---

title: “Time Series Assignment”
author: “Anuja Saira Abraham”
date: “2024-09-09”
output: pdf_document

Data Collection

The “Electricity” dataset in the TSA library in R contains monthly electricity demand data spanning from the year 1973 to 2005. This dataset is commonly used in time series analysis and forecasting studies due to its extensive time span and relevance to real-world applications. This dataset is particularly popular because it provides a rich source of historical electricity demand data over a long time period.

The dataset’s significance is further underscored by its frequent mention in the book “Time Series Analysis With Applications in R” by Jonathan D. Cryer and Kung-Sik Chan. The authors likely chose this dataset for its suitability in illustrating various time series analysis techniques covered in the book. By working with this dataset, readers can gain practical insights into applying time series methods to real-world data and understand the challenges and opportunities involved in forecasting electricity demand.

```
library(TSA)
```

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

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

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

```
data("electricity")
```

```
head(electricity)
```

```
##      electricity
## [1,]      160218
## [2,]      143539
## [3,]      148158
## [4,]      139589
## [5,]      147395
## [6,]      161244
```

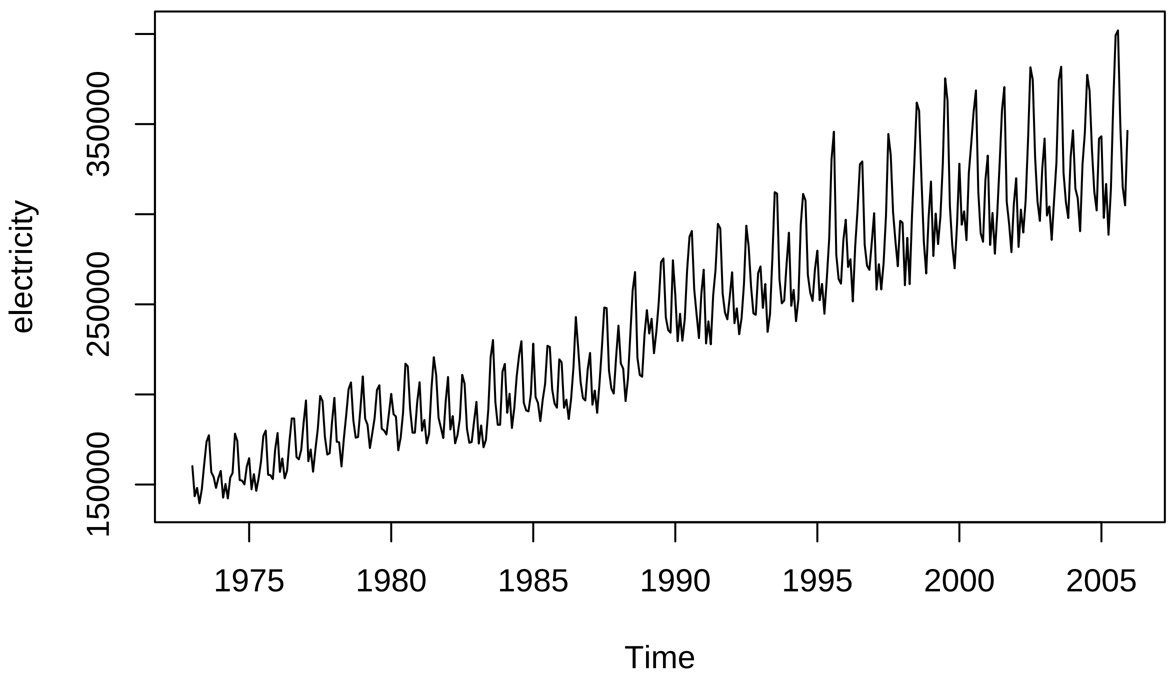
```
tail(electricity)
```

```
##      electricity
## [1,]      399252
## [2,]      401978
## [3,]      348812
## [4,]      315034
## [5,]      304899
## [6,]      346254
```

Dataset Explanation:

- **Date:** The time index for the observations. Each observation represents a specific month, ranging from January 1973 to December 2005.
- **Electricity Demand:** The amount of electricity consumed or demanded during each month, measured in megawatt hours (MWh).

```
plot(electricity,type="l",lwd=1)
```



We are going to select the power demand value for analysis.

```
y = rep(NA,times=length(electricity))
for (i in 1:length(electricity))
  y[i]=electricity[i]
length(y)
```

```
## [1] 396
```

```
n=length(y)
```

Data Exploration and Visualization

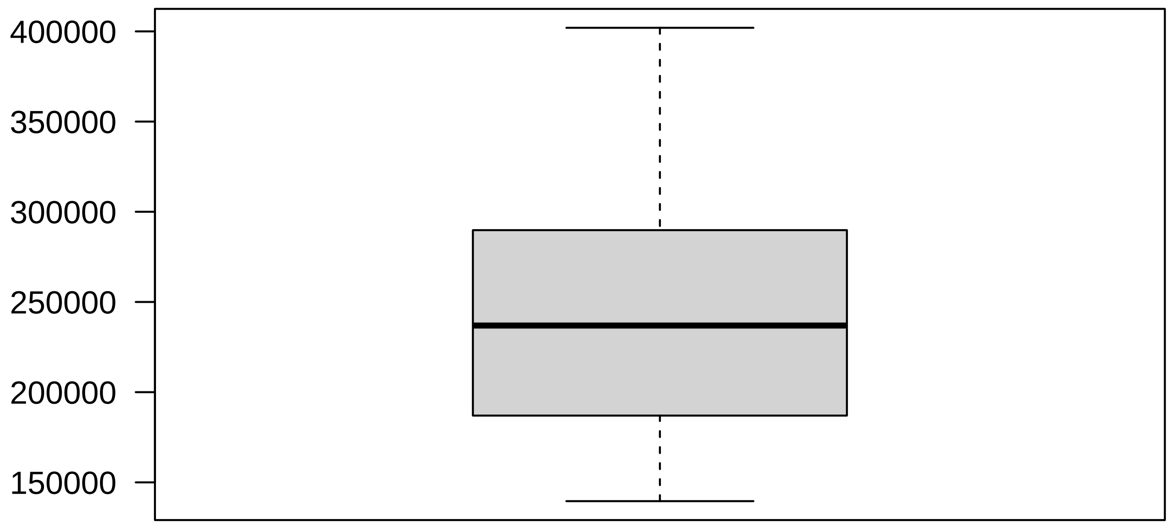
We have plotted the summary of the time series to understand the range of values in the time series. From the result below we can understand that the data value ranges from 139589 to 401978. This means that we might have to do data transformation. The mean of the data is not zero so there is a need of mean adjusting too.

```
summary(y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  139589  187032  236932  241404  289788  401978
```

From the box plot we can understand the mean of the data and the range and the distribution.

```
boxplot(y,las=2)
```



A time series is considered to be stationary if it satisfies the following conditions:

1. **Constant Mean**: The mean of the series remains constant over time.
2. **Constant Variance**: The variance of the series remains constant over time.
3. **Constant Autocovariance**: The autocovariance (covariance between observations at different time lags) of the series remains constant over time.

Common types of non-stationarity in time series data include:

- **Trend**: A systematic change in the mean of the series over time, either increasing or decreasing.

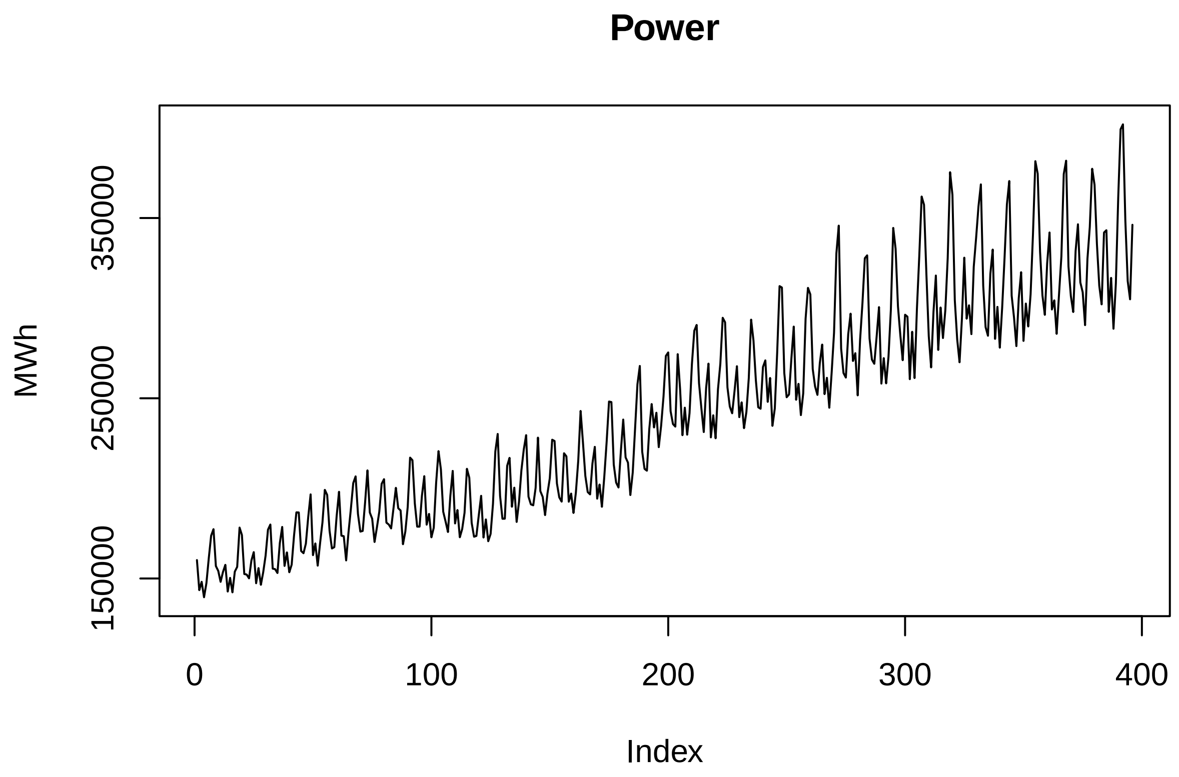
- **Seasonality**: Periodic fluctuations or patterns that occur at regular intervals, such as daily, weekly, or yearly cycles.
- **Heteroscedasticity**: Changes in the variance of the series over time.

The time series data is plotted to understand the structure of the series. The following are the main aspects that were understood from the plot.

- The mean of the data is not 0 because the red line intersects the y axis at around 250000.
- The data is non stationary since the plot shows a presence of trend and seasonality and non constant variance.
- The lower values shows a lower variability as compared to the higher values.
- There is a presence of an upward linear trend.
- There could be a presence of seasonality because there is a periodic pattern.
- The variance is increasing over time. So data transformation is needed

It can be observed that this plot is similar to Milk and Airlines dataset example that was used during the course of the study.

```
val_x=c(0,(1:4)*100)
val_y=(c(1.5:4)*100000)
plot(y,main  = "Power",ylab = "MWh",type="l",xaxt="n",yaxt="n",lwd=1)
axis(1,val_x,val_x)
axis(2,val_y,val_y)
```

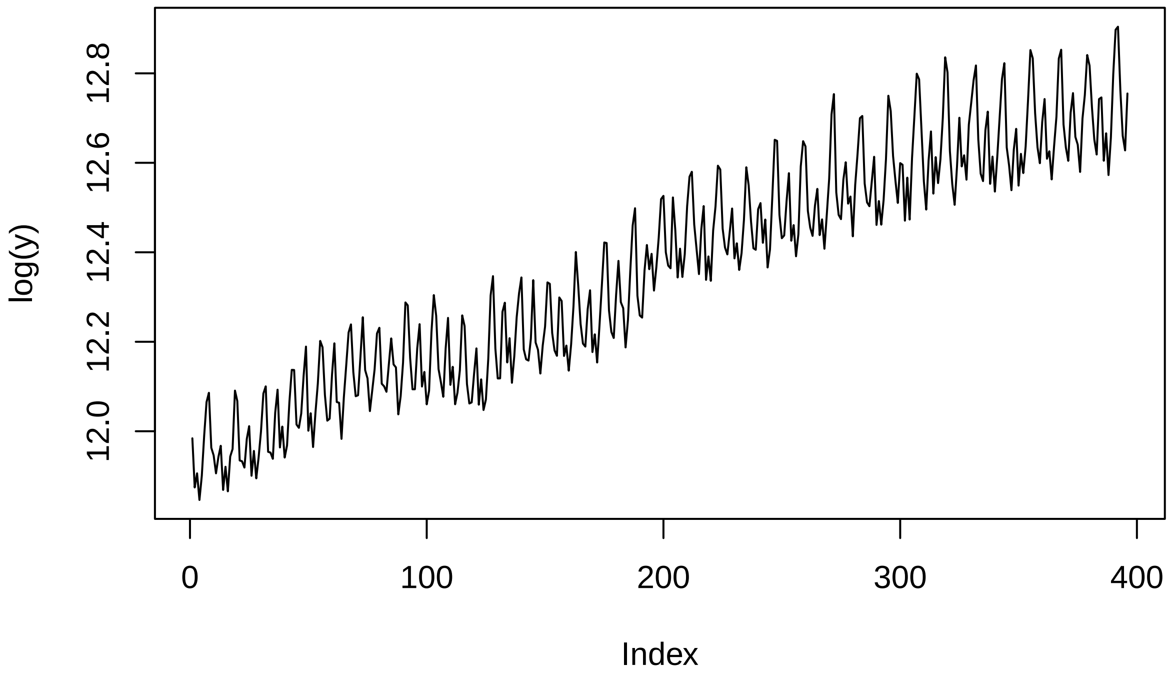


Data Preprocessing

Since the variance of the time series data increases or decreases over time (i.e., heteroscedasticity), transforming the data to a logarithmic scale can stabilize the variance. So we do a data transformation where we change the actual values to logarithmic scale.

After the transformation we can observe that the amount of variation around the upward trend is now much more uniform across high and low values of the series and the problem of non constant variance is mitigated.

```
plot(log(y),type="l")
```

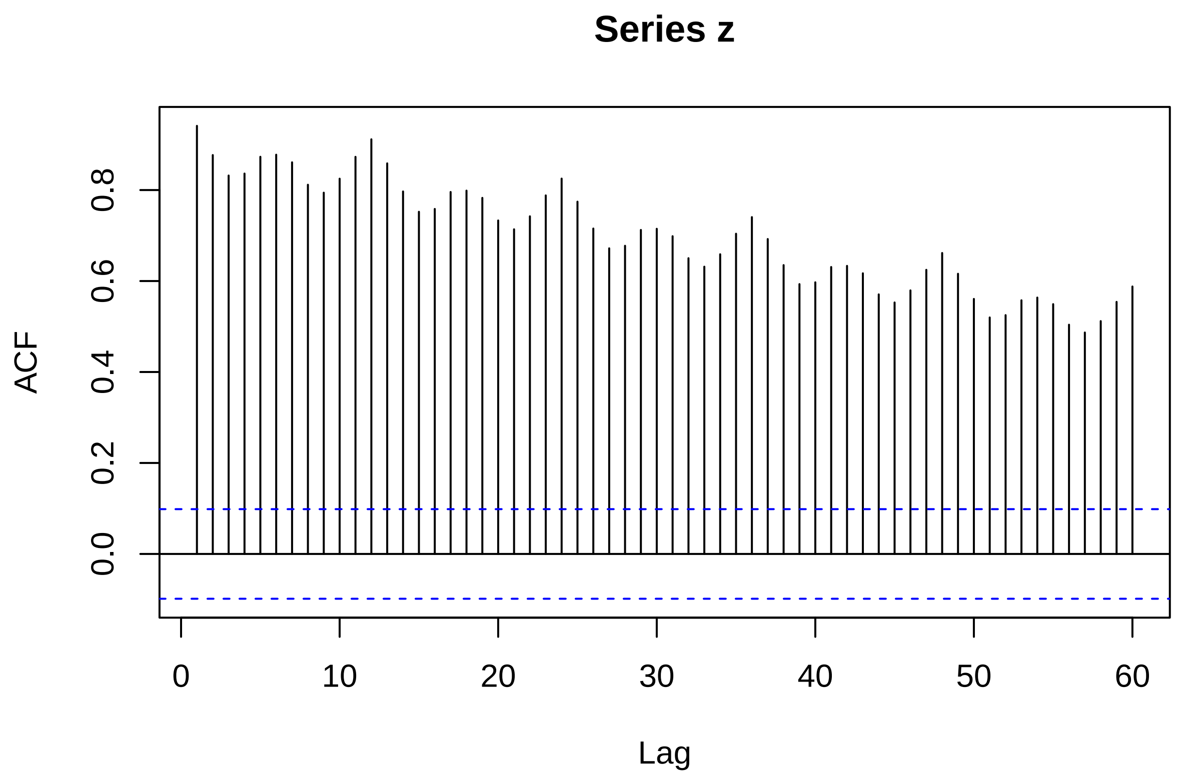


```
z=log(y)
```

ACF measure the correlation between a time series and its own lagged values. In order to identify any Autocorrelation and Patterns; the acf is plotted.

There is a linear trends as the autocorrelation coefficients decreases as the lags increases. There are also peaks in the 12, 24 . . . lags indicating yearly seasonality.

```
acf(z,lag.max = 60)
```

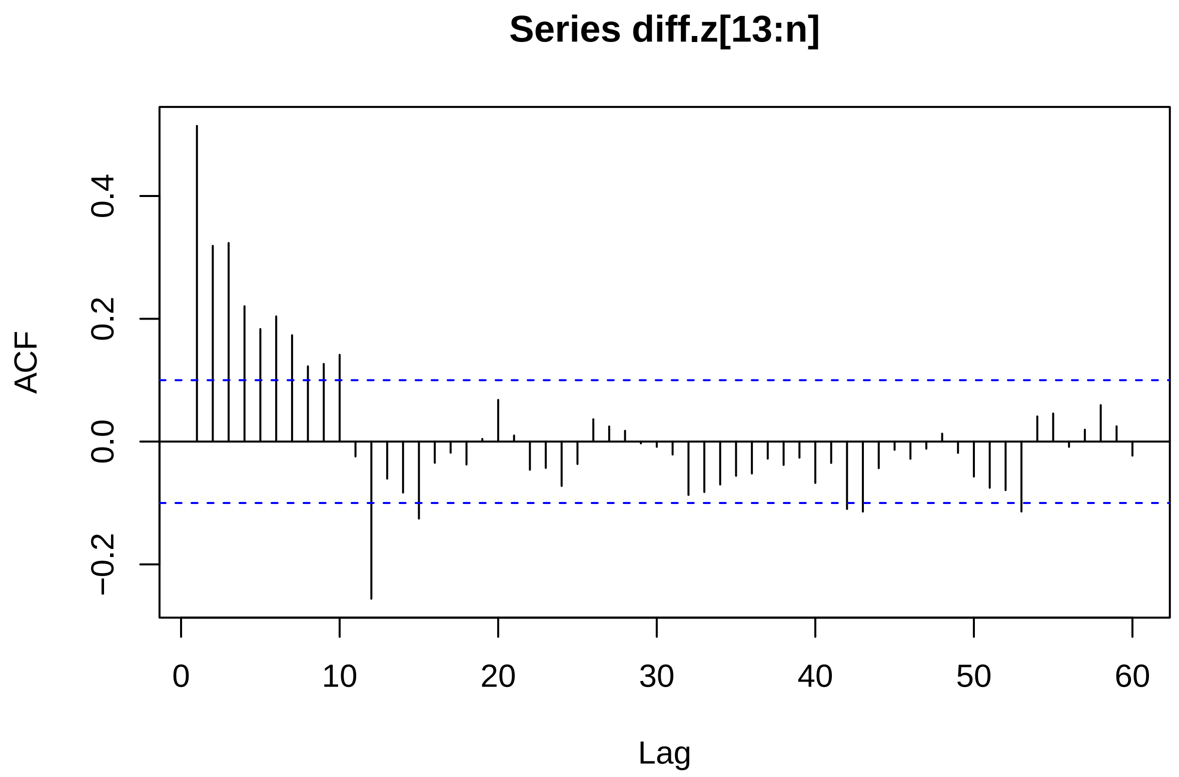


Lets remove the yearly seasonality and check the difference in the ACF plot.

```
n= length(z)
diff.z=rep(NA,times=n)
for (t in 13:n) {
    diff.z[t]=z[t]-z[t-12]
}
```

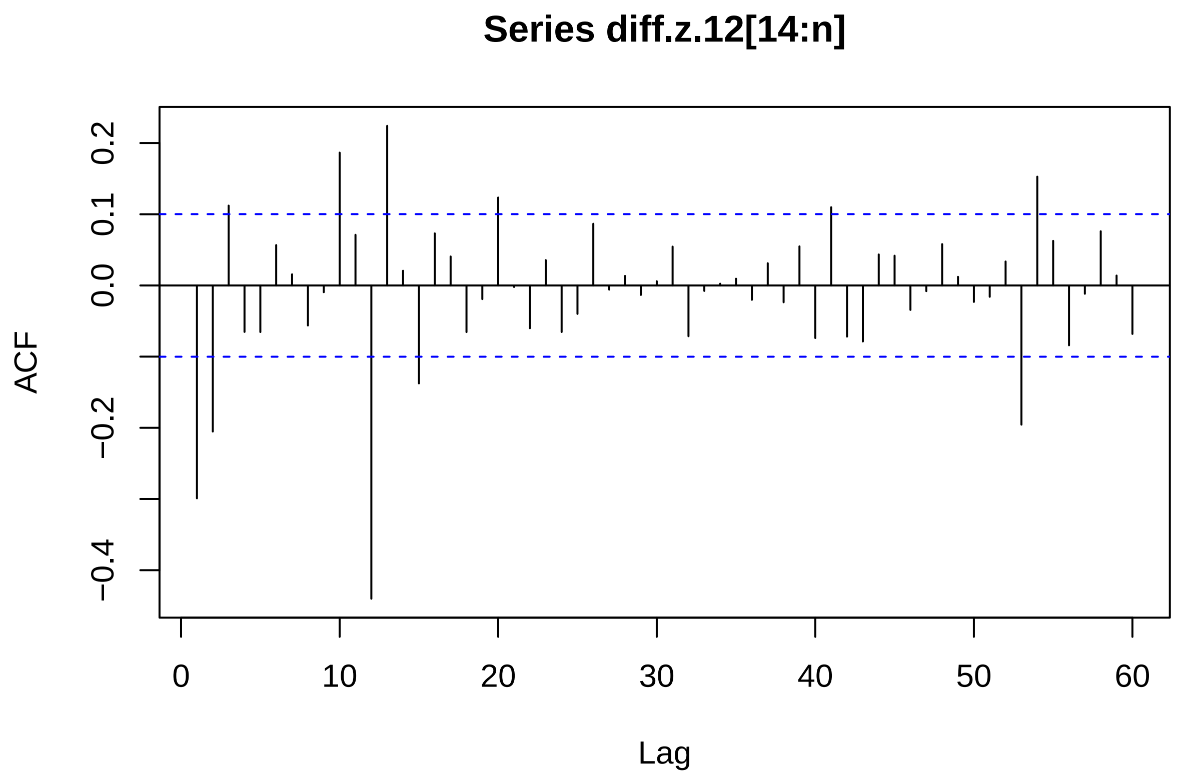
We can see that now there are only linear trends present. Lets remove them too.

```
acf(diff.z[13:n],lag.max = 60)
```



The ACF has a peak in the 1,2 lag and an extremely significant peak in the 12th lag.

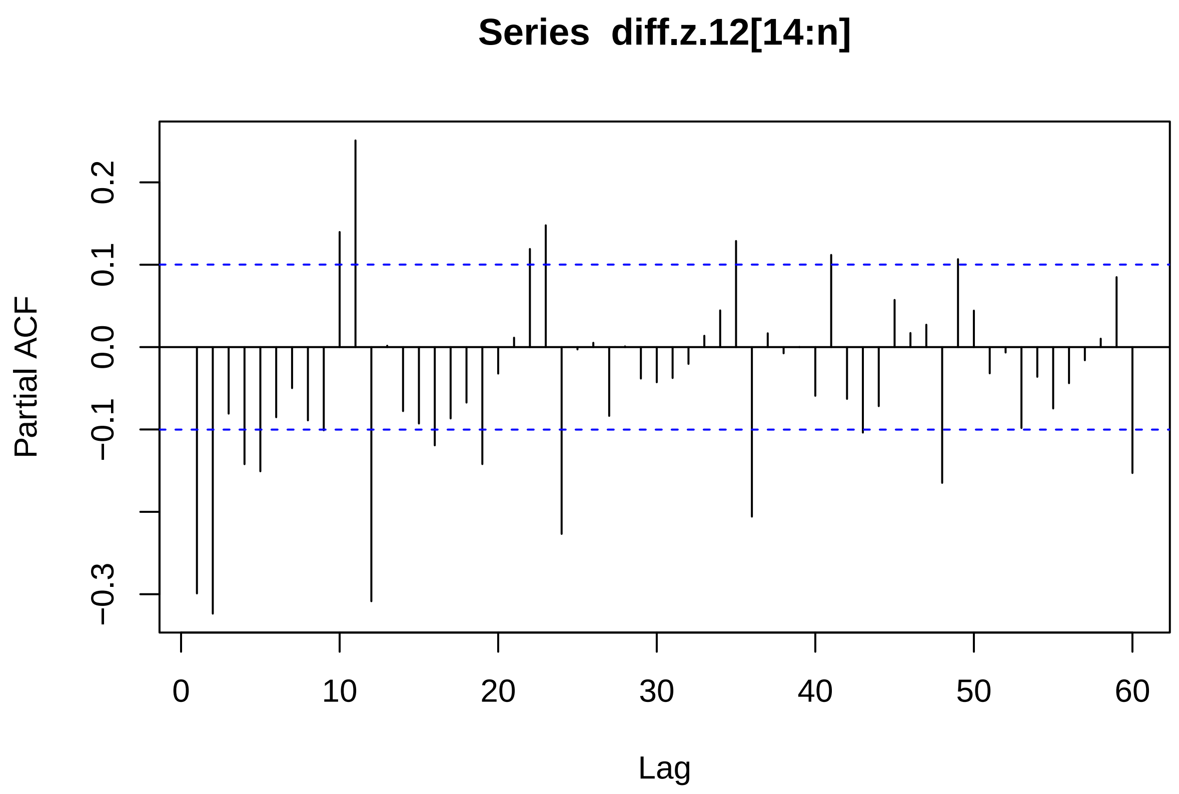
```
diff.z.12=rep(NA,times=n)
for (t in 13:n) {
    diff.z.12[t]=diff.z[t]-diff.z[t-1]
}
acf(diff.z.12[14:n],lag.max = 60)
```



PACF measures the correlation between a time series and its own lagged values after removing the effects of shorter lags.

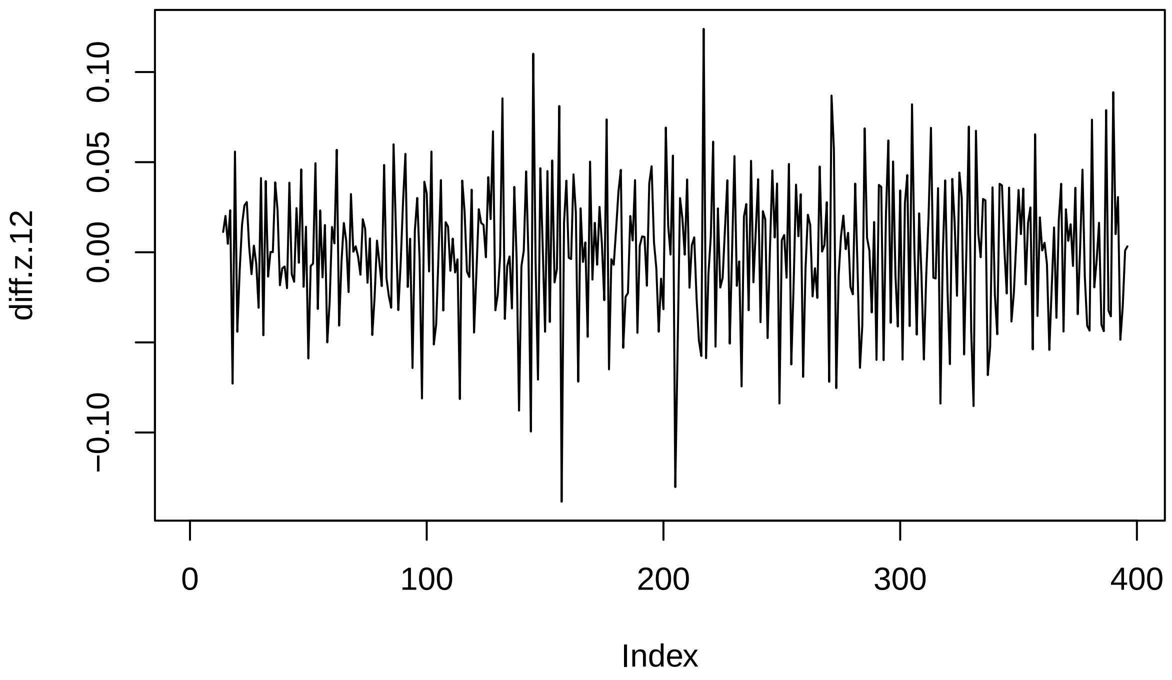
Here there are significant lags present in the 1,2 ; 11,12 th lag

```
pacf(diff.z.12[14:n],lag.max = 60)
```



The plot below shows that with the necessary data transformation we have converted out non stationary time series to a stationary time series ie with constant mean and variance.

```
plot(diff.z.12,type="l")

abline(h=mean(diff.z.12),col="red",lwd=1.5,lty=3)
```



Now that we have removed the trends and seasonality present in the dataset, next we need to transform the data such that the mean is 0. We can see that the mean of the data is 12.36; we subtract 12.36 from every values of the previous dataset.

After the transformation the mean is now approximately 0 and the range varies from -0.51 to +0.54.

```
mean=mean(z)

mean
```

```
## [1] 12.36183
```

```
log.z=z-mean

mean(log.z)
```

```
## [1] -1.012097e-16
```

```
range(log.z)
```

```
## [1] -0.5153678  0.5423272
```

Modeling Approach

In choosing a model, we shall attempt to adhere to the **principle of parsimony**; that is, the model used should require the smallest number of parameters that will adequately represent the time series.

Since the data has seasonality and linearity. We need to choose a seasonal ARIMA model with seasonality 12.

Lets start with (0,1,1)x(0,1,1)12 SARIMA model.

where d = 1, q=1,d=*1*,*q*=1

SARIMA(0,1,1)x(0,1,1)12 model specifies a time series model that includes first-order differencing, a first-order moving average term, and a seasonal moving average term to capture both the non-seasonal and seasonal dynamics of the data.

If the model is inadequate maybe later add some auto regressive terms

$$diff.z.12[t] = A[t]+\theta_1*A[t\text{-}1]+*\eta_{12}A_{t\text{-}12}+\theta_1*\eta_{12}A_{t\text{-}13} where (1+\theta_1 x)(1+\eta_{12}x) = 0 i.e.\ 1+\theta_1 x+\eta_{12}x+\theta_1*\eta_{12}*x^2 = 0$$

We are going to use Maximum Likelihood estimation for the estimation of the coefficients. Maximum Likelihood estimation involves finding the values of the AR, MA, and differencing parameters that maximize the likelihood of observing the time series data. This is typically done using numerical optimization techniques to iteratively search for the parameter values that maximize the likelihood function.

```
ARIMA.elec=arima(log.z[2:n],order=c(0,1,1),include.mean=FALSE,seasonal=list(order=c(0,1,1),period=12),me
log_likelihood <- logLik(ARIMA.elec)
k <- length(coef(ARIMA.elec))
# Number of observations
n <- length(log.z)
# Calculate BIC
BIC <- -2 * log_likelihood + k * log(n)
AIC.tsa=ARIMA.elec$aic
AIC.tsa
```

```
## [1] -1653.617
```

```
BIC
```

```
## 'log Lik.' -1645.654 (df=3)
```

AIC stands for Akaike's Information Criterion. It says to select the model that minimizes

$$AIC = \text{-}2 * log(MLE) + 2k MLE : Maximium\ Likelihood\ Estimate\ and\ k = p + q + 1$$

BIC stands for Bayesian Information Criterion. Is says to select the mode that minimizes

$$BIC = \text{-}2log(MLE) + klog(n)$$

In this case since we are using a single model this approach is useless. So we fill check the accuracy of the model by calculating the RMSE and RAE later.

```
ARIMA.elec
```

```
##
## Call:
## arima(x = log.z[2:n], order = c(0, 1, 1), seasonal = list(order = c(0, 1, 1),
##     period = 12), include.mean = FALSE, method = "ML")
```

```
##
## Coefficients:
##           ma1     sma1
##       -0.5063  -0.8302
## s.e.   0.0755   0.0317
##
## sigma^2 estimated as 0.0007357:  log likelihood = 828.81,  aic = -1653.62
```

The estimated coefficients for the ARIMA model are

for the Moving Average term θ_1: -0.5063

for the Seasonal Moving Average term η_{12}: - 0.8302

the estimated variance σ^2 is 0.0007357

Lower AIC values indicate better fitting models, balancing goodness of fit with model complexity.

Model Development

Now we are going to fit the model with the estimates with the data that was transformed. The data was altered to remove trends,seasonality and altered the mean to zero to act like a stationary time series.

```
est.theta1=ARIMA.elec$coef[1]

est.theta1
```

```
##        ma1
## -0.5063123
```

```
est.eta12=ARIMA.elec$coef[2]
est.eta12
```

```
##       sma1
## -0.8302146
```

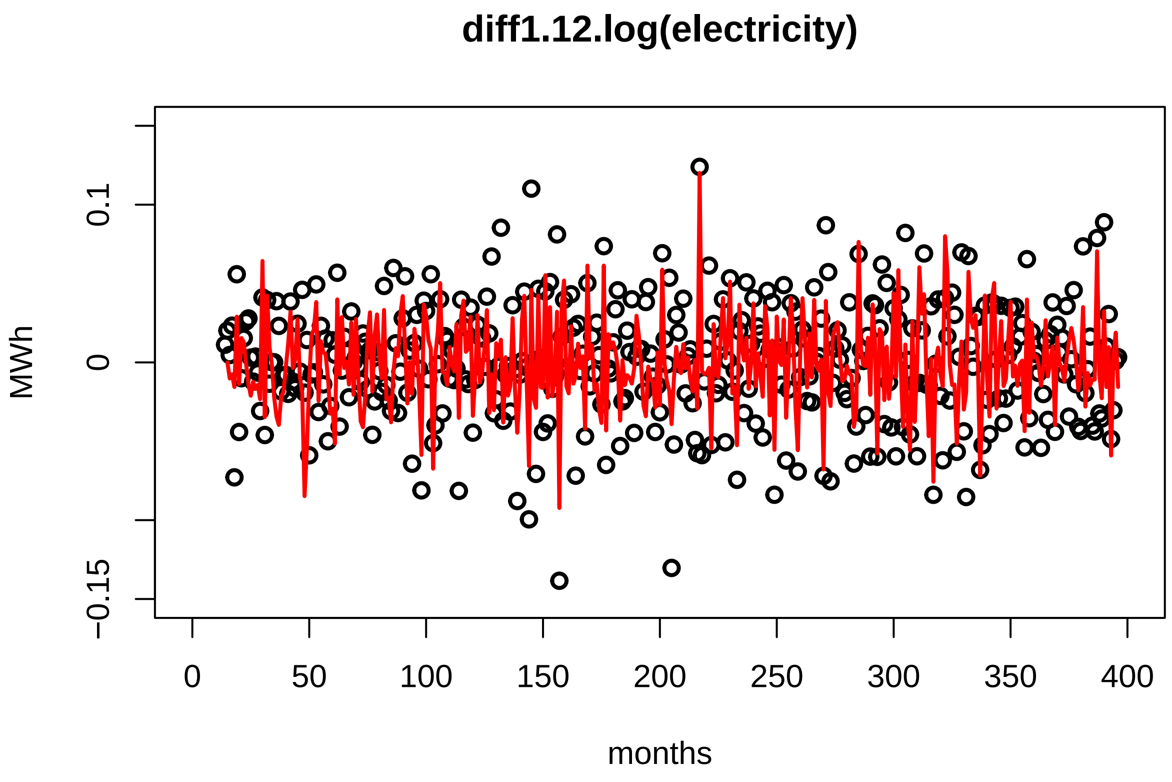
```
est.sigma2=ARIMA.elec$sigma2

est.sigma2
```

```
## [1] 0.0007357051
```

```
diff1.12.log.z.fit=rep(NA,times=n)

aa=rep(0,times=n+15)   # "+ 15" for predictions

for (t in 15:n) {
    diff1.12.log.z.fit[t]=est.theta1*aa[t-1]+est.eta12*aa[t-12]+est.theta1*est.eta12*aa[t-13]
      aa[t]=diff.z.12[t]-diff1.12.log.z.fit[t]
}

range(diff.z.12,na.rm=TRUE)
```

```
## [1] -0.1384264  0.1239677
```

```
val_x=c(0,(1:7)*50,400)
val_y=c(-0.15,-0.1,0,0.1,0.15)
plot(diff.z.12,main="diff1.12.log(electricity)",xlab="months",ylab="MWh",
xlim=c(0,400),ylim=c(-0.15,0.15),
xaxt="n",yaxt="n",type="p",lwd=2)
axis(1,val_x,val_x)
axis(2,val_y,val_y)
lines(diff1.12.log.z.fit,col="red",lwd=2)
```



We can see that the model fits the data quite well. It was even able to capture some extreme points at 210, 390 etc. We can also see that the points at 201 and 153 are not fitted to the model perfectly.

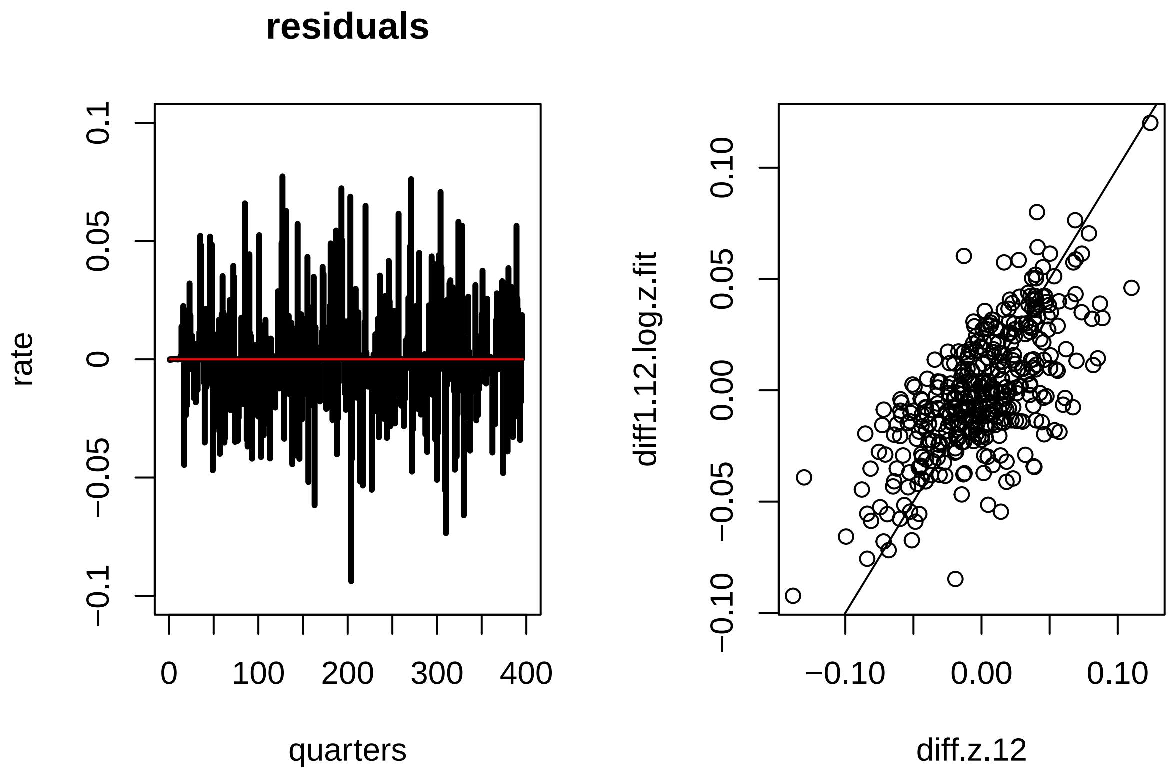
Next lets check the distribution of the residuals. To understand the goodness of the fit and the errors.

In particular, if the points in the graph tend to be close to a diagonal line (meaning that observed and predicted values are the same), it suggests that predictions made by this model are accurate. Slight deviations from this diagonal line indicate areas where under- or over-predictions take place.
If there is tight clustering of points at the center of the graph, then the model's predictions are generally close to what was actually observed in instances when there were significant disparities. Usually, it is desirable as it shows that such a model provides reliable estimates for the data.

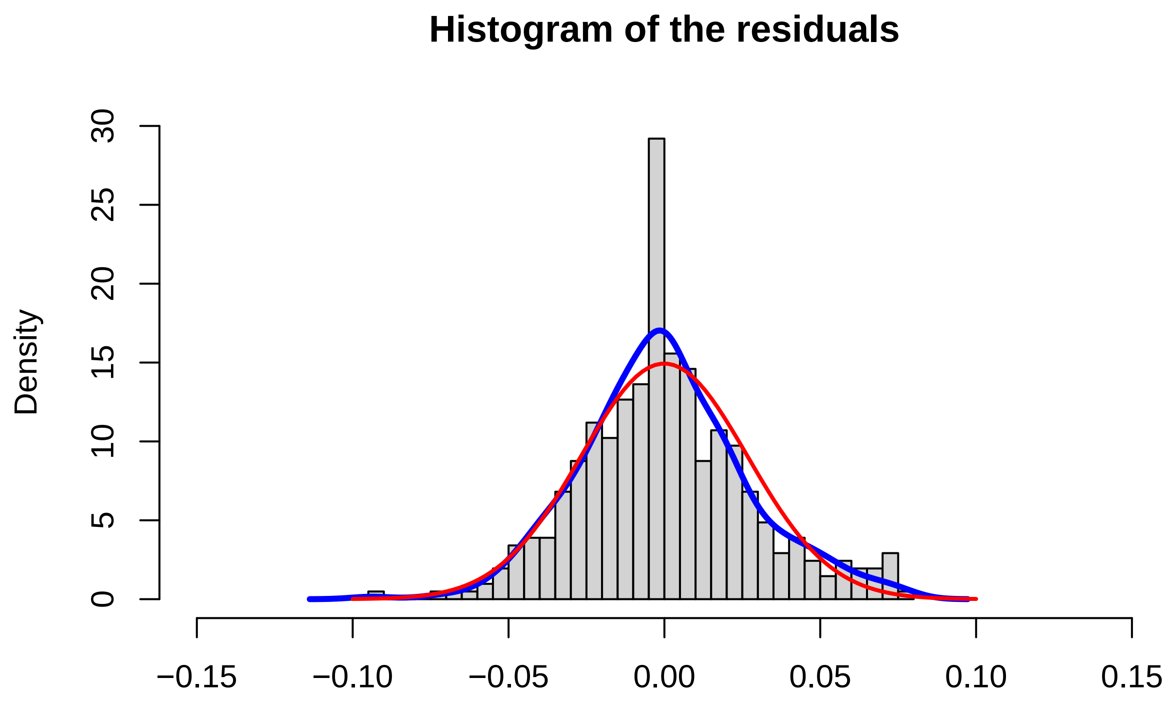
However, visual inspection must be supplemented with quantitative measures of model performance like mean squared error (MSE) to accurately assess how well that a model predicts.

On average, we can tell that a plot with observations and fitted values concentrated around middle is good enough to capture patterns in data.

```
diff1.12.log.z.res=ARIMA.elec$residuals
par(mfrow = c(1,2))
val_x=c(0,(1:8)*50)
val_y=(c(-0.1,-0.05,0,0.05,0.1))
plot(diff1.12.log.z.res,main="residuals",xlab="quarters",ylab="rate",xlim=c(0,400),ylim=c(-0.1,0.1),
xaxt="n",yaxt="n",type="h",lwd=3)
axis(1,val_x,val_x)
axis(2,val_y,val_y)
lines(rep(0,times=n),type="l",col="red",lwd=1)

plot(diff.z.12,diff1.12.log.z.fit)
abline(a=0,b=1)
```



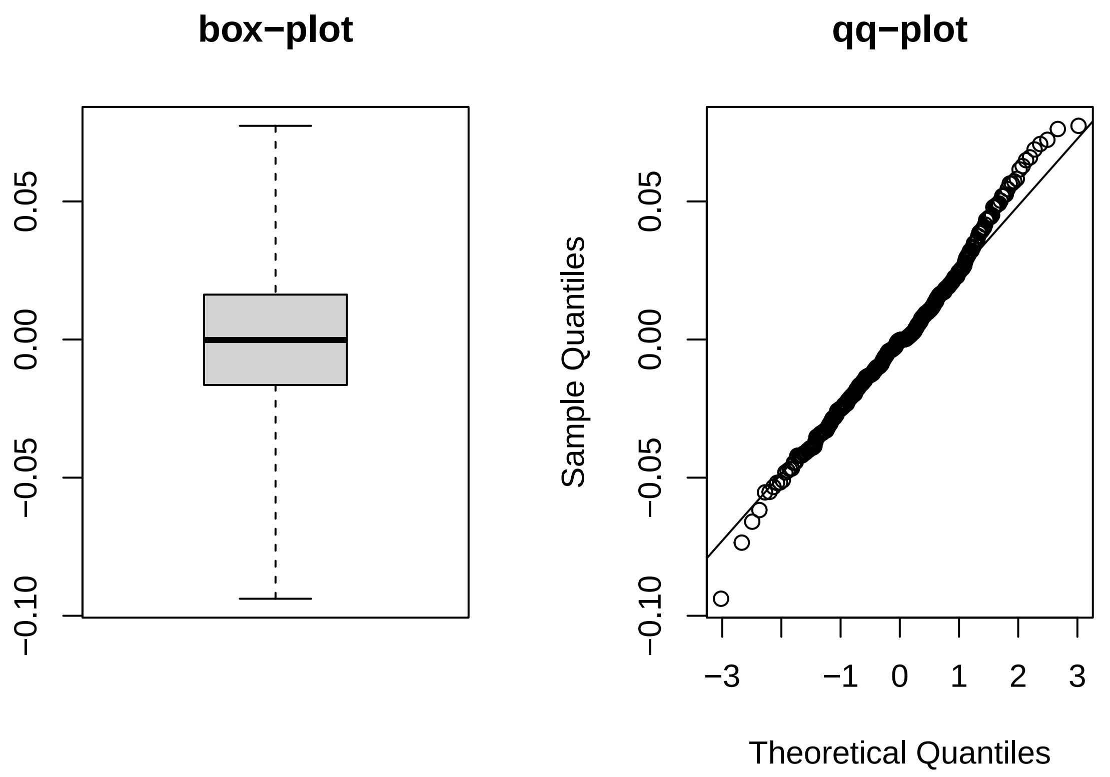
```
hist(aa,main="Histogram of the residuals",xlab="",freq=F,
xlim=c(-0.15,0.15),ylim=c(0,30),breaks=40)
lines(density(diff1.12.log.z.res),col="blue",lwd=3)
zz=seq(-0.10,0.10,length=100)
f.zz=dnorm(zz,mean(diff1.12.log.z.res),sd(diff1.12.log.z.res))
lines(zz,f.zz,col="red",lwd=2)
```



Normality can be checked more carefully by plotting the so-called normal scores or quantile-quantile (QQ) plot. Such a plot displays the quantiles of the data versus the the- oretical quantiles of a normal distribution. With normally distributed data, the QQ plot looks approximately like a straight line.

The Boxplot and the Histogram suggests that the mean is 0 and the residuals are normally distributed.

```
par(mfrow=c(1,2))
boxplot(diff1.12.log.z.res,main="box-plot",range=0)
qqnorm(diff1.12.log.z.res,main="qq-plot")
qqline(diff1.12.log.z.res)
```



The Shapiro-Wilk normality helps us determine if the values in our dataset are likely to have been drawn from a population that follows a normal (Gaussian) distribution.

1. **Null Hypothesis** H_0: The null hypothesis of the Shapiro-Wilk test is that the data are normally distributed.
2. **Alternative Hypothesis** H_a: The alternative hypothesis is that the data are not normally distributed.

If the p-value associated with the test statistic is greater than the chosen significance level (commonly 0.05), we fail to reject the null hypothesis. This means that there is not enough evidence to conclude that the data significantly deviate from a normal distribution.

If the p-value is less than the chosen significance level, we reject the null hypothesis in favor of the alternative hypothesis. This indicates that the data significantly depart from a normal distribution.

Here the p-value = 0.02243 <0.05

Hence not normally distributed

The Jarque-Bera test is another statistical test used to assess the normality of a dataset. It specifically tests whether the sample data have skewness and kurtosis that are consistent with a normal distribution.

- **Null Hypothesis** H_0: The null hypothesis of the Jarque-Bera test is that the data are normally distributed.
- **Alternative Hypothesis** H_a: The alternative hypothesis is that the data are not normally distributed.

If the p-value associated with the test statistic is greater than the chosen significance level (commonly 0.05), we fail to reject the null hypothesis. This means that there is not enough evidence to conclude that the data significantly deviate from a normal distribution based on skewness and kurtosis.

If the p-value is less than the chosen significance level, we reject the null hypothesis in favor of the alternative hypothesis. This indicates that the skewness and kurtosis of the data significantly depart from what would be expected under a normal distribution.

Here the p-value = 7.48e-05 <0.05

Hence not normally distributed

Here the mean = -3.5e-05 which is approximately equal to zero.

The skewness of a dataset measures the asymmetry of its distribution. Specifically, it quantifies the degree to which the distribution of the data deviates from symmetry around its mean.

Overall, a skewness value of 0.2075919 indicates slight positive skewness in the distribution of the residuals, but it is not a cause for major concern, since the skewness value is relatively small (close to zero), the skewness is not very pronounced, and the distribution may still be reasonably symmetric.

```
shapiro.test(diff1.12.log.z.res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  diff1.12.log.z.res
## W = 0.99146, p-value = 0.02243
```

```
library (tseries)
```

```
## Warning: package 'tseries' was built under R version 4.2.3
```

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

```
jarque.bera.test(log.z)
```

```
##
##  Jarque Bera Test
##
## data:  log.z
## X-squared = 19.001, df = 2, p-value = 7.48e-05
```

```
mean(diff1.12.log.z.res) # mean close to zero
```

```
## [1] -3.555857e-05
```

```
skewness(diff1.12.log.z.res)
```

```
## [1] 0.2075919
```

Root mean squared error (RMSE) and Mean absolute error (MAE) are popular metrics for evaluating the performance of time series model. These two metrics help to measure the average size of errors made by the model in its predictions, though they vary in how they compute and interpret these errors.
Root mean squared error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i\text{-}\hat{y}_i)^2}$$

Interpretation; Lower RMSE indicates better prediction quality with lesser difference between predicted and observed values.

Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i\text{-}\hat{y}_i|$$

Unlike RMSE that squares errors, MAE does not give outliers much effect.

Both RMSE and MAE are close to zero indicating that the model is good.

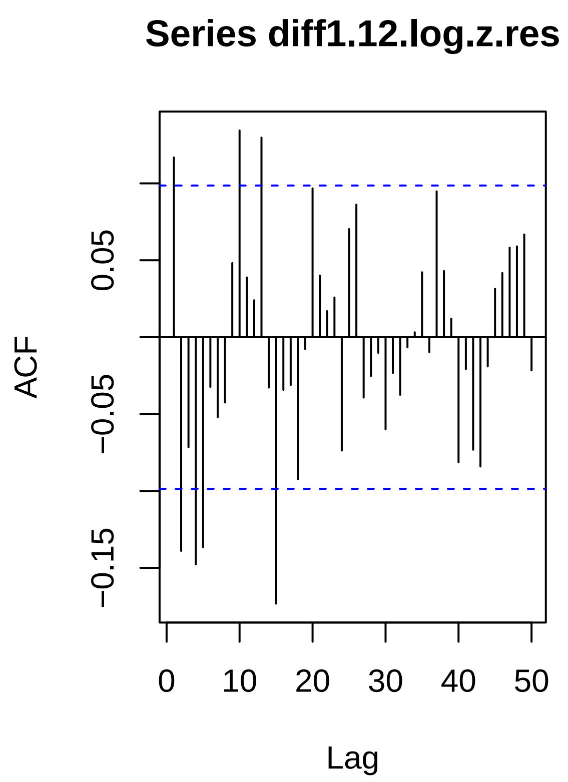
```
RMSE=sqrt(mean(diff1.12.log.z.res^2))

RMSE
```

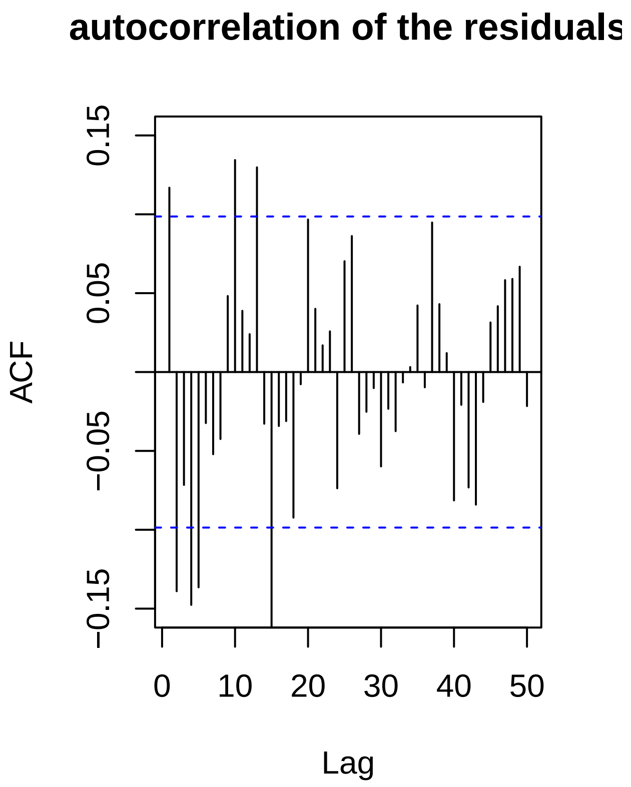
```
## [1] 0.02667391
```

```
MAE=mean(abs(diff1.12.log.z.res))

MAE
```

```
## [1] 0.02041978
```

```
par(mfrow=c(1,2))
plot(acf(diff1.12.log.z.res,lag.max=50),main="autocorrelation of the residuals"
     ,ylim=c(-0.15,0.15))
```





Now lets fit the model with the log(z). ie undo the seasonal and trends data transformation.

```
log.z.fit=rep(NA,times=n)

aa=rep(0,times=n)

log.z.fit[1:13]=log.z[1:13]



for (t in 14:n) {
    log.z.fit[t]=log.z[t-1]+log.z[t-12]-log.z[t-13]+est.theta1*aa[t-1]+est.eta12*aa[t-12]+est.theta1*es
    aa[t]=log.z[t]-log.z.fit[t]
}
```

```
range(log.z)
```

```
## [1] -0.5153678  0.5423272
```

```
shapiro.test(log.z.fit) #Shapiro-Wilk Normality - independent data!
```

```
##
##  Shapiro-Wilk normality test
##
## data:  log.z.fit
## W = 0.96765, p-value = 1.115e-07
```

```
#The residuals are Normal if the p-value is above 0.05
```

We can see that the fitted values are not normally distributed by examining the Shapiro Wilk and Jarque Bera Test. The mean is approximately close to 0 and skewness is negligible.

```
library (tseries)
jarque.bera.test(log.z.fit) #The data are Normal if the p-value is above 0.05
```

```
##
##  Jarque Bera Test
##
## data:  log.z.fit
## X-squared = 19.567, df = 2, p-value = 5.636e-05
```

```
mean(log.z.fit) # mean close to zero
```

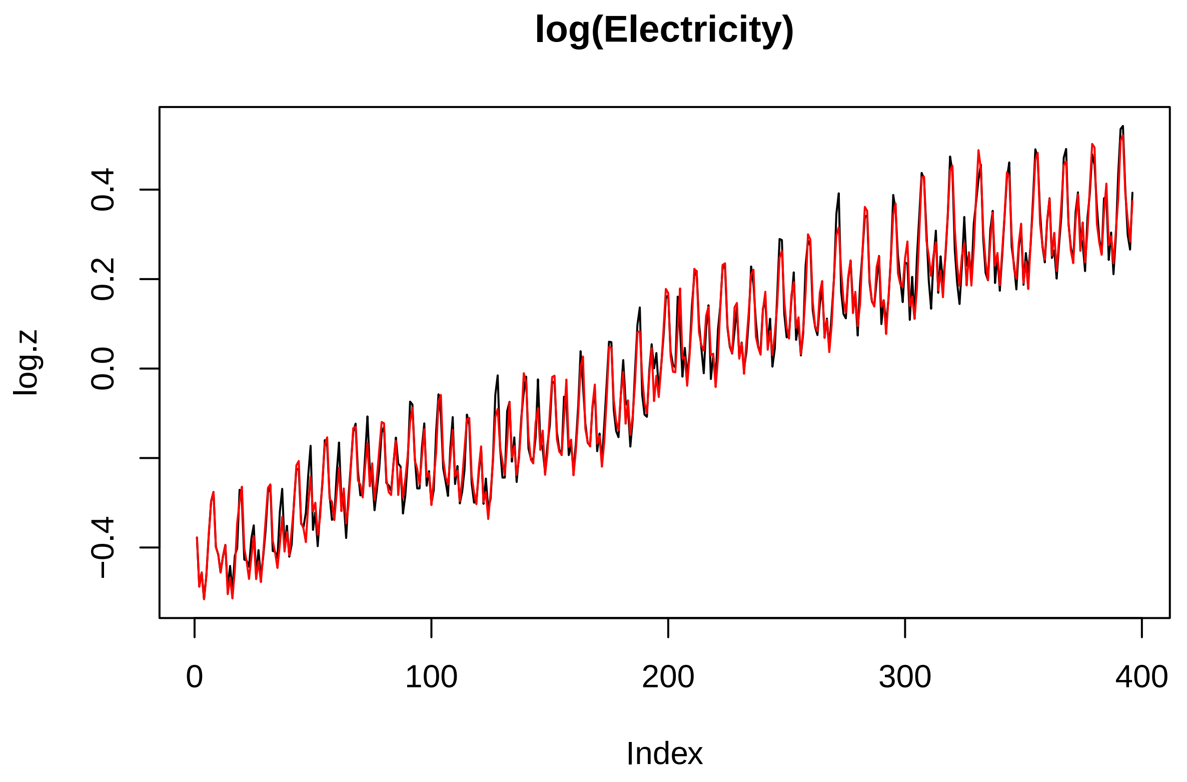
```
## [1] -0.001113064
```

```
skewness(log.z.fit)
```

```
## [1] -0.02793159
```

The red line is the fitted model and the black line is the actual model. The model fits the actual dataset quite well from the plot.

```
plot(log.z,main="log(Electricity)",type="l")
lines(log.z.fit,col="red",lwd=1)
```

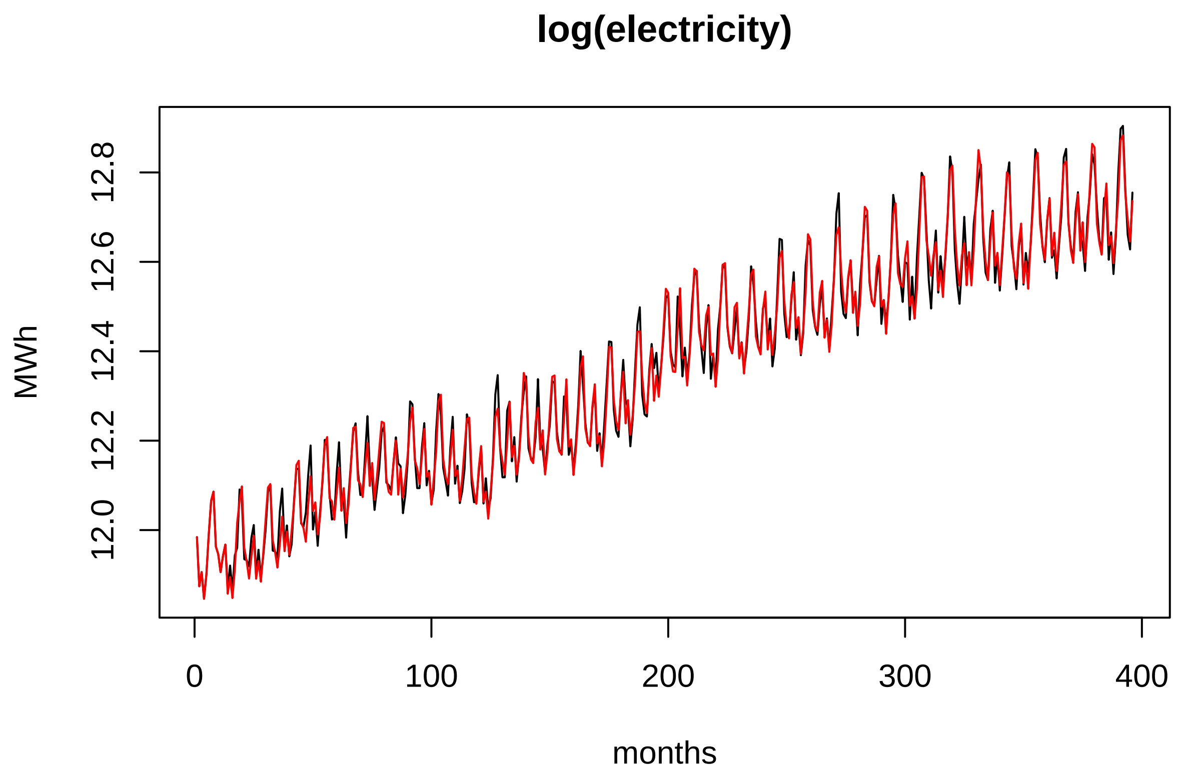


Next we need to remove the mean adjusting that was done.

```
range(log.z+mean)
```

```
## [1] 11.84646 12.90415
```

```
val_x=c(0,(1:9)*24)
val_y=c(0:3)
plot(log.z+mean,main="log(electricity)",xlab="months",ylab="MWh",type = "l")
lines(log.z.fit+mean,col="red",lwd=1)
```

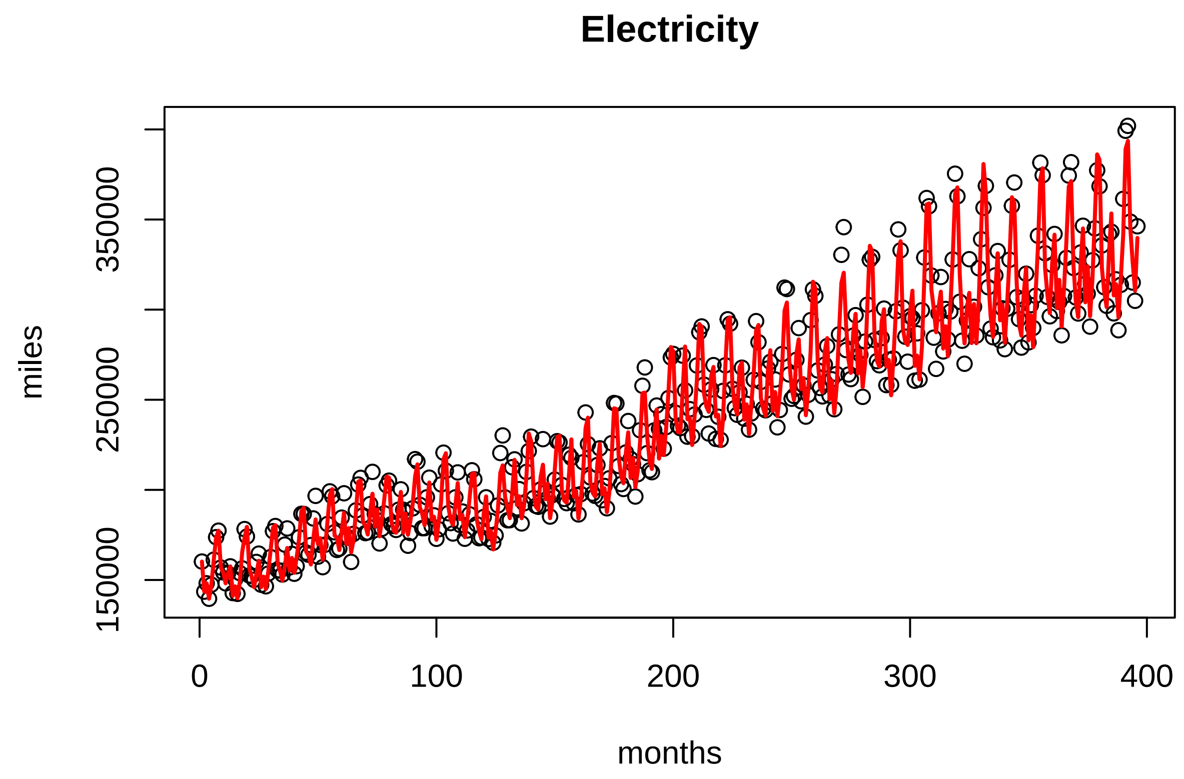


Now lets remove the only pending data transformation that wasn't removed i.e converting the data to log scale. So we undo it but exp(fit+mean)

```
range(y)
```

```
## [1] 139589 401978
```

```
plot(y,main="Electricity",xlab="months",ylab="miles")
lines(exp(log.z.fit+mean),col="red",lwd=2)
```



We can notice that the model has seasonality, trends and increasing variance as the original dataset.

Forecasting

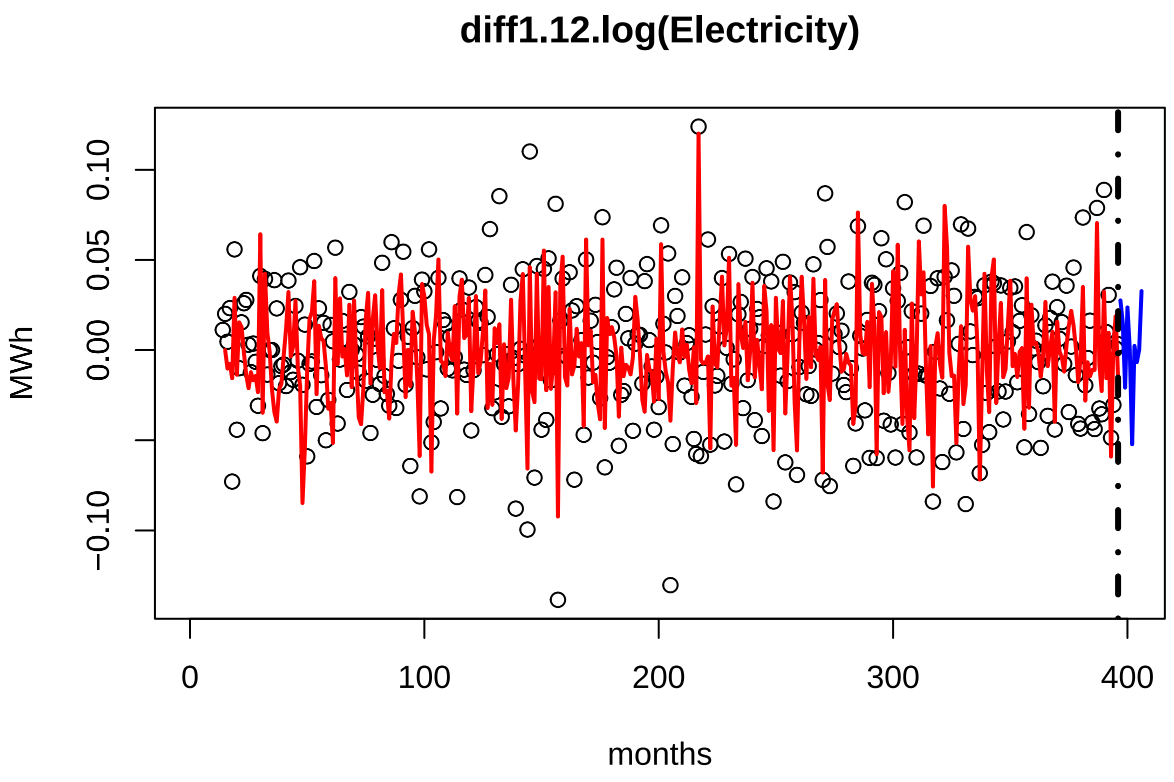
Let us attempt to forecast the electricity consumption for the next 10 months.

$diff1.12.log.z.for[n+1] = \hat{\theta_1} * A[n] + \hat{\eta_{12}} * A[n\text{-}11] + \hat{\theta_1} * \hat{\eta_{12}} * A[n\text{-}12] diff1.12.log.z.for[n+2] = \hat{\eta_{12}} * A[n\text{-}10] + \hat{\theta_1} * \hat{\eta_{12}} * A[n\text{-}11]...$

```
nn=n+10

diff1.12.log.z.for=rep(NA,times=nn)

diff1.12.log.z.for[n+1]=est.theta1*aa[n]+est.eta12*aa[n-11]+est.theta1*est.eta12*aa[n-12]

diff1.12.log.z.for[n+2]=est.eta12*aa[n-10]+est.theta1*est.eta12*aa[n-11]

diff1.12.log.z.for[n+3]=est.eta12*aa[n-9]+est.theta1*est.eta12*aa[n-10]

diff1.12.log.z.for[n+4]=est.eta12*aa[n-8]+est.theta1*est.eta12*aa[n-9]

diff1.12.log.z.for[n+5]=est.eta12*aa[n-7]+est.theta1*est.eta12*aa[n-8]

diff1.12.log.z.for[n+6]=est.eta12*aa[n-6]+est.theta1*est.eta12*aa[n-7]
```

```
diff1.12.log.z.for[n+7]=est.eta12*aa[n-5]+est.theta1*est.eta12*aa[n-6]

diff1.12.log.z.for[n+8]=est.eta12*aa[n-4]+est.theta1*est.eta12*aa[n-5]

diff1.12.log.z.for[n+9]=est.eta12*aa[n-3]+est.theta1*est.eta12*aa[n-4]

diff1.12.log.z.for[n+10]=est.eta12*aa[n-2]+est.theta1*est.eta12*aa[n-3]
```

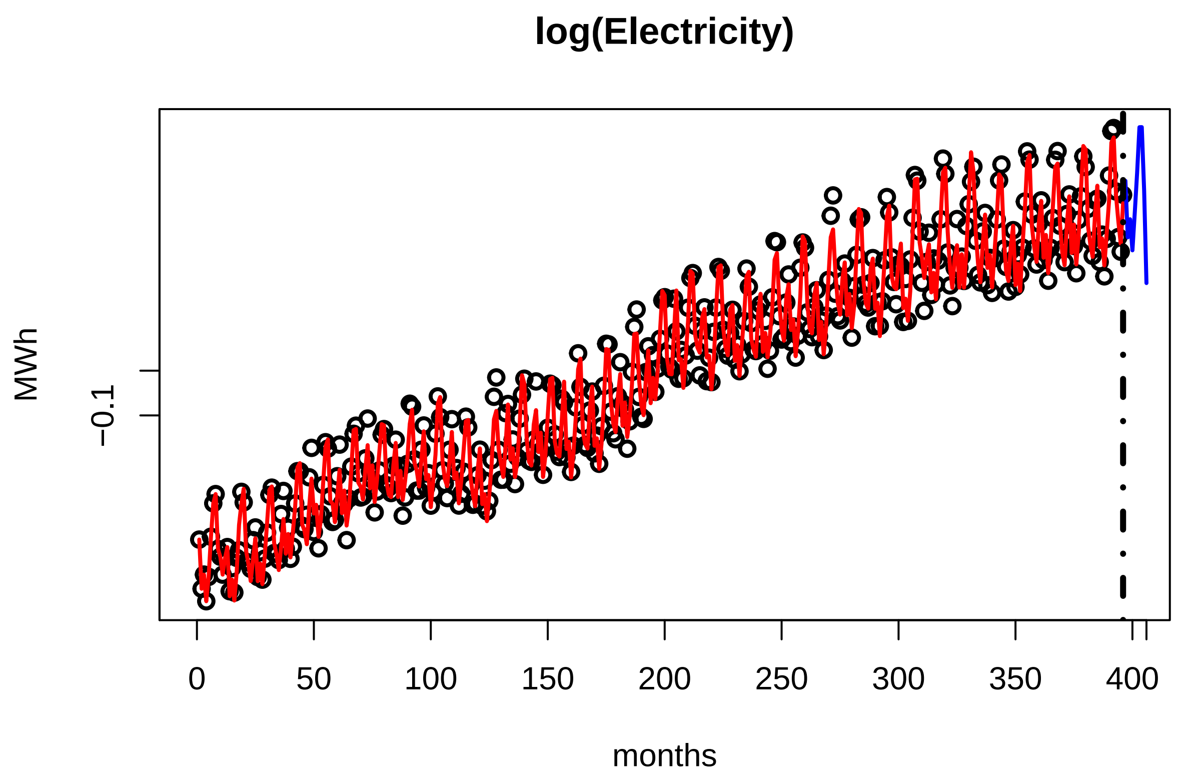
```
plot(diff.z.12,main="diff1.12.log(Electricity)",xlab="months",ylab="MWh",xlim = c(1,400))
lines(diff1.12.log.z.fit,col="red",lwd=2)
lines(diff1.12.log.z.for,type="l",lwd=2,col="blue")
abline(v=n,col="black",lwd=3,lty=4)
```



Now lets forecast after removing the seasonality and trends.

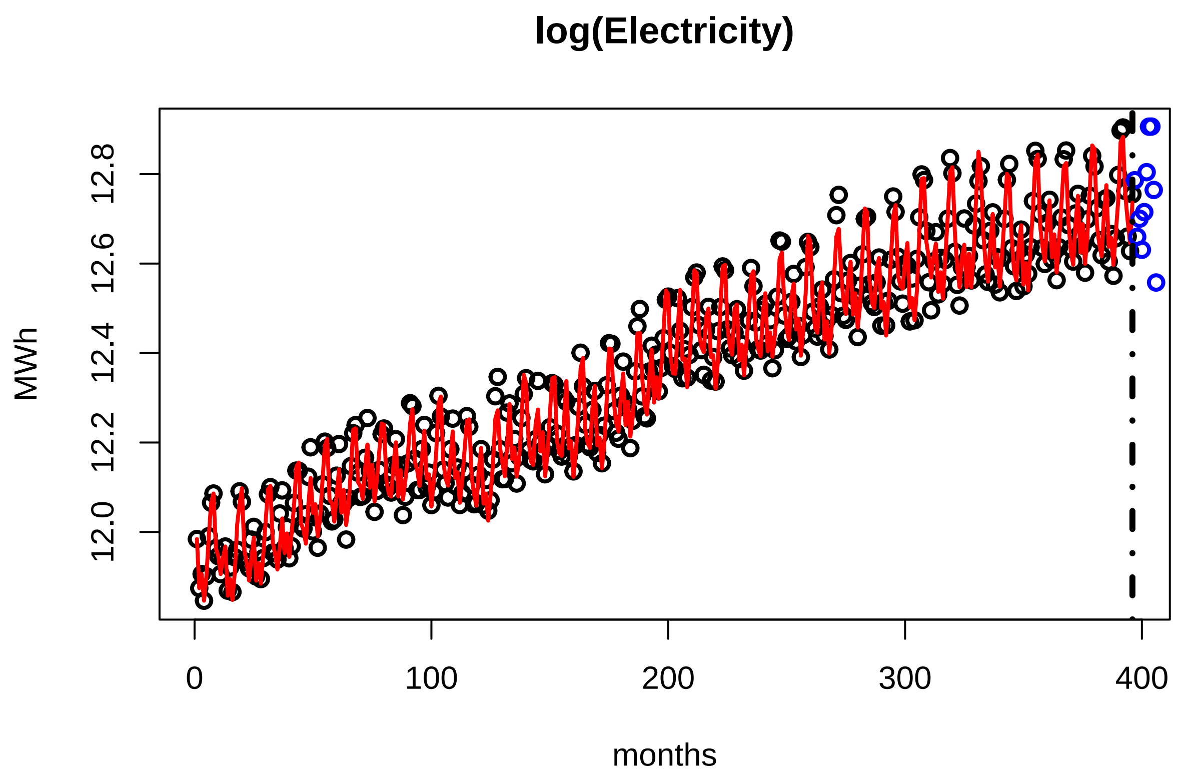
```
log.z.for=rep(NA,times=n)

log.z.for[n+1]=log.z[n]+log.z[n-11]-log.z[n-12]+est.theta1*aa[n]+est.eta12*aa[n-11]+est.theta1*est.eta1

log.z.for[n+2]=log.z.for[n+1]+log.z[n-10]-log.z[n-11]+est.eta12*aa[n-10]+est.theta1*est.eta12*aa[n-11]

log.z.for[n+3]=log.z.for[n+2]+log.z[n-9]-log.z[n-10]+est.eta12*aa[n-9]+est.theta1*est.eta12*aa[n-10]

log.z.for[n+4]=log.z.for[n+3]+log.z[n-8]-log.z[n-9]+est.eta12*aa[n-8]+est.theta1*est.eta12*aa[n-9]
log.z.for[n+5]=log.z.for[n+4]+log.z[n-7]-log.z[n-8]+est.eta12*aa[n-7]+est.theta1*est.eta12*aa[n-8]

log.z.for[n+6]=log.z.for[n+5]+log.z[n-6]-log.z[n-7]+est.eta12*aa[n-6]+est.theta1*est.eta12*aa[n-7]

log.z.for[n+7]=log.z.for[n+6]+log.z[n-5]-log.z[n-6]+est.eta12*aa[n-5]+est.theta1*est.eta12*aa[n-6]

log.z.for[n+8]=log.z.for[n+7]+log.z[n-4]-log.z[n-5]+est.eta12*aa[n-4]+est.theta1*est.eta12*aa[n-5]

log.z.for[n+9]=log.z.for[n+8]+log.z[n-3]-log.z[n-4]+est.eta12*aa[n-3]+est.theta1*est.eta12*aa[n-4]

log.z.for[n+10]=log.z.for[n+4]+log.z[n-7]-log.z[n-3]+est.eta12*aa[n-2]+est.theta1*est.eta12*aa[n-3]
```

```
val_x=c(0,(1:8)*50,nn)
val_y=c(-1.5,-0.1,0,0.75,1.5)
plot(log.z,main="log(Electricity)",xlab="months",ylab="MWh",
xlim=c(0,400),xaxt="n",yaxt="n",type="p",lwd=2)
axis(1,val_x,val_x)
axis(2,val_y,val_y)
lines(log.z.fit,col="red",lwd=2)
lines(log.z.for,type="l",lwd=2,col="blue")
abline(v=n,col="black",lwd=3,lty=4)
```



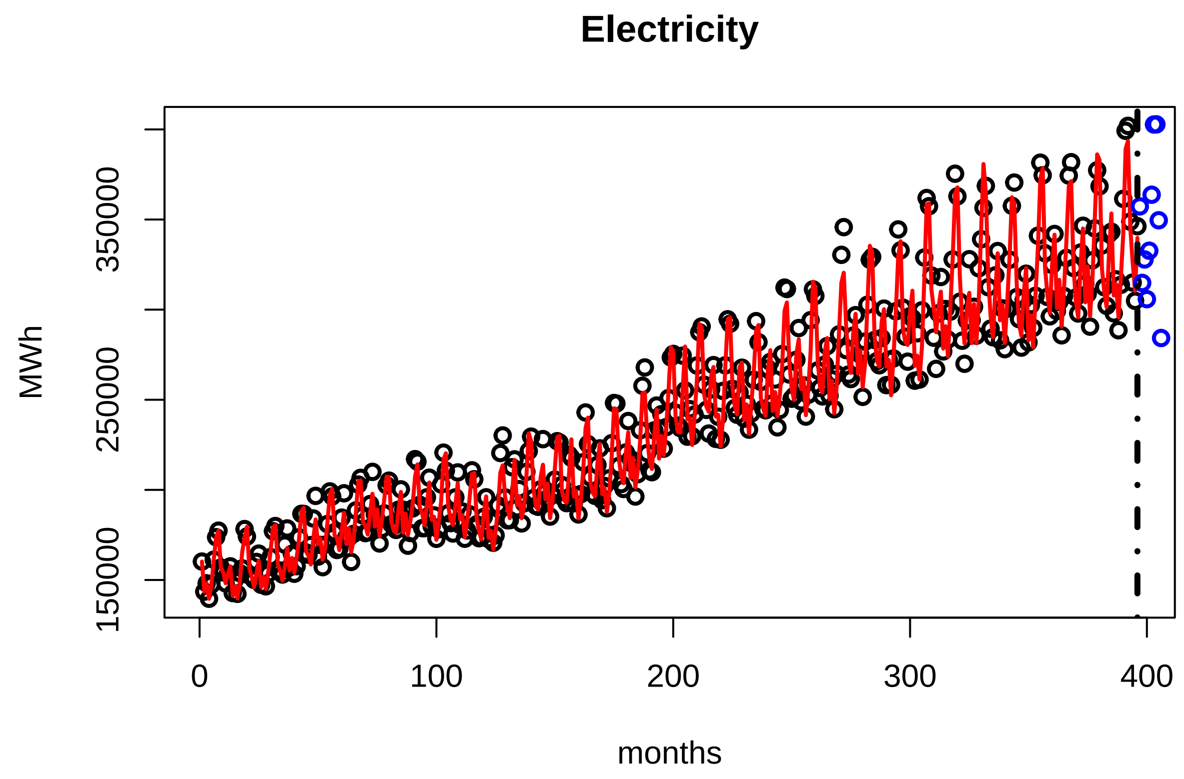
Now lets forecast by adjusting the mean to the actual mean

```
val_x=c(0,(1:9)*24,nn)
val_y=c(0:3)
plot(log.z+mean,main="log(Electricity)",xlab="months",ylab="MWh",type="p",lwd=2)

lines(log.z.fit+mean,col="red",lwd=2)
lines(log.z.for+mean,type="p",lwd=2,col="blue")
abline(v=n,col="black",lwd=3,lty=4)
```



Now lets remove the logarithmic data transformation.

```
val_x=c(0,(1:9)*24,nn)
val_y=val_y=c((0:4)*5)
plot(y,main="Electricity",xlab="months",ylab="MWh",type="p",lwd=2)

lines(exp(log.z.fit+mean),col="red",lwd=2)
lines(exp(log.z.for+mean),type="p",lwd=2,col="blue")
abline(v=n,col="black",lwd=3,lty=4)
```



The confidence interval for forecasting in a time series is computed using the forecasted values and their associated variance estimates. To compute a 95% confidence interval,

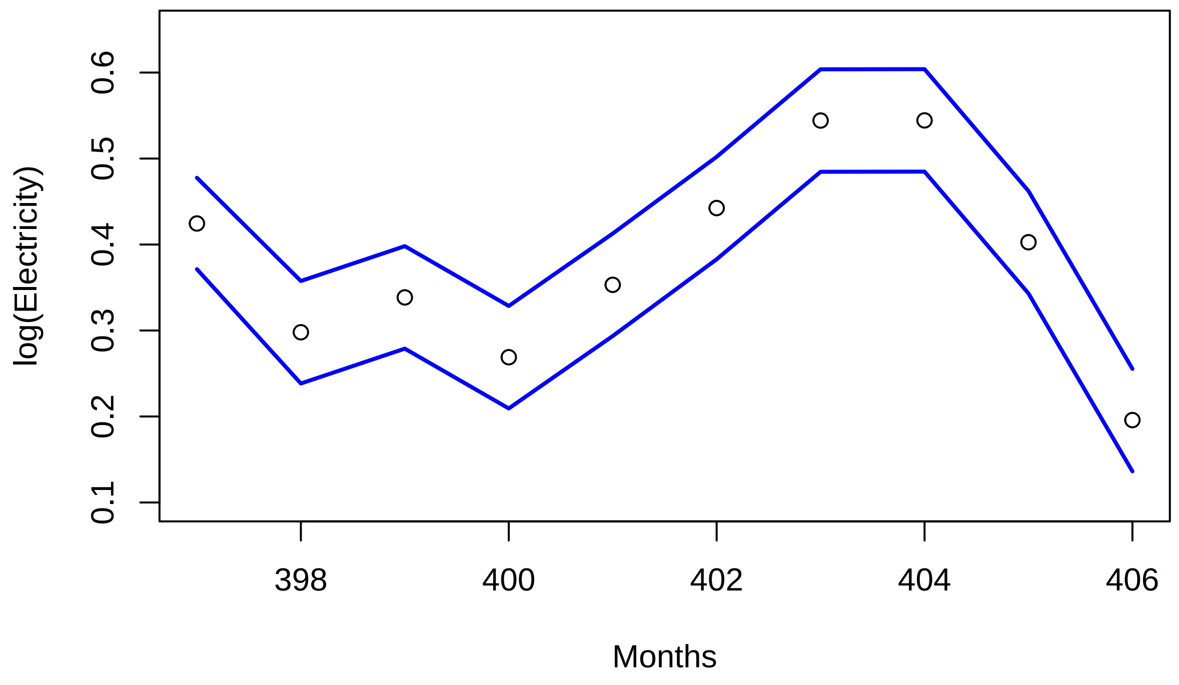
$$P(|Z| < 1.960) = 0.95 Lower\ Bound = Forecasted\ Value\text{-}1.96*\sqrt{EstimatedVariance} Upper\ Bound = Forecasted\ Value+1.9$$

```r
diff1.12.log.z.fit=rep(NA,times=n)

aa=rep(0,times=n+10)
```

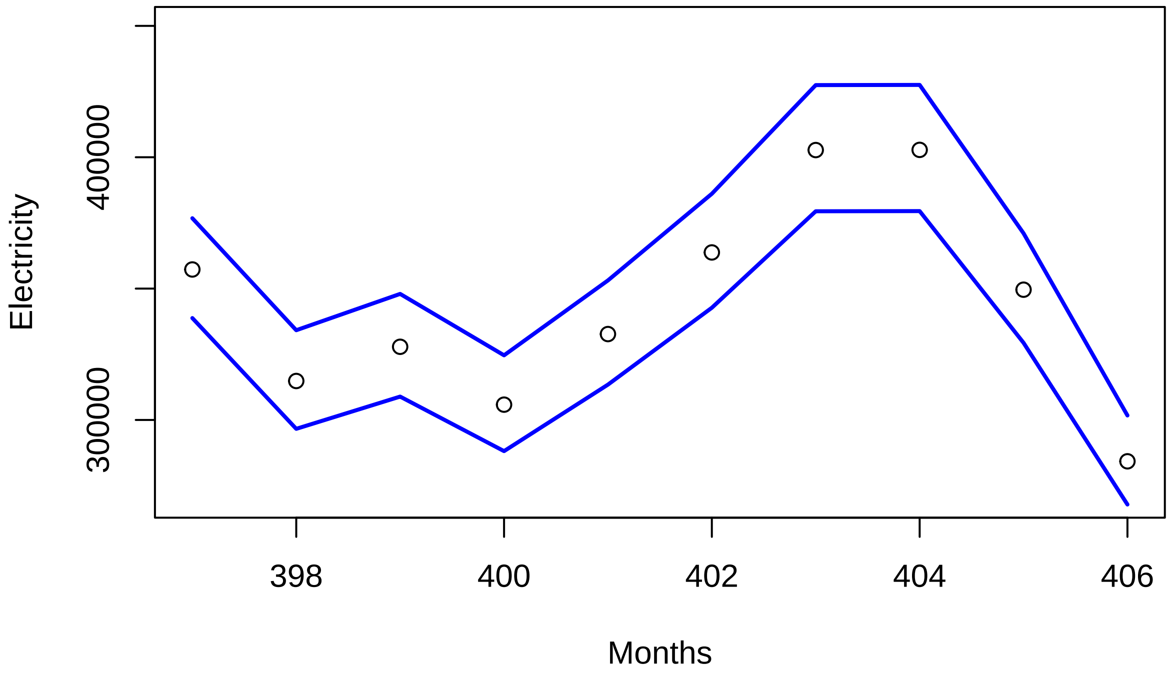
```r
psi=rep(0,times=10)

psi.0=1

psi[1]=est.theta1

# psi[2]=...=psi[10]=0

var.er=rep(0,times=10)

var.er[1]=est.sigma2

for (j in 2:10) {
    var.er[j]=est.sigma2*(psi.0+sum(psi[1:(j-1)]^2))
}
```

```r
# Left (lower bound)

left.er=rep(NA,times=nn)

for (t in (n+1):nn) {
    left.er[t]=log.z.for[t]-1.960*sqrt(var.er[t-n])

}
#P(|Z|<2.576)=0.99

# Right (upper bound)

right.er=rep(NA,times=nn)

for (t in (n+1):nn) {
    right.er[t]=log.z.for[t]+1.960*sqrt(var.er[t-n])
}
```

```r
plot(log.z.for,xlim=c(n+1,nn),ylim=c(0.1,0.65),type="p",xlab = "Months", ylab = "log(Electricity)")
lines(left.er,type="l",col="blue",lwd=2)
lines(right.er,type="l",col="blue",lwd=2)
```



```
plot(exp(log.z.for+mean),xlim=c(n+1,nn),ylim=c(270000,450000),type="p",xlab = "Months", ylab = "Electri
lines(exp(left.er+mean),type="l",col="blue",lwd=2)
lines(exp(right.er+mean),type="l",col="blue",lwd=2)
```



```
#
```

Conclusion

This detailed explanation gives an elaborate overview of the analysis of the "Electricity" dataset in R TSA library, covering different aspects such as its significance, characteristics, initial observations and steps taken for data transformation and analysis. The findings are effectively communicated by the explanation, including the presence of trends, seasonality and non-stationarity in the original dataset and how transformation can stabilize variance while achieving stationarity. Besides that, it also points out why identification of patterns and seasonality in time series data is crucial through techniques like Auto-correlation function (ACF) or partial autocorrelation function (PACF) analysis. In summary, this explanation offers some valuable insight into context and understanding regarding what this analysis means for forecasting and time series modelling.