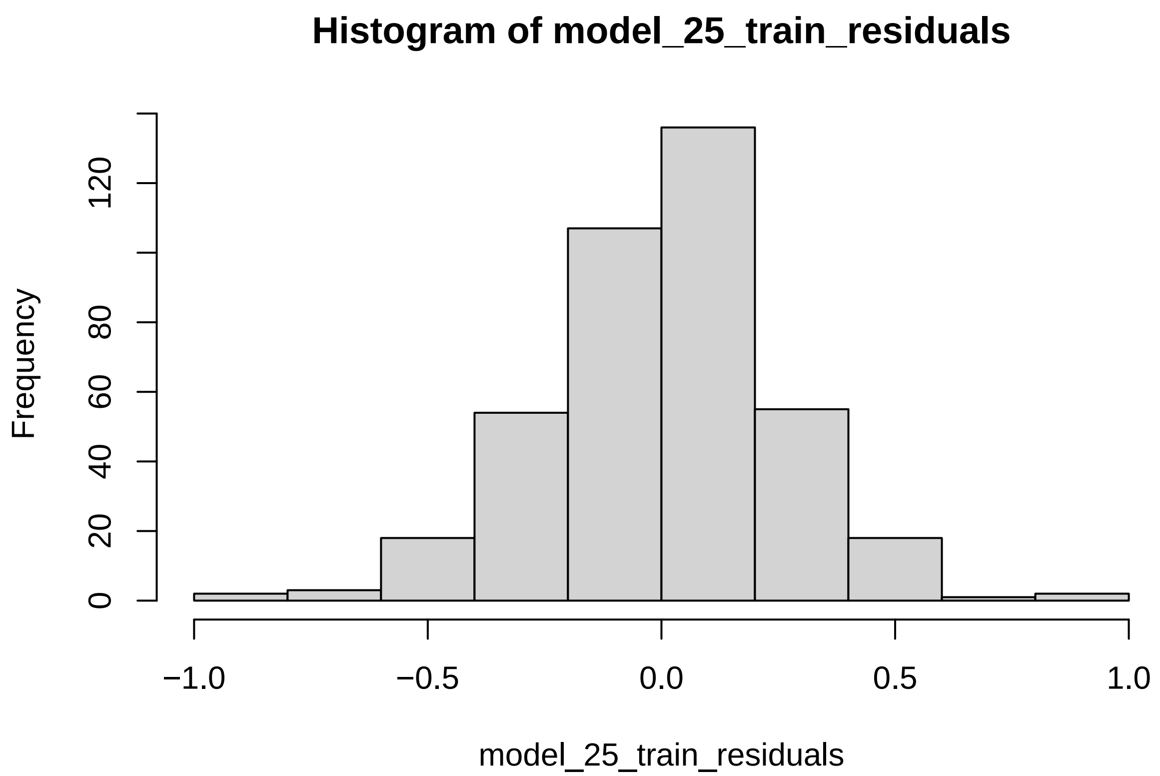
```r
#SARIMA(1,0,1)x(1,1,1)_12
model_25_train <- sarima(Avg_ExtentTS_Train, p=1, d=0, q=1, P=1, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -0.794710
## iter   2 value -1.152892
## iter   3 value -1.325426
## iter   4 value -1.345857
## iter   5 value -1.352890
## iter   6 value -1.359234
## iter   7 value -1.360827
## iter   8 value -1.361190
## iter   9 value -1.361198
## iter  10 value -1.361203
## iter  11 value -1.361205
## iter  12 value -1.361205
## iter  12 value -1.361205
## final  value -1.361205
## converged
## initial  value -1.364327
## iter   2 value -1.364958
## iter   3 value -1.365484
## iter   4 value -1.365574
```

```
## iter   5 value -1.365589
## iter   6 value -1.365590
## iter   7 value -1.365590
## iter   8 value -1.365590
## iter   8 value -1.365590
## iter   8 value -1.365590
## final  value -1.365590
## converged
```




```
model_25_train_residuals = resid(model_25_train$fit)
hist(model_25_train_residuals)
```



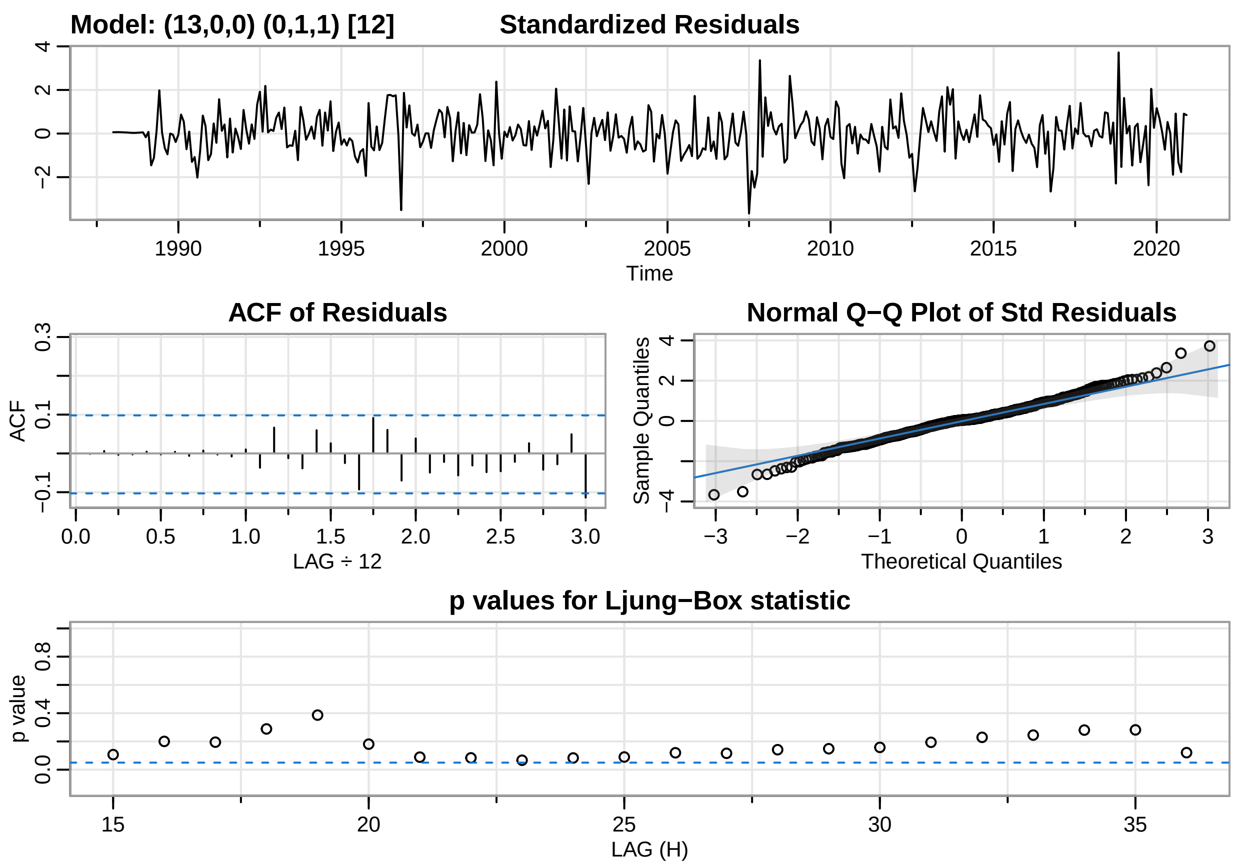

```r
shapiro.test(model_25_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_25_train_residuals
## W = 0.98841, p-value = 0.00309
```

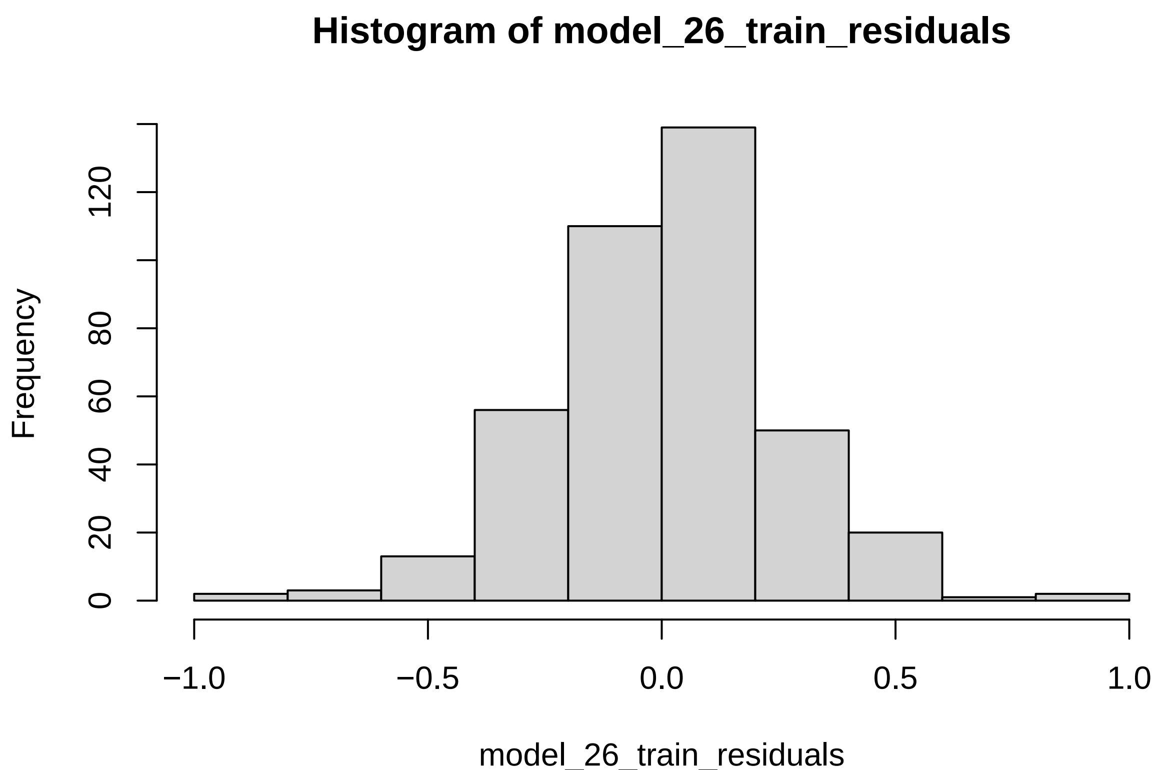
```r
#SARIMA(13,0,0)x(0,1,1)_12
model_26_train <- sarima(Avg_ExtentTS_Train, p=13, d=0, q=0, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -0.794710
## iter   2 value -1.008413
## iter   3 value -1.149498
## iter   4 value -1.247785
## iter   5 value -1.312226
## iter   6 value -1.340321
## iter   7 value -1.362277
## iter   8 value -1.368586
## iter   9 value -1.372893
## iter  10 value -1.376137
## iter  11 value -1.377199
## iter  12 value -1.377346
## iter  13 value -1.377423
## iter  14 value -1.377445
## iter  15 value -1.377456
## iter  16 value -1.377457
## iter  17 value -1.377457
## iter  18 value -1.377458
## iter  19 value -1.377458
```

```
## iter  19 value -1.377458
## iter  19 value -1.377458
## final  value -1.377458
## converged
## initial  value -1.378066
## iter   2 value -1.378820
## iter   3 value -1.379100
## iter   4 value -1.379263
## iter   5 value -1.379282
## iter   6 value -1.379285
## iter   7 value -1.379285
## iter   8 value -1.379285
## iter   8 value -1.379285
## iter   8 value -1.379285
## final  value -1.379285
## converged
```




```
model_26_train_residuals = resid(model_26_train$fit)
hist(model_26_train_residuals)
```



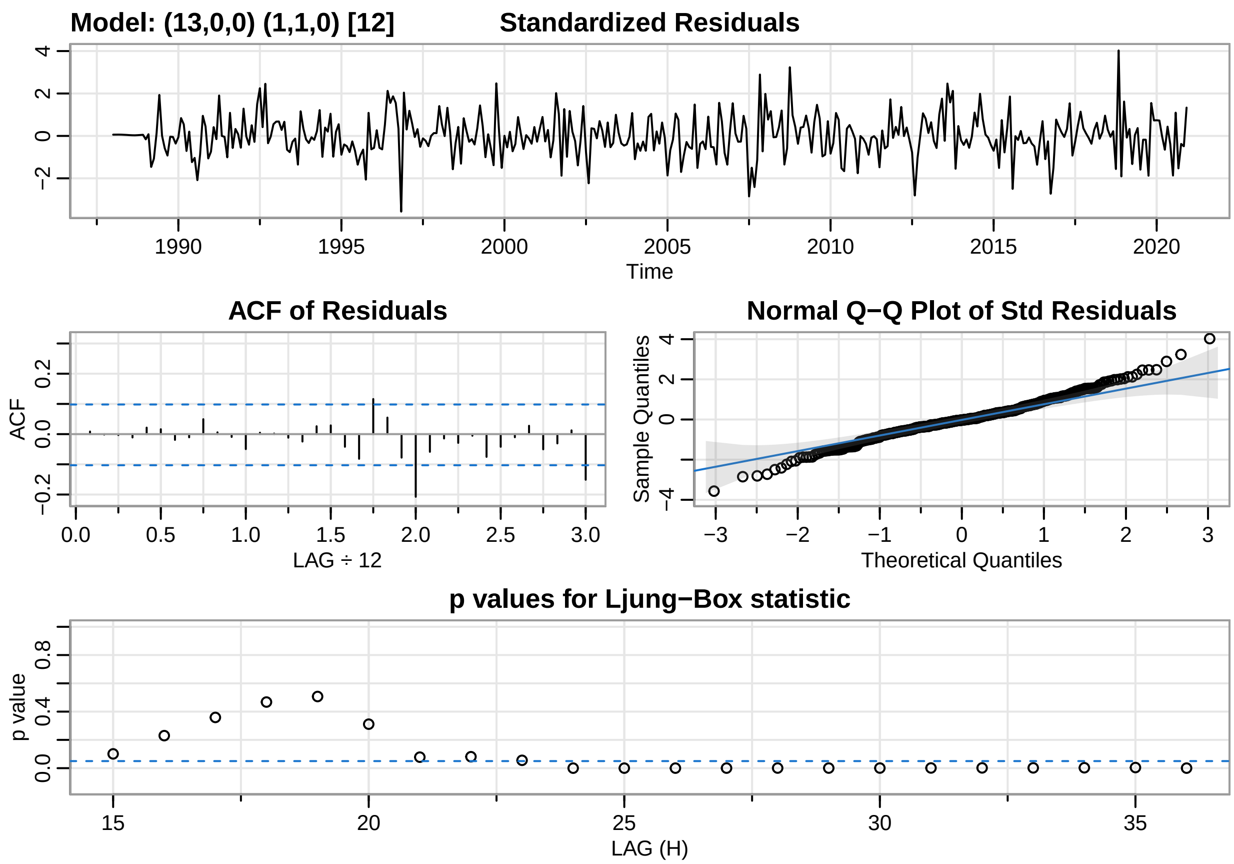

```
shapiro.test(model_26_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_26_train_residuals
## W = 0.98804, p-value = 0.002459
```

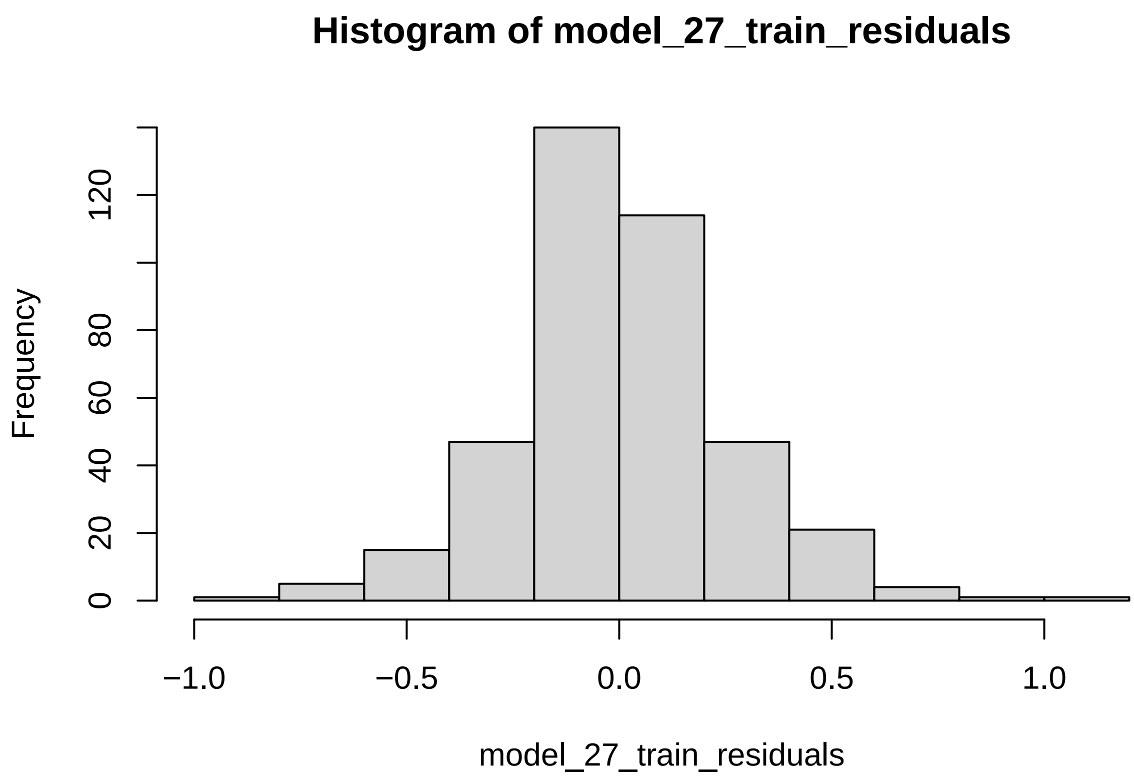
```
#SARIMA(13,0,0)x(1,1,0)_12
model_27_train <- sarima(Avg_ExtentTS_Train, p=13, d=0, q=0, P=1, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -0.801164
## iter   2 value -1.017847
## iter   3 value -1.143844
## iter   4 value -1.232274
## iter   5 value -1.284447
## iter   6 value -1.311521
## iter   7 value -1.320540
## iter   8 value -1.328806
## iter   9 value -1.334262
## iter  10 value -1.338005
## iter  11 value -1.339163
## iter  12 value -1.339774
## iter  13 value -1.340389
## iter  14 value -1.340593
## iter  15 value -1.340695
## iter  16 value -1.340719
## iter  17 value -1.340730
## iter  18 value -1.340734
## iter  19 value -1.340734
```

```
## iter  20 value -1.340735
## iter  20 value -1.340735
## iter  20 value -1.340735
## final  value -1.340735
## converged
## initial  value -1.338080
## iter   2 value -1.338274
## iter   3 value -1.338681
## iter   4 value -1.338714
## iter   5 value -1.338795
## iter   6 value -1.338818
## iter   7 value -1.338842
## iter   8 value -1.338848
## iter   9 value -1.338850
## iter  10 value -1.338850
## iter  11 value -1.338850
## iter  11 value -1.338850
## iter  11 value -1.338850
## final  value -1.338850
## converged
```




```
model_27_train_residuals = resid(model_27_train$fit)
hist(model_27_train_residuals)
```



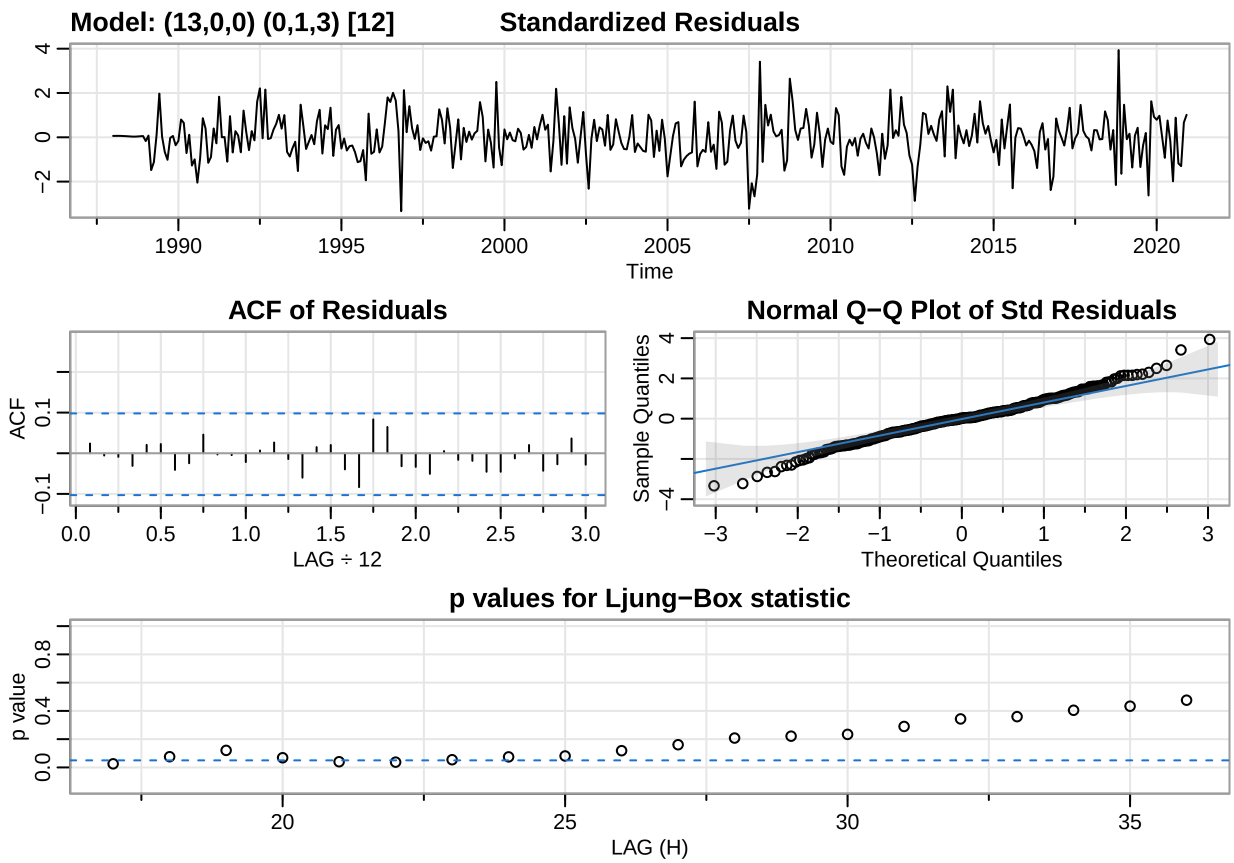

```
shapiro.test(model_27_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_27_train_residuals
## W = 0.98468, p-value = 0.0003358
```

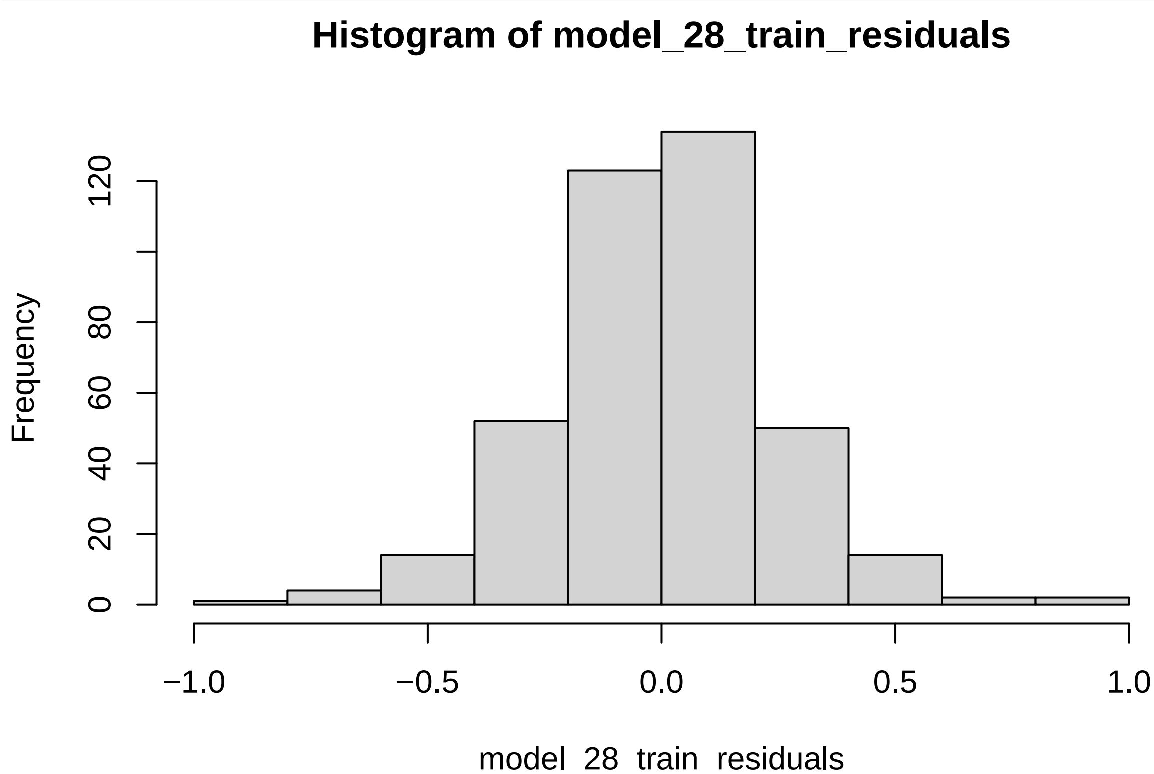
```
#SARIMA(13,0,0)x(0,1,3)_12
model_28_train <- sarima(Avg_ExtentTS_Train, p=13, d=0, q=0, P=0, D=1, Q=3, S=12 , details = TRUE)
```

```
## initial  value -0.794710
## iter   2 value -1.013936
## iter   3 value -1.173142
## iter   4 value -1.265645
## iter   5 value -1.324556
## iter   6 value -1.329399
## iter   7 value -1.354647
## iter   8 value -1.368673
## iter   9 value -1.381611
## iter  10 value -1.389125
## iter  11 value -1.391039
## iter  12 value -1.391755
## iter  13 value -1.392861
## iter  14 value -1.394528
## iter  15 value -1.395490
## iter  16 value -1.396737
## iter  17 value -1.396990
## iter  18 value -1.397072
## iter  19 value -1.397085
```

```
## iter  20 value -1.397089
## iter  21 value -1.397089
## iter  22 value -1.397089
## iter  22 value -1.397089
## iter  22 value -1.397089
## final  value -1.397089
## converged
## initial  value -1.392610
## iter   2 value -1.392693
## iter   3 value -1.392839
## iter   4 value -1.392864
## iter   5 value -1.392891
## iter   6 value -1.392893
## iter   7 value -1.392897
## iter   8 value -1.392898
## iter   9 value -1.392903
## iter  10 value -1.392904
## iter  11 value -1.392908
## iter  12 value -1.392910
## iter  13 value -1.392911
## iter  14 value -1.392912
## iter  15 value -1.392912
## iter  16 value -1.392912
## iter  16 value -1.392912
## iter  16 value -1.392912
## final  value -1.392912
## converged
```




```
model_28_train_residuals = resid(model_28_train$fit)
hist(model_28_train_residuals)
```



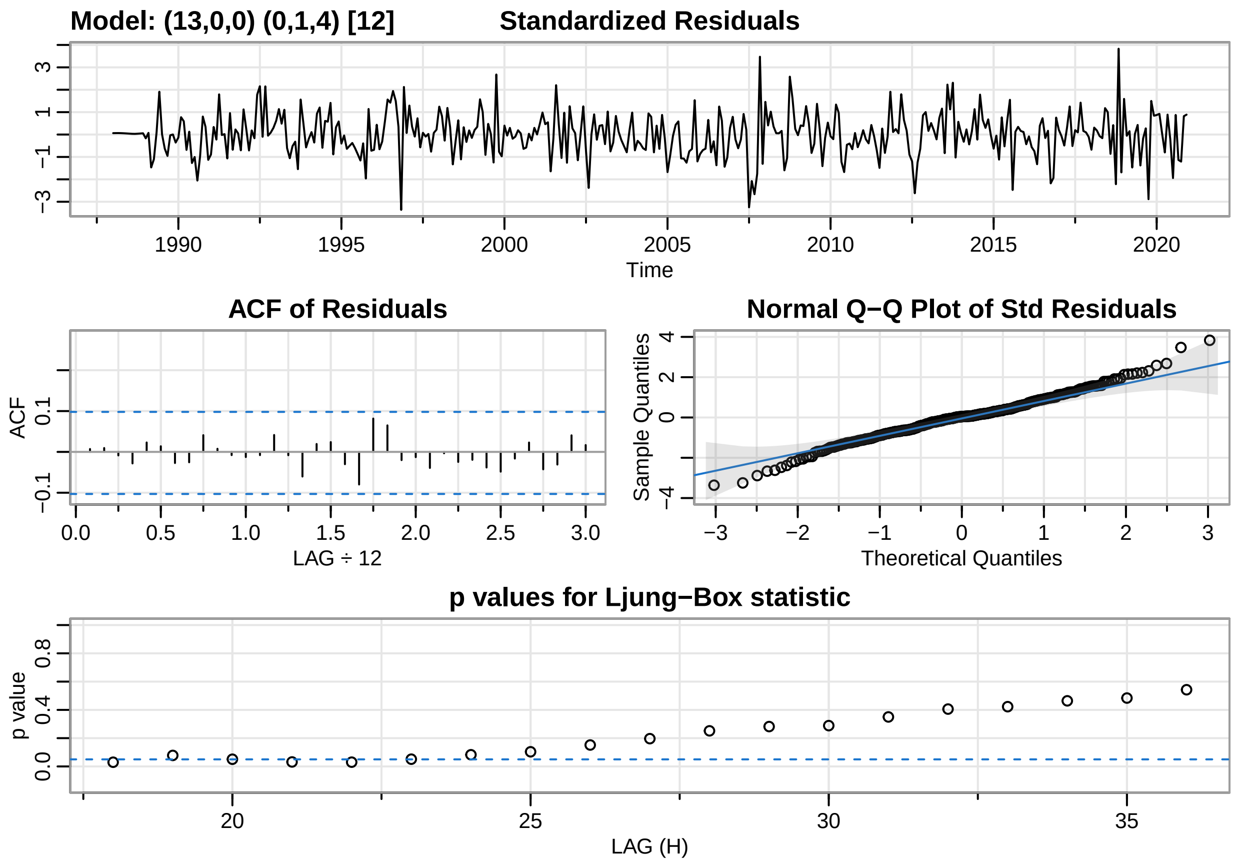
```r
shapiro.test(model_28_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_28_train_residuals
## W = 0.9863, p-value = 0.0008591
```

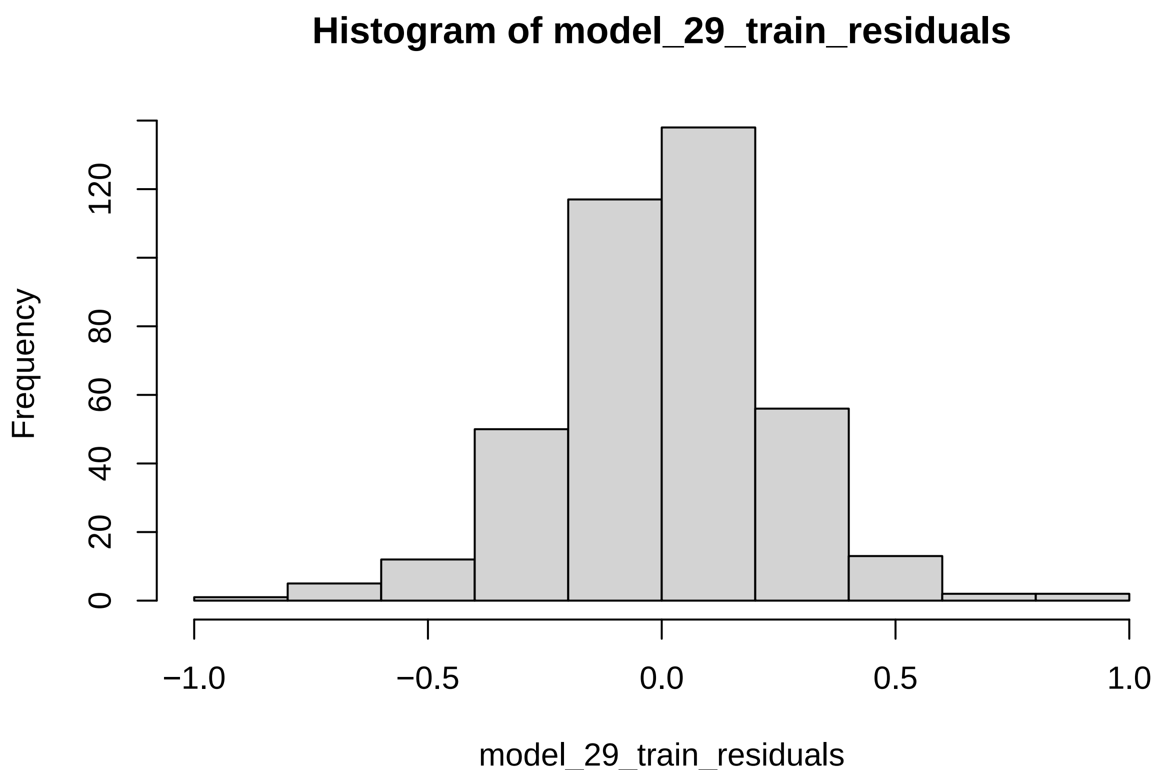
```r
#SARIMA(13,0,0)x(0,1,4)_12
model_29_train <- sarima(Avg_ExtentTS_Train, p=13, d=0, q=0, P=0, D=1, Q=4, S=12 , details = TRUE)
```

```
## initial  value -0.794710
## iter   2 value -1.018370
## iter   3 value -1.185015
## iter   4 value -1.269162
## iter   5 value -1.338386
## iter   6 value -1.361375
## iter   7 value -1.373123
## iter   8 value -1.389868
## iter   9 value -1.390417
## iter  10 value -1.395151
## iter  11 value -1.396699
## iter  12 value -1.398587
## iter  13 value -1.399541
## iter  14 value -1.401027
## iter  15 value -1.401975
## iter  16 value -1.402455
## iter  17 value -1.402620
## iter  18 value -1.402660
## iter  19 value -1.402671
## iter  20 value -1.402672
## iter  21 value -1.402673
## iter  22 value -1.402673
## iter  22 value -1.402673
## iter  22 value -1.402673
## final  value -1.402673
## converged
## initial  value -1.397598
## iter   2 value -1.397676
## iter   3 value -1.397908
## iter   4 value -1.397918
## iter   5 value -1.397948
## iter   6 value -1.397953
## iter   7 value -1.397964
## iter   8 value -1.397968
## iter   9 value -1.397974
## iter  10 value -1.397981
## iter  11 value -1.397982
## iter  12 value -1.397982
## iter  13 value -1.397983
## iter  14 value -1.397983
## iter  14 value -1.397983
## iter  14 value -1.397983
## final  value -1.397983
```

```
## converged
```




```
model_29_train_residuals = resid(model_29_train$fit)
hist(model_29_train_residuals)
```



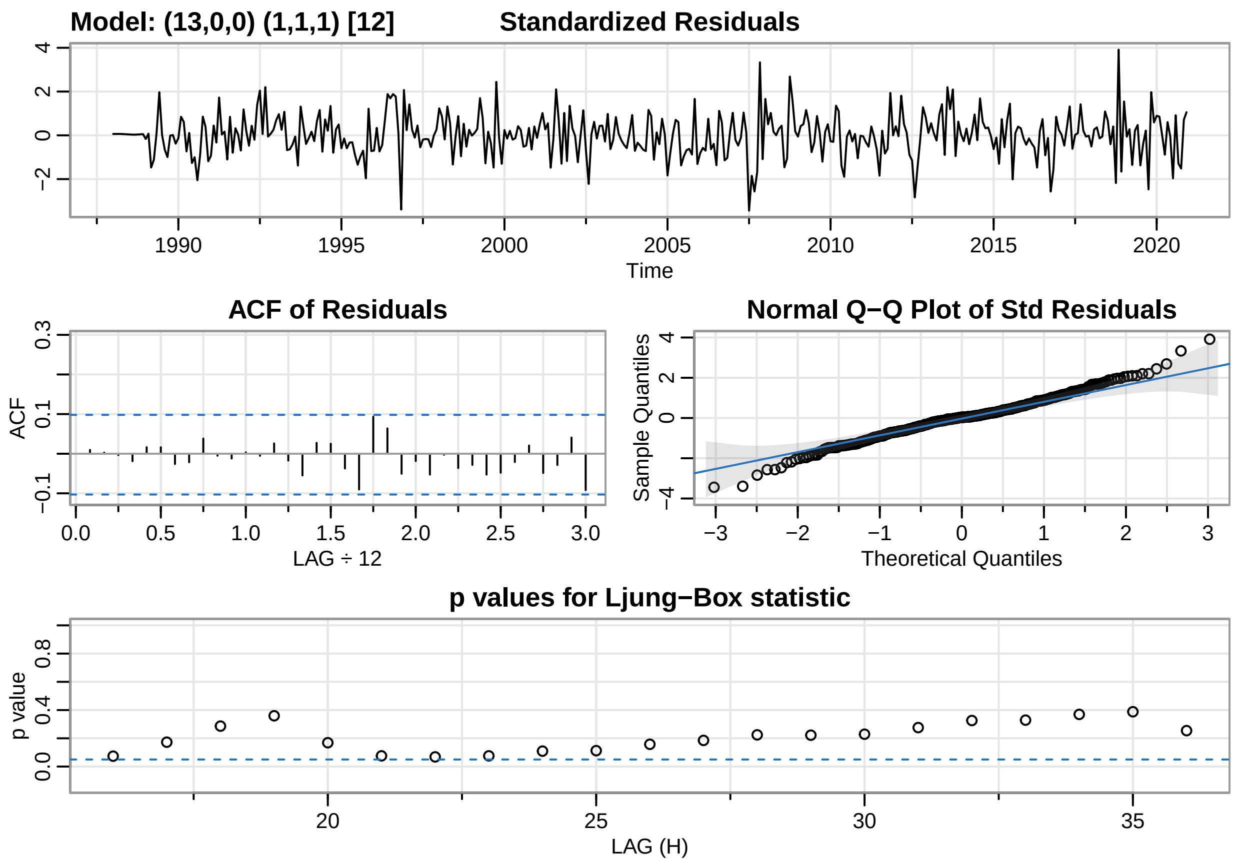

```
shapiro.test(model_29_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_29_train_residuals
## W = 0.98626, p-value = 0.0008373
```

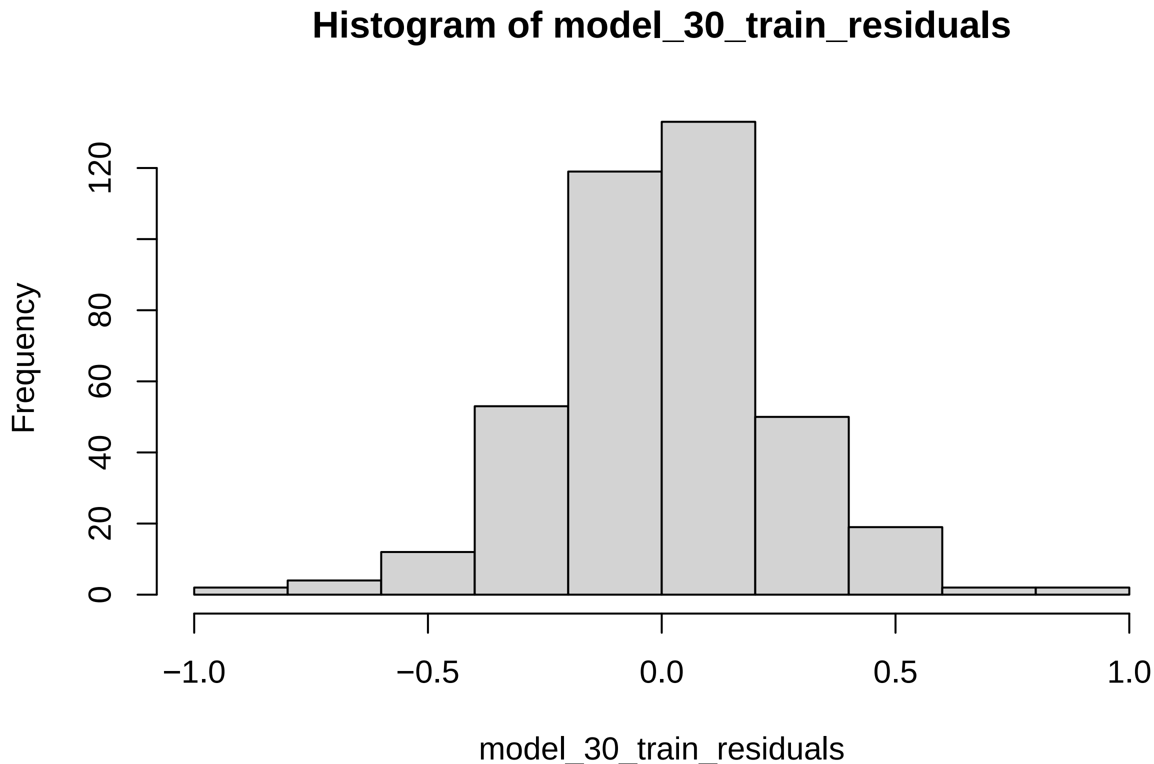
```
#SARIMA(13,0,0)x(1,1,1)_12
model_30_train <- sarima(Avg_ExtentTS_Train, p=13, d=0, q=0, P=1, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -0.801164
## iter   2 value -1.035924
## iter   3 value -1.163984
## iter   4 value -1.273600
## iter   5 value -1.313108
## iter   6 value -1.345098
## iter   7 value -1.350268
## iter   8 value -1.361690
## iter   9 value -1.364074
## iter  10 value -1.364704
## iter  11 value -1.365211
## iter  12 value -1.365454
## iter  13 value -1.365788
## iter  14 value -1.366112
## iter  15 value -1.366710
## iter  16 value -1.367095
## iter  17 value -1.367549
## iter  18 value -1.367600
## iter  19 value -1.367614
```

```
## iter  20 value -1.367619
## iter  21 value -1.367620
## iter  22 value -1.367621
## iter  23 value -1.367621
## iter  23 value -1.367621
## iter  23 value -1.367621
## final  value -1.367621
## converged
## initial  value -1.378413
## iter   2 value -1.381305
## iter   3 value -1.381612
## iter   4 value -1.383730
## iter   5 value -1.384338
## iter   6 value -1.385117
## iter   7 value -1.385893
## iter   8 value -1.386815
## iter   9 value -1.387004
## iter  10 value -1.387072
## iter  11 value -1.387079
## iter  12 value -1.387081
## iter  13 value -1.387081
## iter  14 value -1.387081
## iter  14 value -1.387081
## iter  14 value -1.387081
## final  value -1.387081
## converged
```



```r
model_30_train_residuals = resid(model_30_train$fit)
hist(model_30_train_residuals)
```



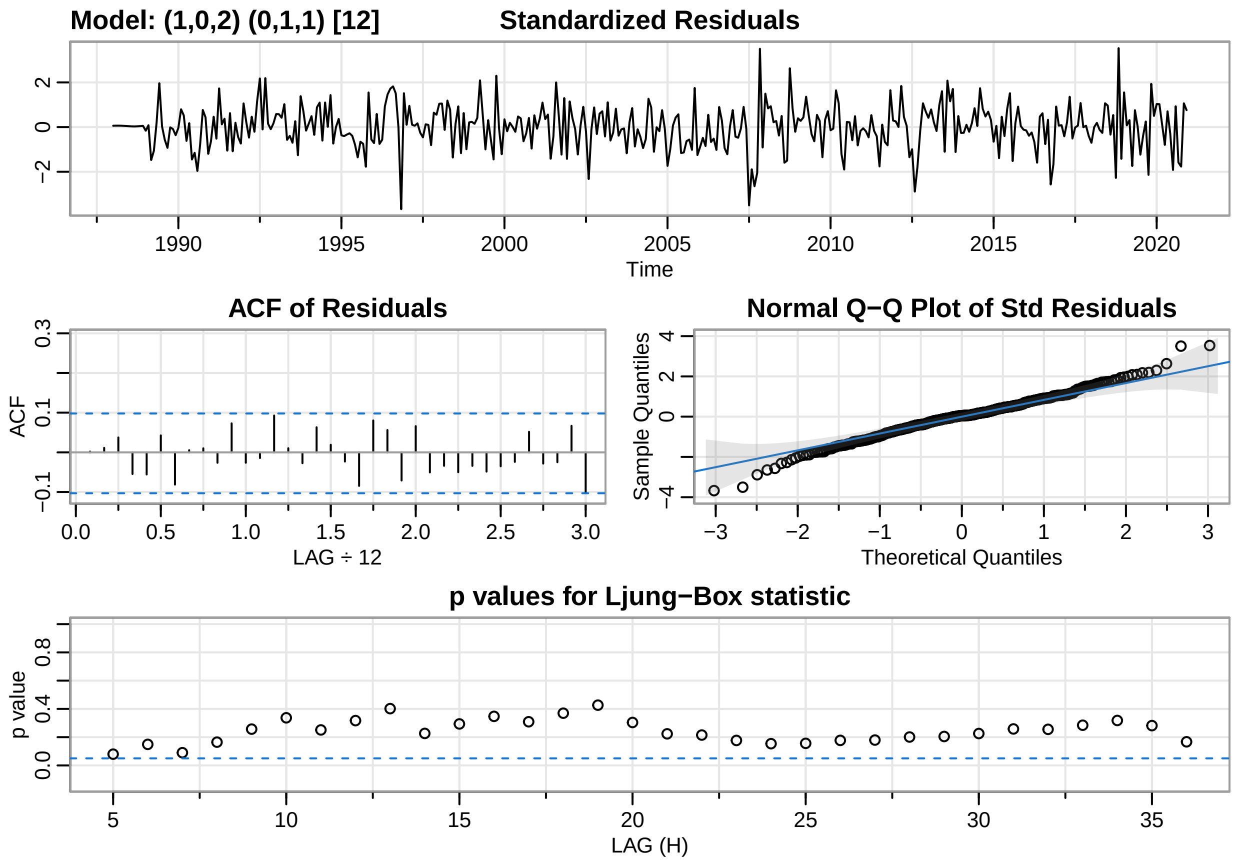

```r
shapiro.test(model_30_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_30_train_residuals
## W = 0.98623, p-value = 0.00082
```

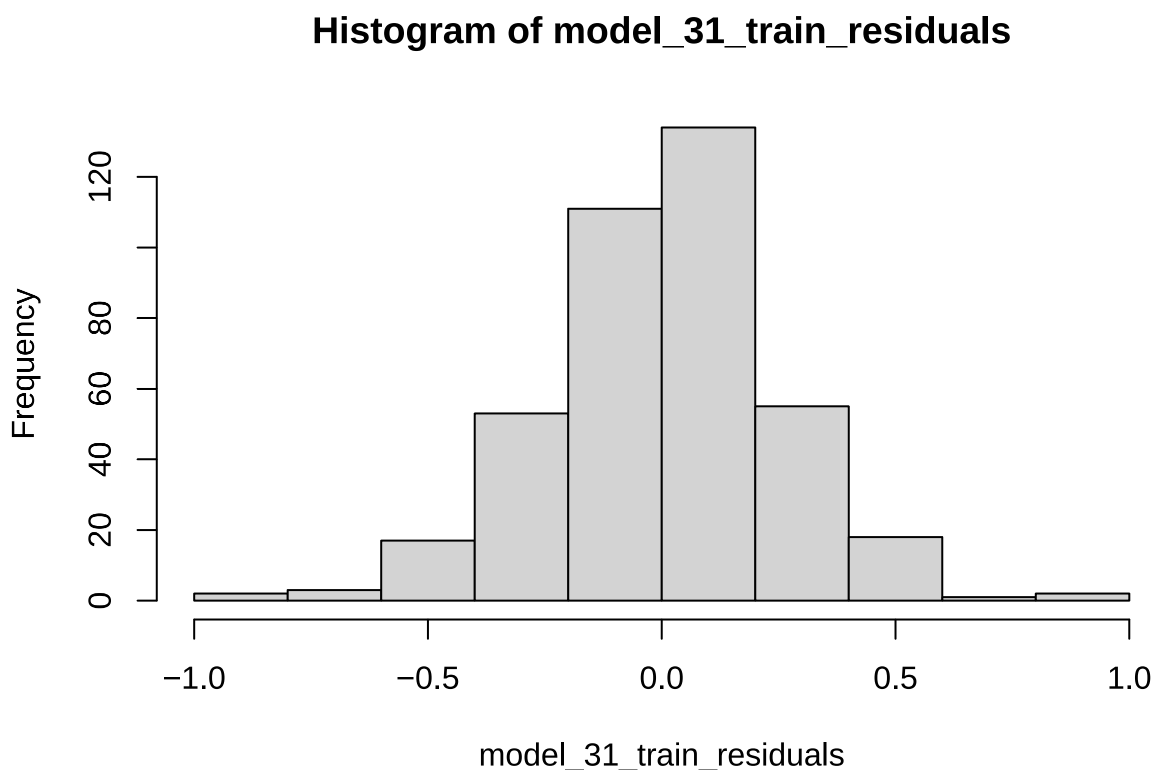
```r
#SARIMA(1,0,2)x(0,1,1)_12
model_31_train <- sarima(Avg_ExtentTS_Train, p=1, d=0, q=2, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -0.801086
## iter   2 value -1.138952
## iter   3 value -1.342237
## iter   4 value -1.355399
## iter   5 value -1.360693
## iter   6 value -1.362722
## iter   7 value -1.364221
## iter   8 value -1.365966
## iter   9 value -1.367305
## iter  10 value -1.367779
## iter  11 value -1.367834
## iter  12 value -1.367835
## iter  12 value -1.367835
## final  value -1.367835
## converged
## initial  value -1.365277
```

```
## iter   2 value -1.365339
## iter   3 value -1.365342
## iter   4 value -1.365342
## iter   5 value -1.365343
## iter   6 value -1.365344
## iter   7 value -1.365344
## iter   7 value -1.365344
## iter   7 value -1.365344
## final  value -1.365344
## converged
```




```
model_31_train_residuals = resid(model_31_train$fit)
hist(model_31_train_residuals)
```



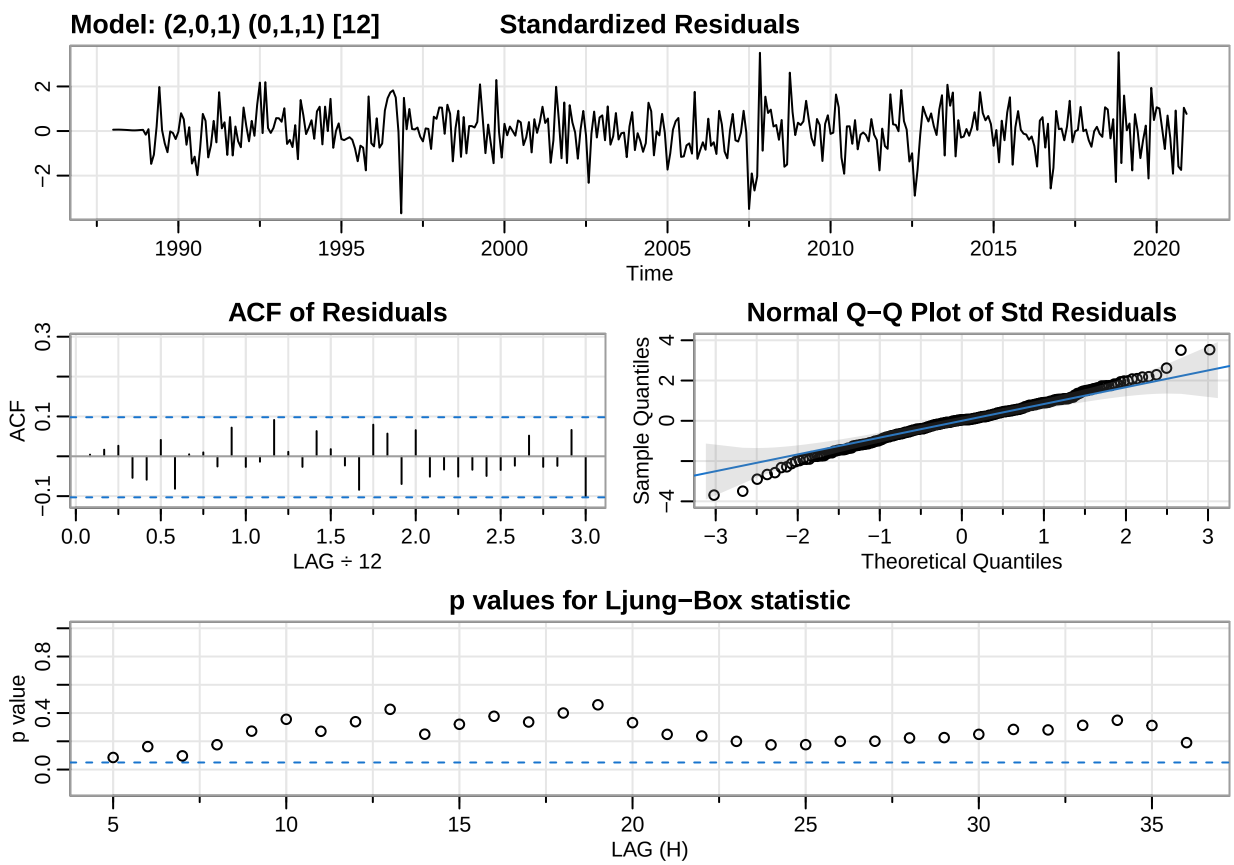

```
shapiro.test(model_31_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_31_train_residuals
## W = 0.98783, p-value = 0.002153
```

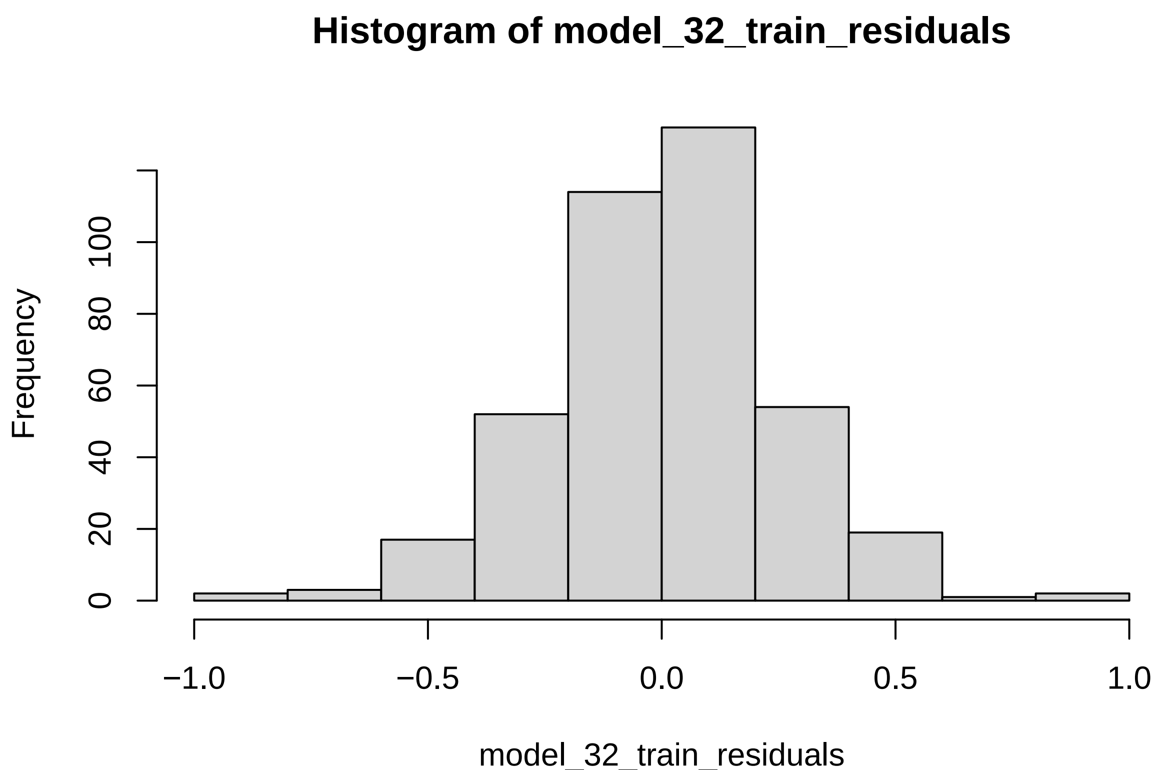
```
#SARIMA(2,0,1)x(0,1,1)_12
model_32_train <- sarima(Avg_ExtentTS_Train, p=2, d=0, q=1, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -0.799783
## iter   2 value -1.147624
## iter   3 value -1.332335
## iter   4 value -1.355514
## iter   5 value -1.366626
## iter   6 value -1.366675
## iter   7 value -1.366841
## iter   8 value -1.366857
## iter   9 value -1.366899
## iter  10 value -1.366998
## iter  11 value -1.367081
## iter  12 value -1.367109
## iter  13 value -1.367111
## iter  14 value -1.367112
## iter  14 value -1.367112
## iter  14 value -1.367112
## final  value -1.367112
## converged
## initial  value -1.365359
```

```
## iter   2 value -1.365381
## iter   3 value -1.365409
## iter   4 value -1.365409
## iter   5 value -1.365410
## iter   6 value -1.365412
## iter   7 value -1.365416
## iter   8 value -1.365420
## iter   9 value -1.365421
## iter  10 value -1.365421
## iter  10 value -1.365421
## iter  10 value -1.365421
## final  value -1.365421
## converged
```




```
model_32_train_residuals = resid(model_32_train$fit)
hist(model_32_train_residuals)
```



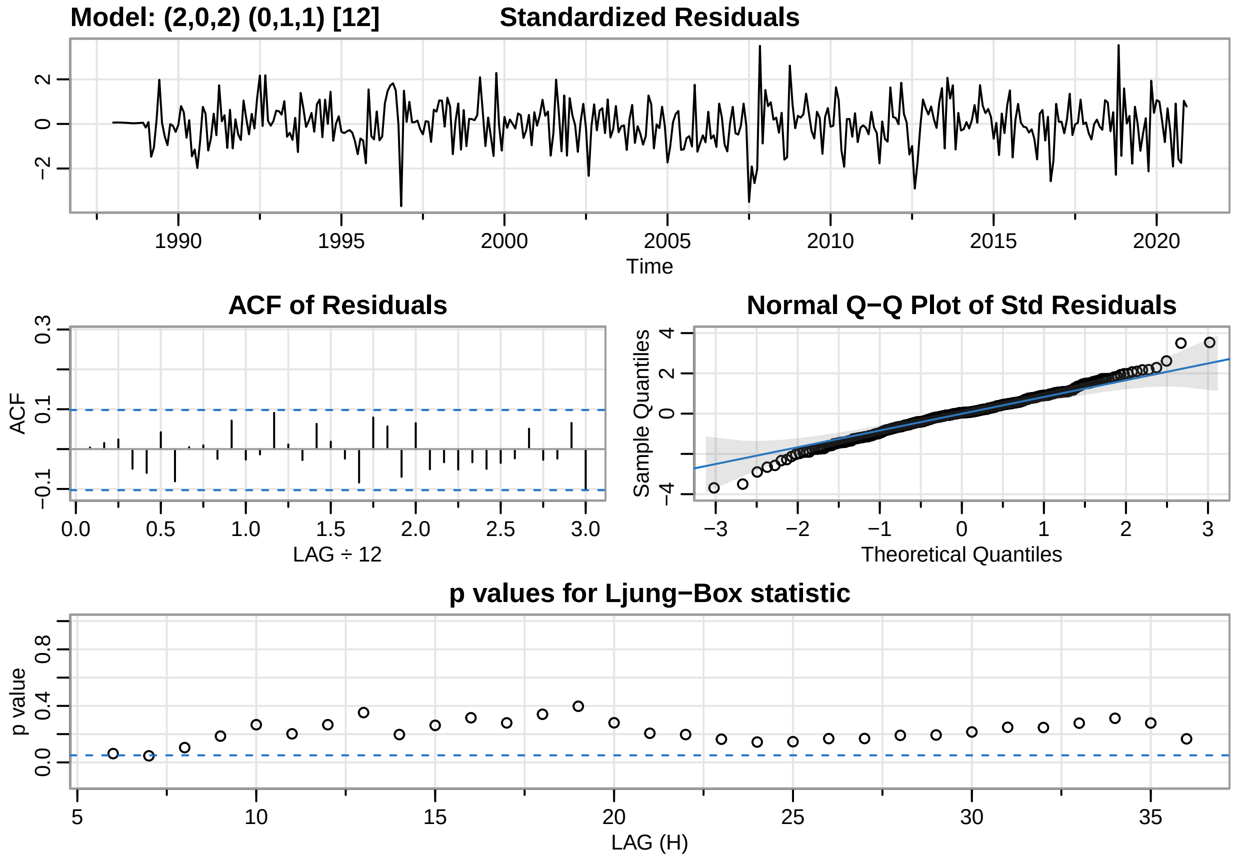

```
shapiro.test(model_32_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_32_train_residuals
## W = 0.98755, p-value = 0.001822
```

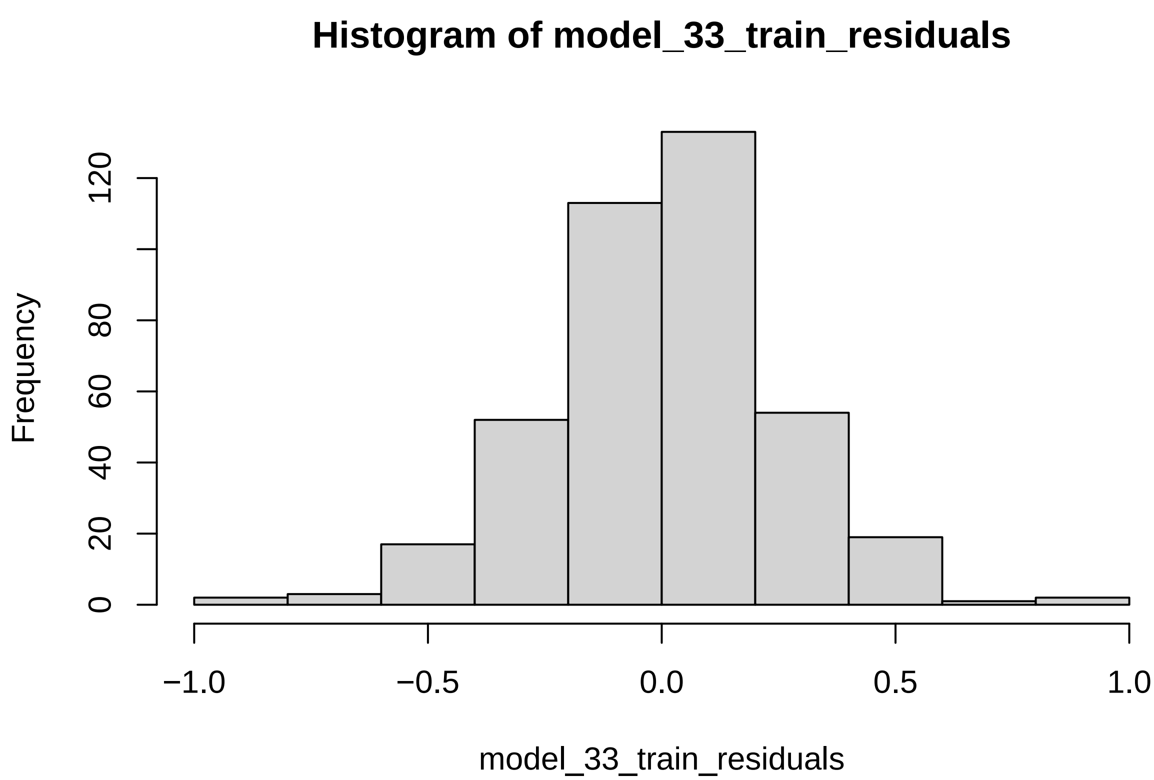
```
#SARIMA(2,0,2)x(0,1,1)_12
model_33_train <- sarima(Avg_ExtentTS_Train, p=2, d=0, q=2, P=0, D=1, Q=1, S=12 , details = TRUE)
```

```
## initial  value -0.799783
## iter   2 value -1.066061
## iter   3 value -1.228003
## iter   4 value -1.328895
## iter   5 value -1.356060
## iter   6 value -1.366108
## iter   7 value -1.366516
## iter   8 value -1.367171
## iter   9 value -1.367298
## iter  10 value -1.367313
## iter  11 value -1.367313
## iter  11 value -1.367313
## iter  11 value -1.367313
## final  value -1.367313
## converged
## initial  value -1.365536
## iter   2 value -1.365571
## iter   3 value -1.365588
## iter   4 value -1.365590
```

```
## iter   5 value -1.365591
## iter   6 value -1.365591
## iter   6 value -1.365591
## iter   6 value -1.365591
## final  value -1.365591
## converged
```




```
model_33_train_residuals = resid(model_33_train$fit)
hist(model_33_train_residuals)
```



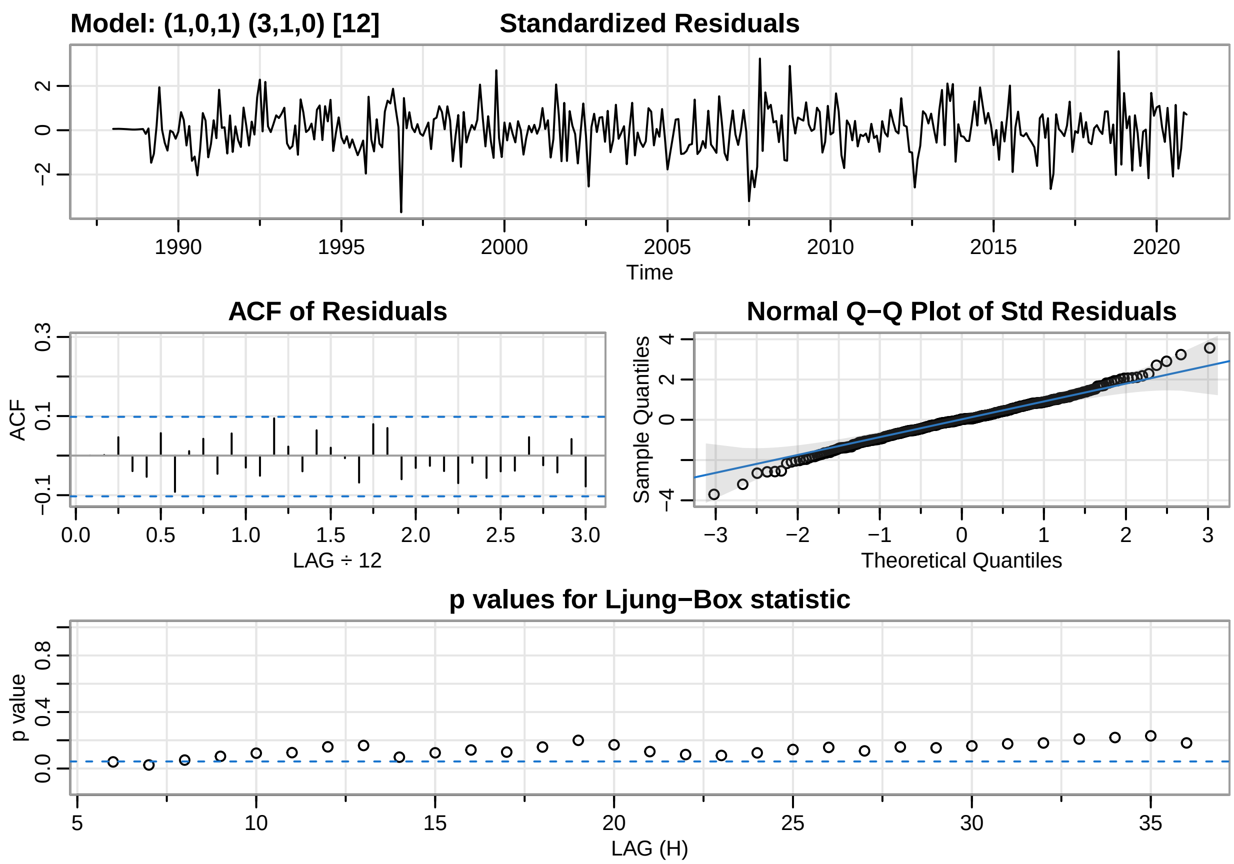

```
shapiro.test(model_33_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_33_train_residuals
## W = 0.98761, p-value = 0.001885
```

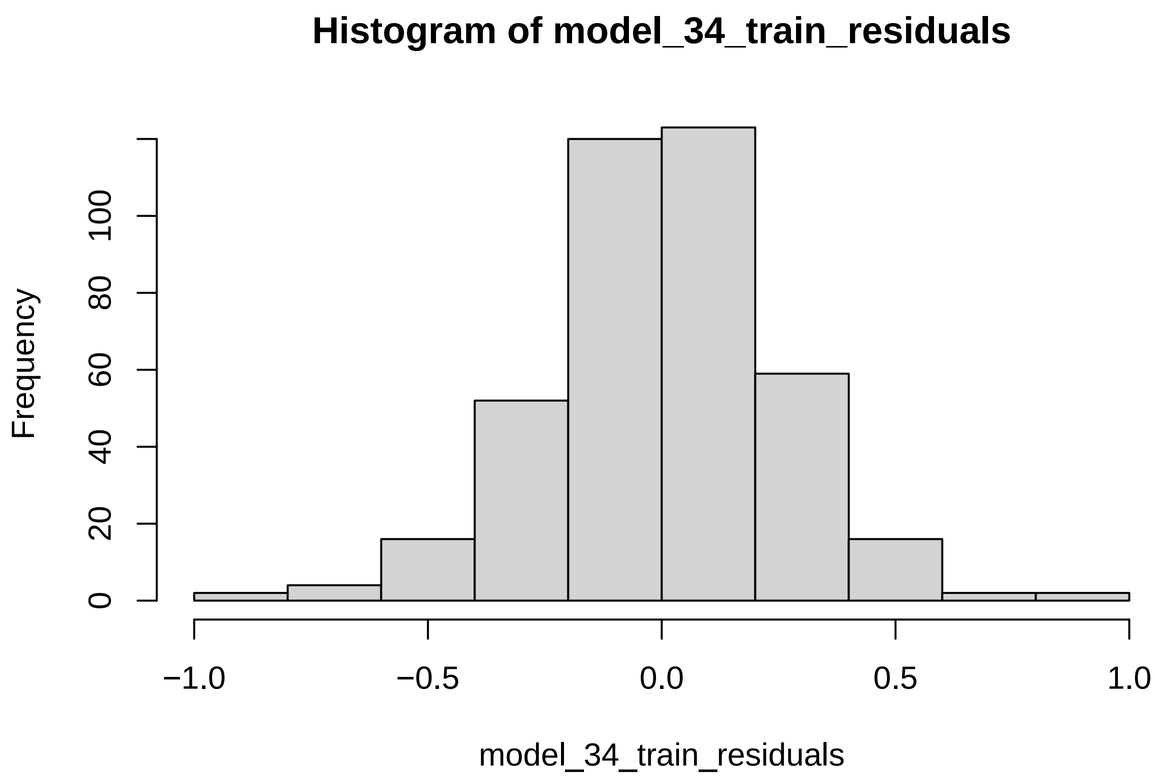
```
# Weird combo of optimal parameters for d=0, and d=1
#SARIMA(1,0,1)x(3,1,0)_12
model_34_train <- sarima(Avg_ExtentTS_Train, p=1, d=0, q=1, P=3, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -0.795069
## iter   2 value -1.107339
## iter   3 value -1.327476
## iter   4 value -1.351082
## iter   5 value -1.353717
## iter   6 value -1.360317
## iter   7 value -1.360498
## iter   8 value -1.360526
## iter   9 value -1.360547
## iter  10 value -1.360548
## iter  11 value -1.360548
## iter  11 value -1.360548
## iter  11 value -1.360548
## final  value -1.360548
## converged
## initial  value -1.361106
## iter   2 value -1.361247
## iter   3 value -1.361301
```

```
## iter   4 value -1.361318
## iter   5 value -1.361322
## iter   6 value -1.361323
## iter   7 value -1.361324
## iter   8 value -1.361324
## iter   8 value -1.361324
## iter   8 value -1.361324
## final  value -1.361324
## converged
```




```
model_34_train_residuals = resid(model_34_train$fit)
hist(model_34_train_residuals)
```



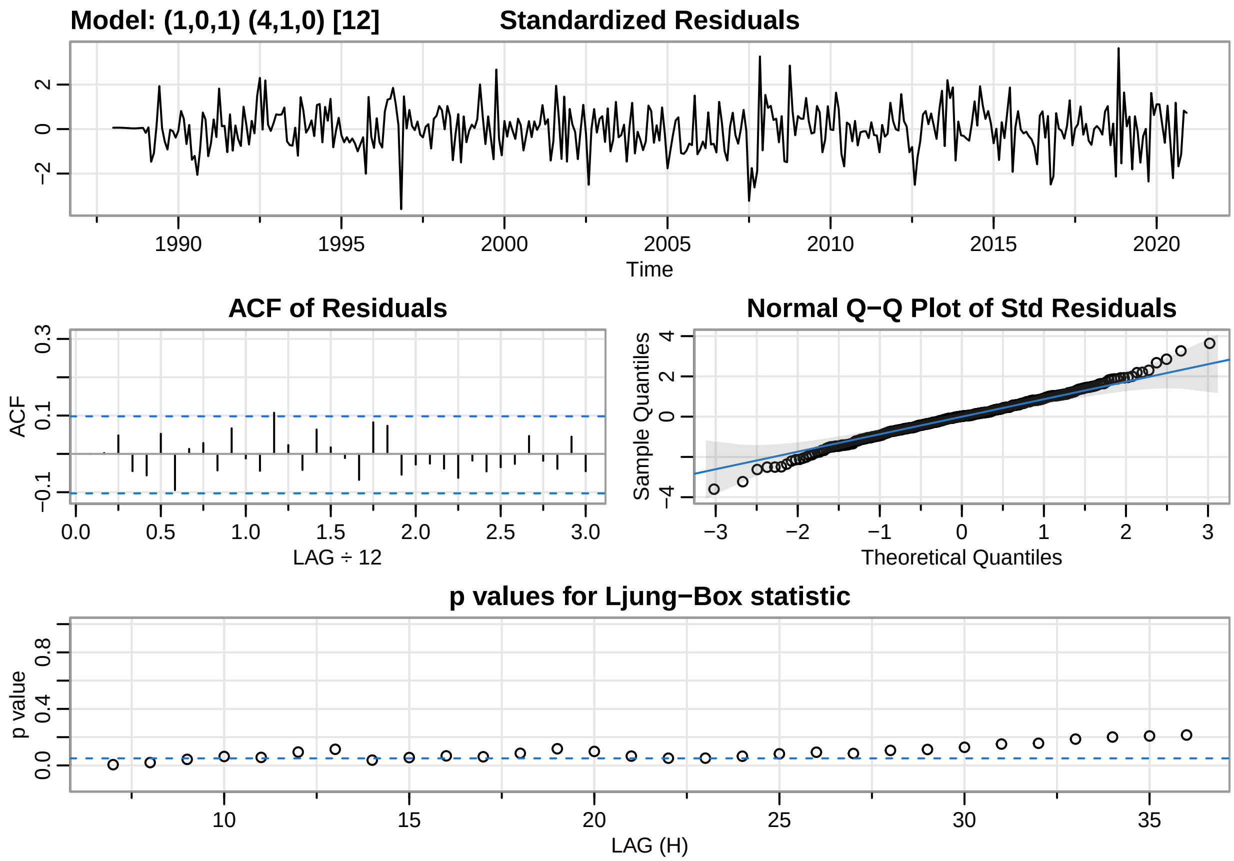

```
shapiro.test(model_34_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_34_train_residuals
## W = 0.99035, p-value = 0.01064
```

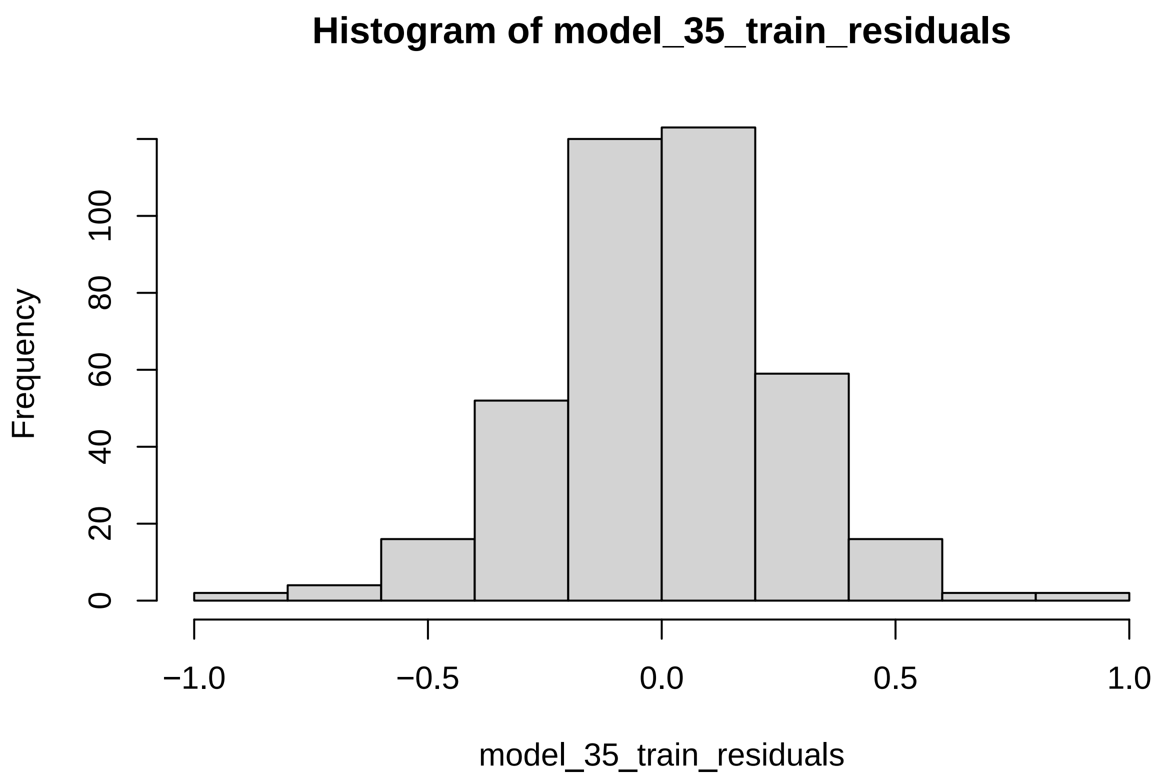
```
# Weird combo of optimal parameters for d=0, and d=1
#SARIMA(1,0,1)x(3,1,0)_12
model_35_train <- sarima(Avg_ExtentTS_Train, p=1, d=0, q=1, P=4, D=1, Q=0, S=12 , details = TRUE)
```

```
## initial  value -0.804892
## iter   2 value -1.078475
## iter   3 value -1.326207
## iter   4 value -1.348467
## iter   5 value -1.357442
## iter   6 value -1.366662
## iter   7 value -1.367642
## iter   8 value -1.367689
## iter   9 value -1.367707
## iter  10 value -1.367713
## iter  11 value -1.367715
## iter  12 value -1.367715
## iter  12 value -1.367715
## iter  12 value -1.367715
## final  value -1.367715
## converged
## initial  value -1.365081
## iter   2 value -1.365423
```

```
## iter   3 value -1.365564
## iter   4 value -1.365571
## iter   5 value -1.365575
## iter   6 value -1.365577
## iter   7 value -1.365577
## iter   8 value -1.365577
## iter   8 value -1.365577
## iter   8 value -1.365577
## final  value -1.365577
## converged
```




```
model_35_train_residuals = resid(model_34_train$fit)
hist(model_35_train_residuals)
```




```
shapiro.test(model_35_train_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_35_train_residuals
## W = 0.99035, p-value = 0.01064
```

#TODO try 'SARIMA(1,0,3)x(0,1,1)_12', 'SARIMA(3,0,1)x(0,1,1)_12', 'SARIMA(2,0,3)x(0,1,1)_12', 'SARIMA(3,0,2)x(0,1,1)_12', 'SARIMA(3,0,3)x(0,1,1)_12'

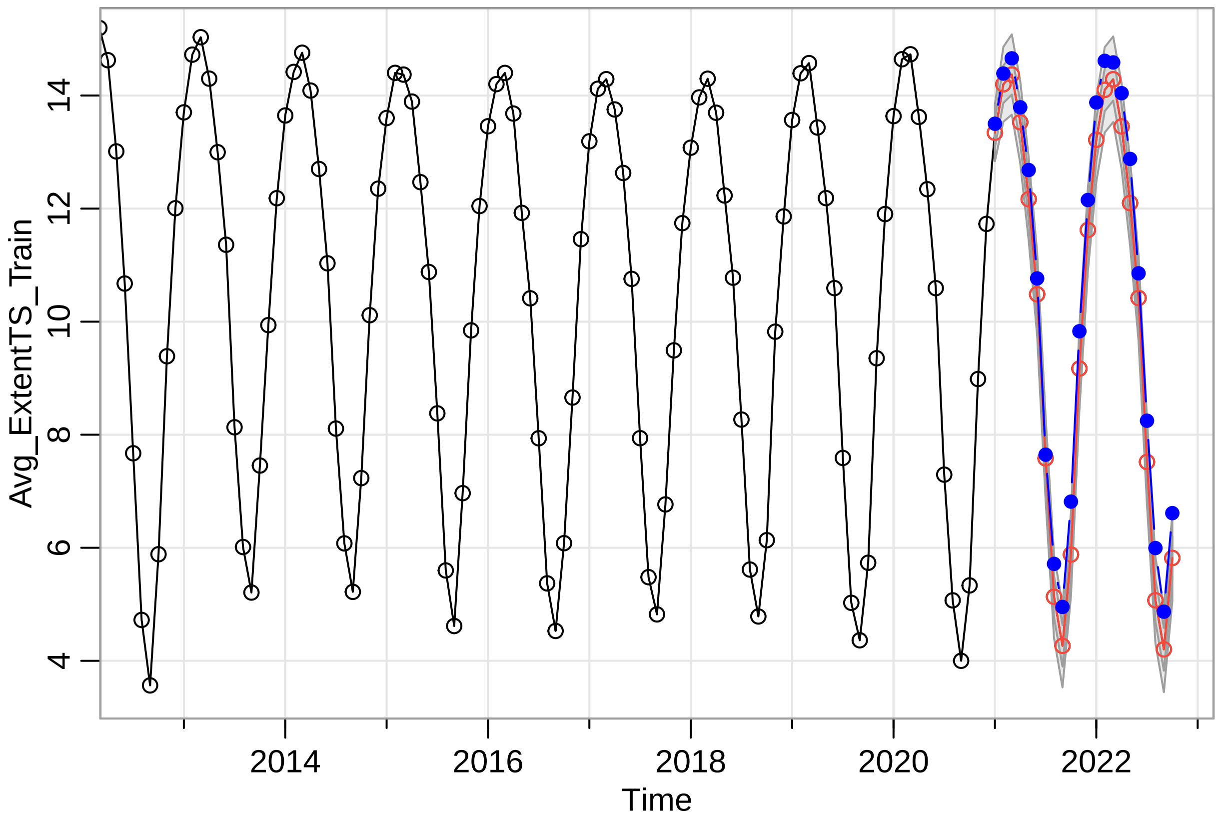
```
library(huxtable)
goodness_of_fit <- hux(
        Model = c('SARIMA(1,0,1)x(0,1,1)_12', 'SARIMA(1,0,1)x(1,1,0)_12', 'SARIMA(1,0,1)x(0,1,3)_12', 'S
                  'SARIMA(13,0,0)x(0,1,1)_12', 'SARIMA(13,0,0)x(1,1,0)_12', 'SARIMA(13,0,0)x(0,1,3)_12'
                  'SARIMA(1,0,2)x(0,1,1)_12', 'SARIMA(2,0,1)x(0,1,1)_12', 'SARIMA(2,0,2)x(0,1,1)_12', 'S
        AIC = c(model_21_train$AIC, model_22_train$AIC, model_23_train$AIC, model_24_train$AIC, model_25
                model_26_train$AIC, model_27_train$AIC, model_28_train$AIC, model_29_train$AIC, model_30
                model_31_train$AIC, model_32_train$AIC, model_33_train$AIC, model_34_train$AIC, model_35
        AICc = c(model_21_train$AICc, model_22_train$AICc, model_23_train$AICc, model_24_train$AICc, mod
                model_26_train$AICc, model_27_train$AICc, model_28_train$AICc, model_29_train$AICc, mode
                model_31_train$AICc, model_32_train$AICc, model_33_train$AICc, model_34_train$AICc, mode
        BIC = c(model_21_train$BIC, model_22_train$BIC, model_23_train$BIC, model_24_train$BIC, model_25
                model_26_train$BIC, model_27_train$BIC, model_28_train$BIC, model_29_train$BIC, model_30
                model_31_train$BIC, model_32_train$BIC, model_33_train$BIC, model_34_train$BIC, model_35
        MSE = c(mean(model_21_train_residuals^2), mean(model_22_train_residuals^2), mean(model_23_train_
                mean(model_26_train_residuals^2), mean(model_27_train_residuals^2), mean(model_28_train_
                mean(model_31_train_residuals^2), mean(model_32_train_residuals^2), mean(model_33_train_
      )

goodness_of_fit %>%
```

```
set_number_format(col=c(2,3,4,5), value=3) %>%
set_bottom_border(c(1,11,14), everywhere) %>%
set_bold(c(2,4,5,6,12,13), everywhere) %>%
set_background_color(c(2,4,5,6,12,13), everywhere, "grey95")
```

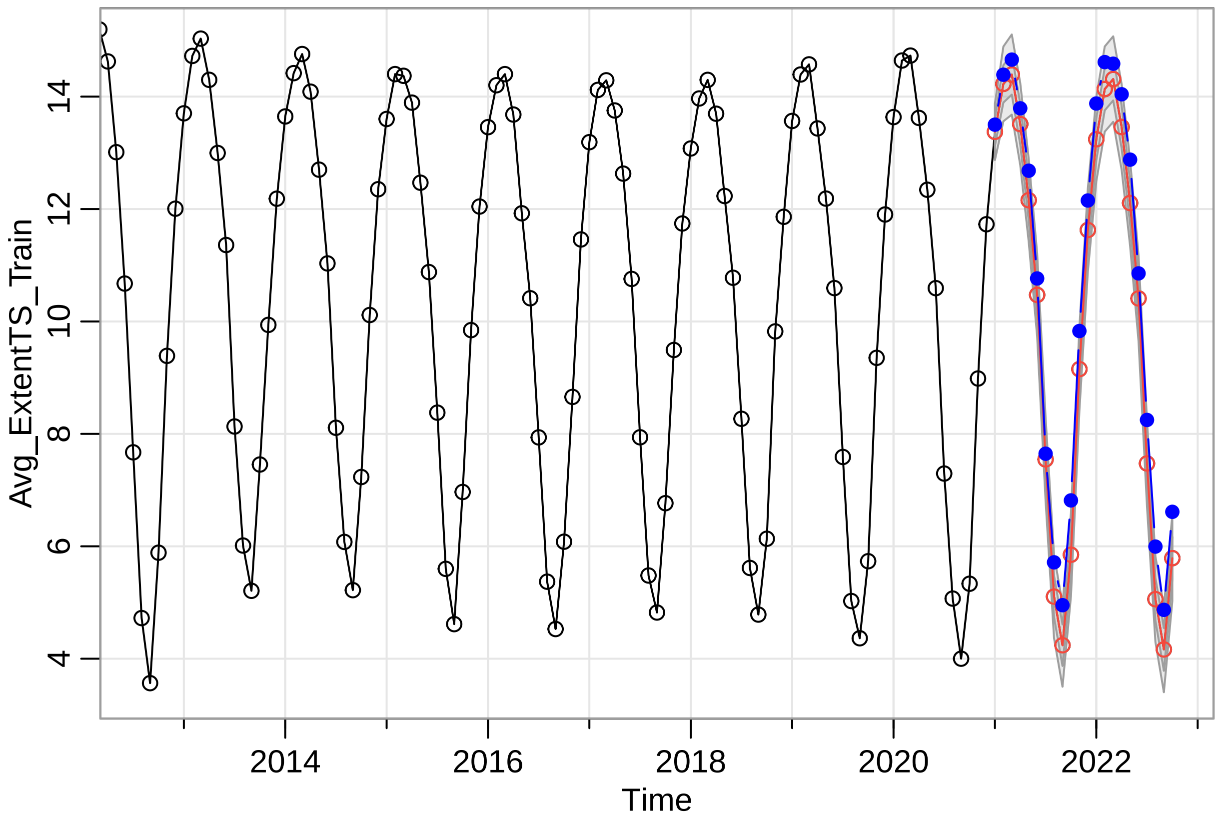
Model	AIC	AICc	BIC	MSE
SARIMA(1,0,1)x(0,1,1)__12	**0.133**	**0.134**	**0.185**	**0.061**
SARIMA(1,0,1)x(1,1,0)__12	0.250	0.250	0.301	0.070
SARIMA(1,0,1)x(0,1,3)__12	**0.137**	**0.137**	**0.209**	**0.061**
SARIMA(1,0,1)x(0,1,4)__12	**0.134**	**0.135**	**0.216**	**0.060**
SARIMA(1,0,1)x(1,1,1)__12	**0.138**	**0.138**	**0.200**	**0.061**
SARIMA(13,0,0)x(0,1,1)__12	0.163	0.166	0.327	0.060
SARIMA(13,0,0)x(1,1,0)__12	0.244	0.247	0.408	0.066
SARIMA(13,0,0)x(0,1,3)__12	0.146	0.150	0.331	0.058
SARIMA(13,0,0)x(0,1,4)__12	0.141	0.146	0.336	0.057
SARIMA(13,0,0)x(1,1,1)__12	0.152	0.156	0.327	0.059
SARIMA(1,0,2)x(0,1,1)__12	**0.138**	**0.139**	**0.200**	**0.061**
SARIMA(2,0,1)x(0,1,1)__12	**0.138**	**0.139**	**0.200**	**0.061**
SARIMA(2,0,2)x(0,1,1)__12	0.143	0.144	0.215	0.061
SARIMA(1,0,1)x(3,1,0)__12	0.152	0.152	0.224	0.062
SARIMA(1,0,1)x(4,1,0)__12	0.148	0.149	0.231	0.062

Model Selection

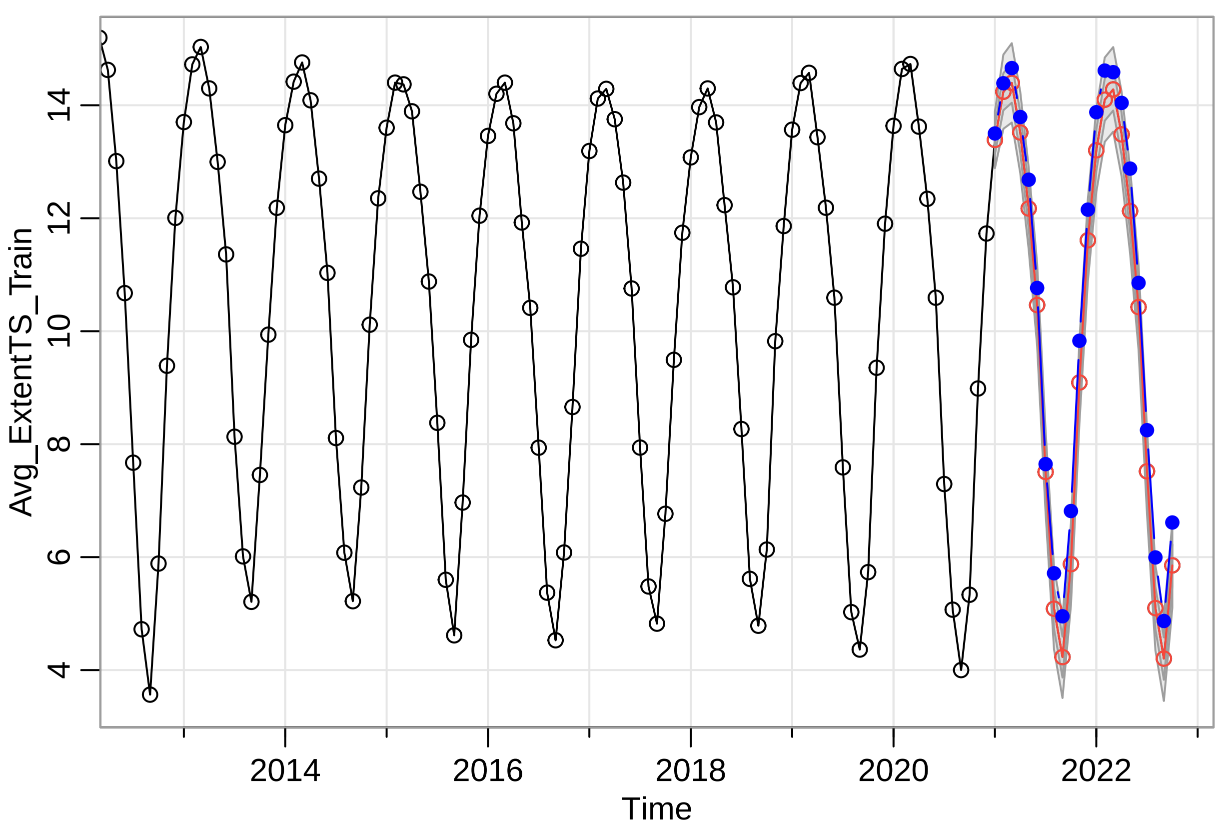
```
model_21_train_forecast<- sarima.for(Avg_ExtentTS_Train, n.ahead=22, p=1,d=0,q=1,P=0,D=1,Q=1,S=12)
lines(Avg_ExtentTS_Test,col='blue',type='b',pch=16)
```



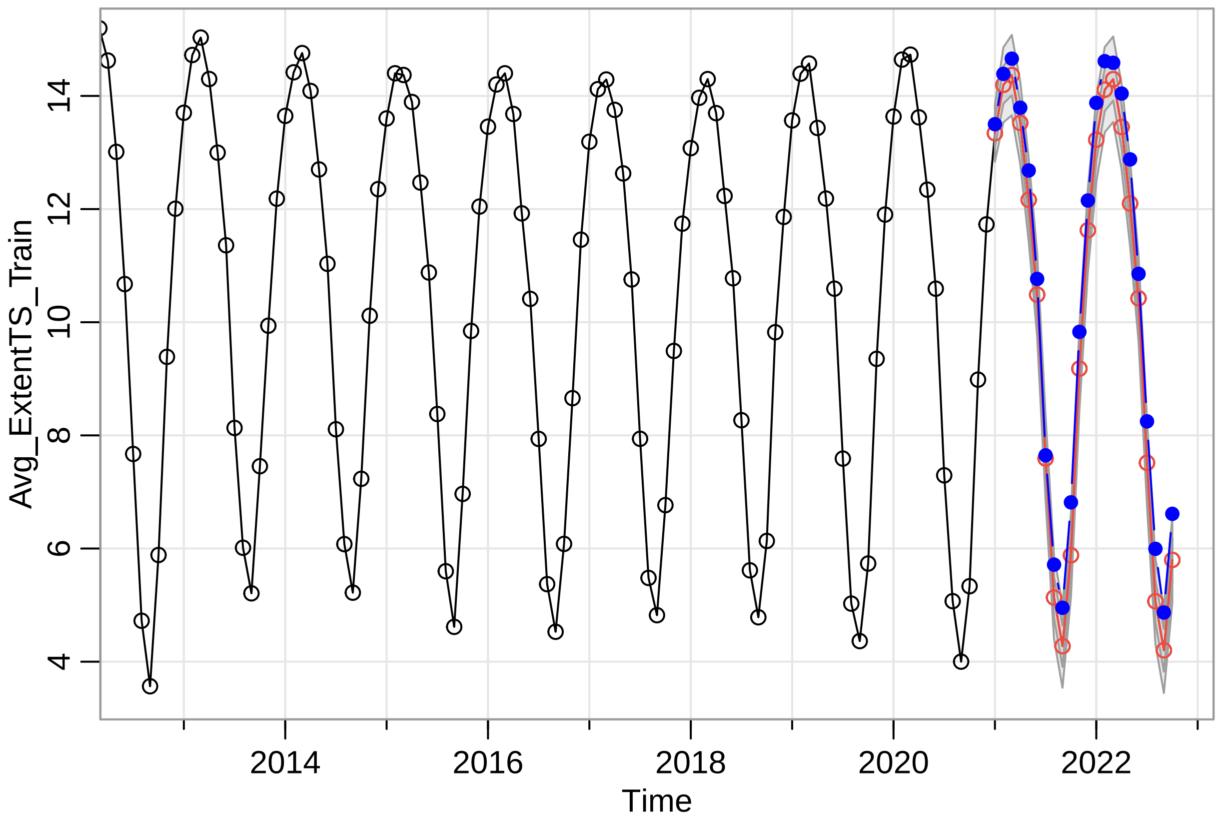

```
model_23_train_forecast<- sarima.for(Avg_ExtentTS_Train, n.ahead=22, p=1,d=0,q=1,P=0,D=1,Q=3,S=12)
lines(Avg_ExtentTS_Test,col='blue',type='b',pch=16)
```




```r
model_24_train_forecast<- sarima.for(Avg_ExtentTS_Train, n.ahead=22, p=1,d=0,q=1,P=0,D=1,Q=4,S=12)
lines(Avg_ExtentTS_Test,col='blue',type='b',pch=16)
```



```
model_25_train_forecast<- sarima.for(Avg_ExtentTS_Train, n.ahead=22, p=1,d=0,q=1,P=1,D=1,Q=1,S=12)
lines(Avg_ExtentTS_Test,col='blue',type='b',pch=16)
```




```
mean((model_21_train_forecast$pred-Avg_ExtentTS_Test)^2)
```

```
## [1] 0.3343123
```

```
mean((model_23_train_forecast$pred-Avg_ExtentTS_Test)^2)
```

```
## [1] 0.3455438
```

```
mean((model_24_train_forecast$pred-Avg_ExtentTS_Test)^2)
```

```
## [1] 0.3370083
```

```
mean((model_25_train_forecast$pred-Avg_ExtentTS_Test)^2)
```

```
## [1] 0.3340769
```