# Time Series Analysis:
# Total Renewable Energy Consumption in Worldwide

Author: Zejie (Sandy) Gao
University of California, Santa Barbara
PSTAT 274: TIME SERIES
Instructor: Dr. Raya Feldman
Fall 2023

*Abstract*

This project applies time series analysis to forecast global renewable energy consumption, focusing on data from January 2002 to November 2022, sourced from Kaggle. I have used techniques such as autocovariance, autocorrelation, Box-Cox transformations, and spectral analysis. The chosen SARIMA(2, 1, 2) * (0, 1, 1)[12] model demonstrates reliable forecast precision, performing well with the observed data. These findings are instrumental for policymakers and investors in the tracing the usage of sustainable energy resources.

## Introduction

This project focuses on forecasting global renewable energy consumption using a Kaggle data set spanning from January 2002 to November 2022, comprising 250 monthly observations. I am interest in this field because of the influential role of sustainable energy in the global energy landscape. Considering the scarcity of fossil fuels continuing to be a crucial problem, the utilization of renewable energy become the future choice for humans.

For enhanced model reliability, the data was split into training and test sets. The first 228 observations, covering 19 years, formed the training set for modeling, while the last 22 observations, representing the final two years, served as the test set.

The initial time series analysis of the training set revealed a trend and clear seasonality that is not stationary. To achieve stationarity, a key step was transforming the data including Box-Cox transformation and differencing techniques.

During the modeling phase, twelve candidate models were initially considered. The selection was based on AICc values and detailed parameter estimations. The top two models were further analyzed, including diagnostic checks for stationarity, invertibility, and residual analysis. Spectral analysis helped refine the model selection by identifying various oscillatory behaviors in the data.

The final step was validating the chosen model against the test set. Despite a slight underestimation tendency, the model's forecasts closely matched the observed data's confidence interval, indicating high precision and reliability. This makes the model an effective tool for forecasting future renewable energy trends.

The findings of this study provide valuable insights for strategic planning and policy development in the energy sector. In addition, the data set was sourced from Kaggle's World Energy Statistics (https://www.kaggle.com/datasets/akhiljethwa/world-energy-statistics), and the analysis was performed using the R programming language.

## Data Description

Originally, the data was collected from the U.S. Energy Information Administration. It encompasses comprehensive world energy statistics from 1973 to 2022, dealing monthly value of energy production, consumption, imports, exports with various energy sources including renewable energy, nuclear energy, and fossil fuels. All numbers from data are measured in Quadrillion British Thermal Units.

For my analysis, I focus specifically on a subset of this extensive data set, concentrating on Total Renewable Energy Consumption (TREC) over the most recent 21 years. This select includes data from January 2002 to November 2022, providing a focused view on the recent trends in renewable energy usage. The series I am examining comprises 250 monthly observations, each recording the total global consumption of renewable energy.

From the figure 1, we see that this series exhibits the three obvious characteristics. The plot analysis reveals three key characteristics of the time series. Firstly, a seasonal pattern is evident, with fluctuation from the consumption curve. Secondly, there's a consistent upward trend, indicating a growing global need for renewable energy over the past two decades. Lastly, the data shows relatively stable over time.

These characteristics suggest a non-stationary nature of the time series, which will be explained in greater detail in the next section of the research.

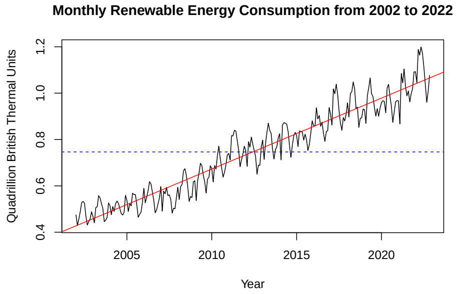

Figure 1: Monthly renewable energy consumption from year 2002 to 2022

# Methodology

## 1. Data Division

For better assess my model, I utilized a data division by splitting the data set into training and tests sets. The training set is used for modeling, while the tests set is reserved for validation. Given the limited size of the data, a larger proportion was allocated to the training set to ensure reliable performance. Specifically, rows 1 to 228, corresponding to the first 19 years, were used for training. The remaining rows, from 229 to 250, representing the last two years, were designated as the test data. This results in 228 observations in the train set and 22 in the test data.

## 2. Time Series Analysis (Training Set)

Based on the analysis of Figures 2 and 3, the training data displays a significant upward trend, as indicated by the red trend line, and a repeating annual pattern suggestive of seasonality. The mean, represented by the blue line, confirms the presence of an increasing trend. The seasonal decomposition shows a consistent upward trend and a clear seasonal pattern with predictable fluctuations. There are no abrupt changes in the data, and the variance does not display noticeable changes over time.

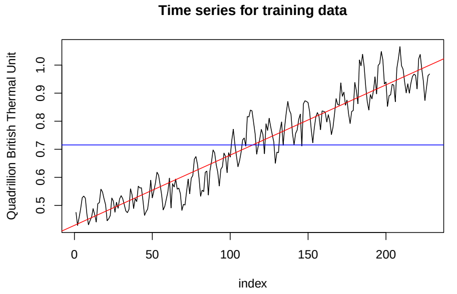

Figure 2: Graph of time series for training set

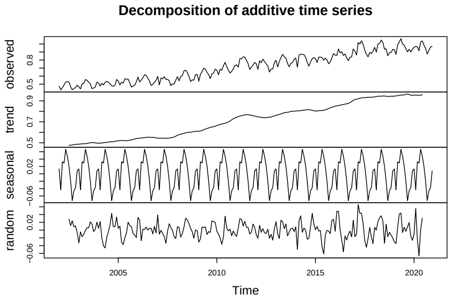

Figure 3: Graph of Decomposition of time series

Despite a relative stable variance, a Box-Cox transformation will be assessed for potential variance stabilization and normalize the data. The PACF plots (figure 4) confirm seasonality at lag 12, 24.., suggesting the need for seasonal differencing or transformation to prepare the data for accurate modeling and forecasting.

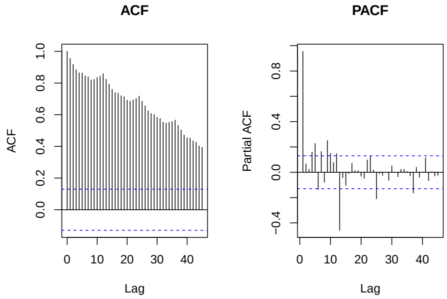

Figure 4: Graph of ACF and PACF on time series for training set

## 3. Transformation to Stationarity

Stabilizing the Variance and Removing Trend & Seasonality:

In the transformation process, I applied a Box-Cox transformation with a lambda of 0.505, effectively stabilizing the variance and normalizing the data distribution. Figure 7's histogram of the transformed data seems not have very large change, and Figure 6's time plot also hard to see the change of variance by eyes. But its ok since our dash line for confidence interval from figure 5 not including 0 or 1 validates the choice of transformation, moving away from standard log or no transformation.

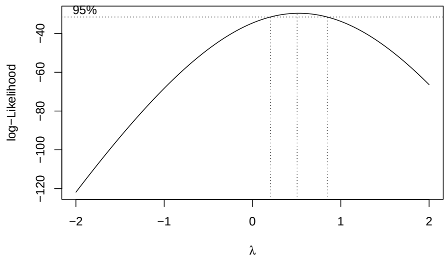

Figure 5: Lambda plot with log-like method

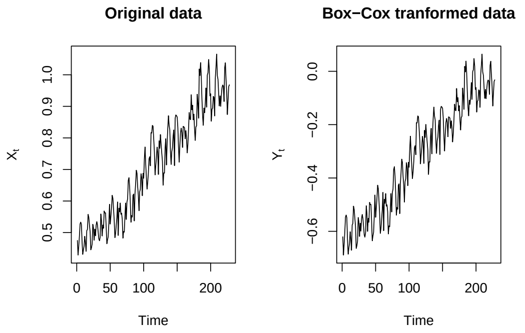

Figure 6: Graph of time series for two data

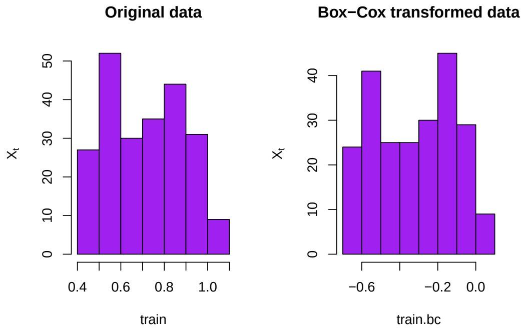

Figure 7: Histogram of two data

Seasonality was still evident in the transformed data, as shown by a cyclical pattern in the PACF plot (Figure 8), with significant correlations every 12 lags, indicating a yearly cycle. Differencing at lag 12 (Figure 9) looks reduced this seasonality, confirmed by the lowered variance (0.002086514).

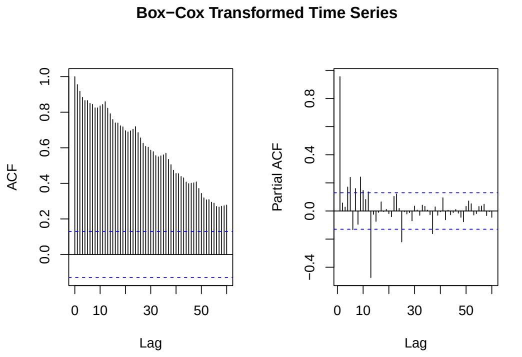

Figure 8: ACF and PCF after Box-Cox transformation

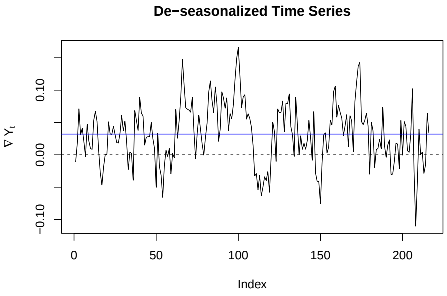

Figure 9: Time series graph after differing at lag 12 on train.bc

```r
var(train.bc)
```

```
## [1] 0.04267634
```

```r
var(train.bc.diff12)
```

```
## [1] 0.002086514
```

To address any trend, I applied additional differencing at lag 1 to the seasonally adjusted data. Despite the visual absence of a red trend from De-seasonalized Times Series (Figure 9), the further reduction in variance from tibble 0.001471763 indicates the need for this step.

```
## # A tibble: 3 x 2
##   Parameter                Value
##   <chr>                    <dbl>
## 1 var(train.bc)          0.0427
## 2 var(train.bc.diff12)   0.00209
## 3 var(train.bc.diff1.12) 0.00147
```

Finally, Figure 10 shows the de-trended and de-seasonalized time series, now with no evident trend or seasonal pattern and stable variance, achieving stationary necessary for further time series analysis.

## 4. Model Identification

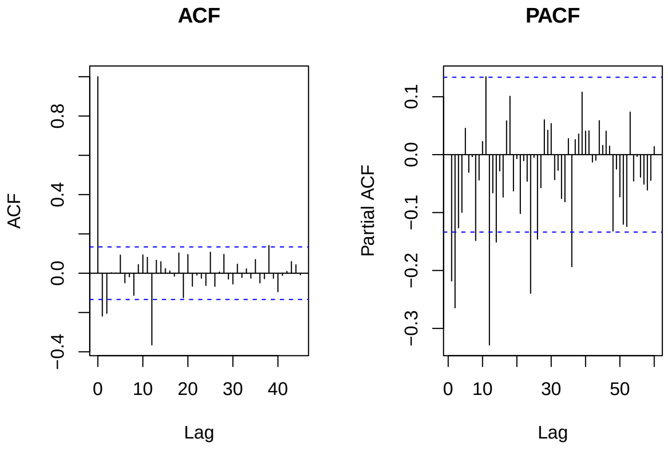

Figure 11: ACF and PCF plot on the de-trended/seasonalized train.bc

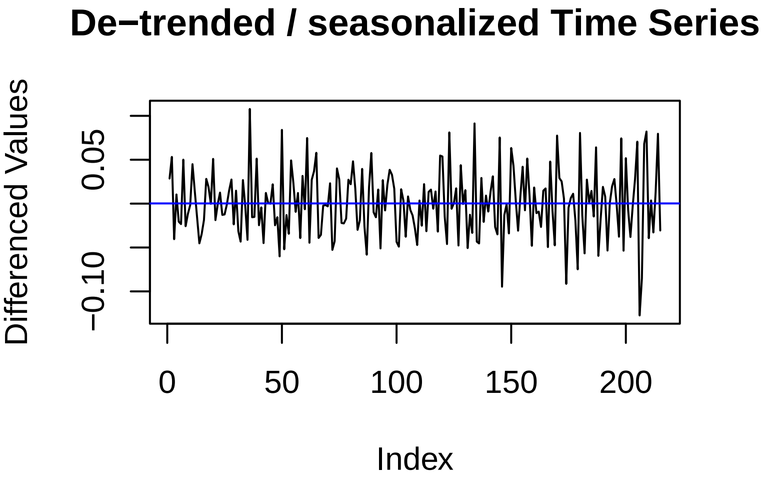

Figure 10: Time series graph after differing at lag 1&12 on train.bc

Based on figure 11, I could find possible seasonal ARIMA models to fit. It is written as follows:

$$SARIMA : (p, d, q) * (P, D, Q)s$$

where (p, d, q) is the non-seasonal part of the model and (P, D, Q) is the seasonal part of the model.

Seasonal Components:

For this part, mainly focus on the seasonal lags h = 1s, 2s, etc.

- One seasonal differencing (D = 1) at lag (s = 12) has been applied, implying the removal of the seasonal effect once per year.
- The ACF indicates a pronounced peak at the first seasonal lag, suggesting a seasonal MA component of (Q = 1).
- Strong peaks in the PACF at the first three seasonal lags suggest potential seasonal AR components, with P being 1, 2, or 3.

Non-Seasonal Components:

In this case focus on the within season lags, h = 1, . . . ,11.

- A single non-seasonal differencing (d = 1) was employed to eliminate any trend, aligning with the evidence from the ACF and PACF.
- Significant spikes in the ACF at h = 1, and h = 2 and cut off after 2, indicate a non-seasonal MA component with possible values of (q = 0) or (q = 2).
- The PACF reveals significant correlations at lags 1 and 2, while lags 8 and 11 only marginally touch the confidence interval. Given their proximity to the threshold, lags 8 and 11 are not considered robust enough for inclusion in the model. Therefore, the non-seasonal AR component is selected as (p = 0) or (p = 2).

### 5. Model Fitting

As an illustration above, I fit the following twelve models:

- fit_1: SARIMA (0,1,0) × (1,1,1) s=12
- fit_2: SARIMA (0,1,0) × (2,1,1) s=12
- fit_3: SARIMA (0,1,0) × (3,1,1) s=12
- fit_4: SARIMA (0,1,2) × (1,1,1) s=12
- fit_5: SARIMA (0,1,2) × (2,1,1) s=12
- fit_6: SARIMA (0,1,2) × (3,1,1) s=12
- fit_7: SARIMA (2,1,0) × (1,1,1) s=12
- fit_8: SARIMA (2,1,0) × (2,1,1) s=12
- fit_9: SARIMA (2,1,0) × (3,1,1) s=12
- fit_10: SARIMA (2,1,2) × (1,1,1) s=12
- fit_11: SARIMA (2,1,2) × (2,1,1) s=12
- fit_12: SARIMA (2,1,2) × (3,1,1) s=12

Selection of Models Based on AICc:

```
AICc <- c(fit_1$AICc,fit_2$AICc,fit_3$AICc,fit_4$AICc,fit_5$AICc,fit_6$AICc,fit_7$AICc,fit_8$AICc,fit
AICc
```

```
##  [1] -4.031917 -4.022369 -4.015759 -4.172806 -4.163533 -4.155130 -4.134873
##  [8] -4.125651 -4.117861 -4.165768 -4.160298 -4.147464
```

I identified the AICc (Akaike Information Criterion) values for each fitted SARIMA models. The criterion guided me in comparing the models effectively by balancing the model fit with the complexity. The models with the lower AICc values are preferred as they are indicative of a better fit. From this comparative analysis, the models that yielded the smaller AICc values and hence were selected for further evaluation and parameter estimation steps are as follows:

- fit_4 with an AICc of -4.172806,

- fit_10 with an AICc of -4.165768,

- fit_5 with an AICc of -4.163533,

- fit_11 with an AICc of -4.160298,

- fit_6 with an AICc of -4.155130.

Parameter estimation:

After fitting a SARIMA model, the next step is to check the significance of the coefficients. This is done by examining if their confidence intervals include zero. In the examples provided, certain coefficients are deemed not significant (confidence interval contains zero) and are set to zero in the revised model. Lastly, make an decision on whether use new model by comparing AICc of orignal and new. Choose the one with smaller AICc.

Since it would be too long to write down all checking coefficient step for all 5 models, I will take `fit_4` SARIMA model as an exemplar to showcase the procedure on parameter estimation and the rest of the model will follow similar steps.

In evaluating the `fit_4` SARIMA model, only SAR1 coefficient in `fit_4` has confidence interval that include zero (0.1286 ± 2*0.0912), suggesting it's not significant.

Based on this, a revised model, `fit.4_new`, was created with SAR1 set to zero. Comparing AICc values of both models showed a slight improvement in `fit.4_new` (from -4.172806 to -4.173044), leading to its selection as the preferred model.

```
fit_4$ttable
```

```
##      Estimate     SE t.value p.value
## ma1   -0.3139 0.0677 -4.6383  0.0000
## ma2   -0.3013 0.0759 -3.9717  0.0001
## sar1   0.1286 0.0912  1.4097  0.1601
## sma1  -0.9170 0.0922 -9.9503  0.0000
```

```
fit.4_new <- sarima(train.bc, p = 0, d = 1, q = 2, P = 1, D = 1, Q = 1, S = 12,
                    details = FALSE,
                    fixed = c(NA, NA, 0, NA))
```

```
fit_4$AICc
```

```
## [1] -4.172806
```

```
fit.4_new$AICc
```

```
## [1] -4.173044
```

```
AICc_new <- c(fit.4_new$AICc,fit_10$AICc,fit.5_new$AICc,fit.11_new$AICc,fit.6_new$AICc)
AICc_new
```

```
## [1] -4.173044 -4.165768 -4.173044 -4.173827 -4.173044
```

After completing the parameter estimation and model comparison for a range of SARIMA models, two have stood out based on their Akaike Information Criterion corrected (AICc) values:

Model A: `fit.11_new` with AICc = -4.173827, SARIMA(2, 1, 1) * (0, 1, 1)[12]

Model B: `fit.4_new` with AICc = -4.173044, SARIMA(0, 1, 2) * (0, 1, 1)[12]

These models, now identified as the final candidates, will be subjected to the next phase of diagnostic checks to validate their fit and effectiveness in capturing the time series' underlying patterns.

## 6. Diagnostic Checking

### 6.1 Check the model stationarity/invertibility

- **Model A: SARIMA(2, 1, 2) * (0, 1, 1)[12]** ,`fit.11_new` from train_bc (after box-cox transformation)

$$(1+0.3058B-0.2873B^2)(1-B)(1-B^{12})Xt=(1-0.6689B^2)(1-0.85211B^{12})Zt$$

In this model, the seasonal Autoregressive (SAR) part lacks of coefficent. For the Autoregressive (AR) part, the roots, indicated by blue dots in the plot, are found inside the unit circle, signifying that the actual roots (in red) lie outside the unit circle, ensuring the model's stationary. The figure 12 visually confirms this. Conversely, the SMA part, with a coefficient of 0.85211, suggests that its roots are also outside the unit circle, contributing to the model's invertibility. Similarly, for the moving average (MA) part, the roots lie outside the unit circle, as shown in the figure 13.

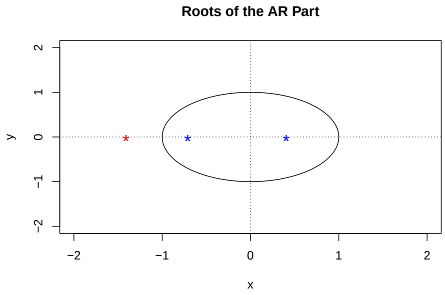

Figure 12: Roots of the AR Part

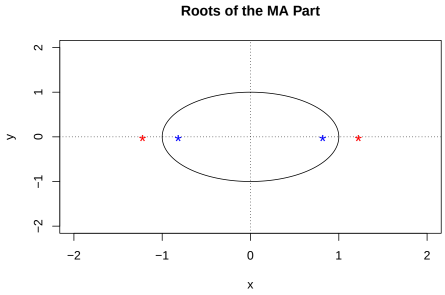

Figure 13: Roots of the MR Part

- **Model B: SARIMA(0, 1, 2) × (0, 1, 1)[12]**, `fit.4_new` from train_bc (after box-cox transformation)

$$(1 - B)(1 - B12)Xt = (1 - 0.3146B - 0.2969B^2)(1 - 0.8515B^{12})Zt$$

As a pure MA model, it is alway stationary. Thus, I will focus on checking invertibility. The roots of the MA part are analyzed through the figure 14, indicating the roots are outside. Then, it confirms the model's invertibility.

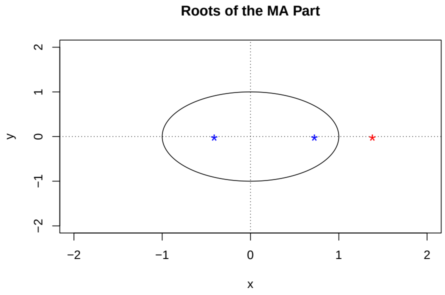

Figure 14: Roots of the MR Part

These analyses of both Model A and Model B provide insights into their stationary and invertibility and ready for the next residual diagnostic.

### 6.2 Residual diagnostic:

**Model A:** To check if the residuals of model A follow White Noise distribution, I perform several diagnostic tools in this section. Firstly, checking normality assumptions and get the plots below.

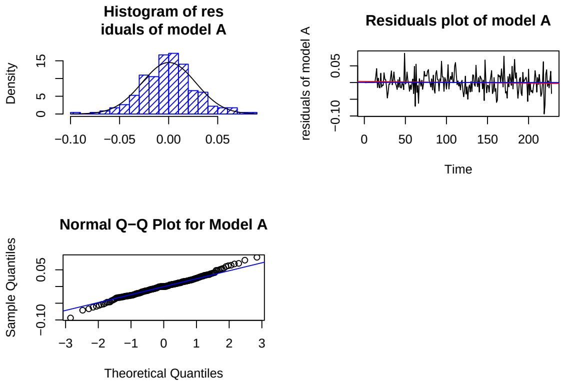

Figure 15: More graph about Normality check for model A

The analysis of Model A's residuals suggests they exhibit characteristics of white noise, having a constant mean and relatively stable variance. The use of a histogram, QQ-plot, and the Shapiro-Wilk Normality Test, which yielded a value of 0.1401202, supports the hypothesis that Model A's residuals are normally distributed.

```
## # A tibble: 4 x 2
##   Method                         p_value
##   <chr>                            <dbl>
## 1 Shapiro-Wilk Normality Test      0.140
## 2 Box-Pierce Test                  0.698
## 3 Ljung-Box Test                   0.664
## 4 McLeod-Li Test                   0.179
```

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

```
##
## Call:
## ar(x = resA, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  0.0007546
```

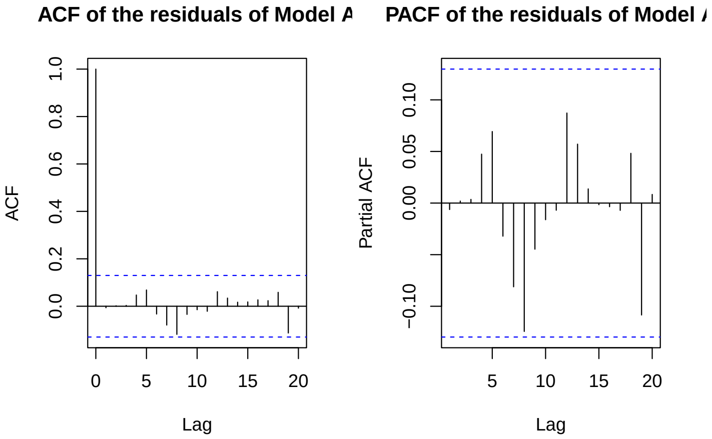

Figure 16: ACF and PACF for model A's residual

Additionally, the results from the Box-Pierce Test (0.6980155), Ljung-Box Test (0.6644465), and McLeod-Li Test (0.1788162) further reinforce this conclusion. Notably, the p-values for all these tests are greater than 0.05, indicating that the residuals pass these statistical tests lack of autocorrelation.

The lack of significant lags exceeding the confidence interval in both the Autocorrelation Function (ACF) and the Partial Autocorrelation Function (PACF) from figure 16 further confirms the adequacy of Model A. This comprehensive analysis points to Model A being a well-fitted model for the data.

**Model B:** To check if the residuals of model B follow White Noise distribution, I perform same diagnostic tools as model A did. Firstly, checking normality assumptions and get the plots below shows good normal distribution of Model B. P-values from alll tests were greater than 0.05, indicating that Model B successfully passes these tests. This suggests that the residuals are normally distributed, adhering to the assumptions of White Noise. While the Autocorrelation Function (ACF) and the Partial Autocorrelation Function (PACF) plots for Model B showed a slight deviation with one significant lag, this is not substantial to affect the White Noise assumption. Overall, the diagnostic tools indicate that Model B's residuals can be considered to follow a White Noise distribution.

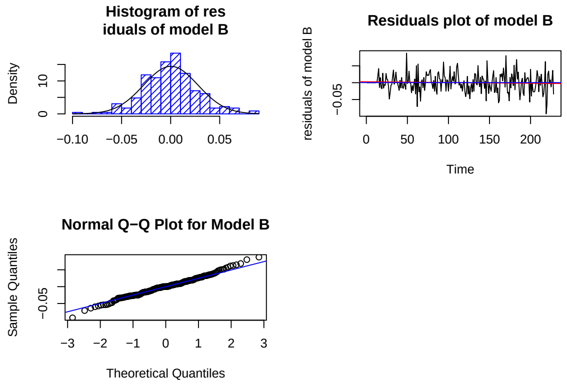

Figure 17: More graph about Normality check for model B

```
## # A tibble: 4 x 2
##   Method                          p_value
##   <chr>                             <dbl>
## 1 Shapiro-Wilk Normality Test   0.269
## 2 Box-Pierce Test                0.381
## 3 Ljung-Box Test                 0.339
## 4 McLeod-Li Test                 0.200
```

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

```
##
## Call:
## ar(x = resB, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  0.000764
```

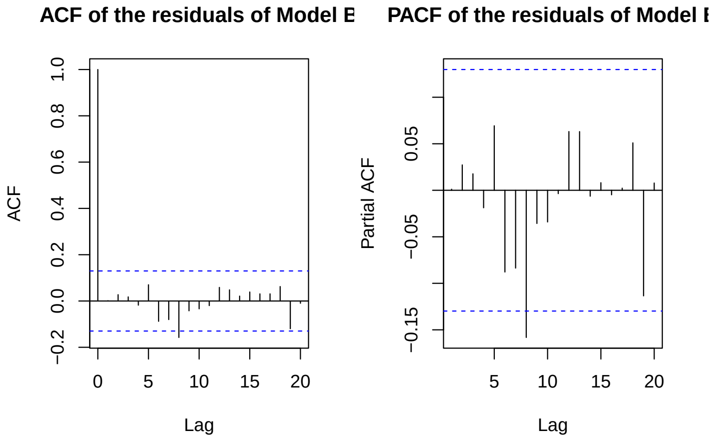

Figure 18: ACF and PACF for model B's residual

In the end, both model A and model B past the diagnostic. Since model A has less AICc, I decide to use model A for forecasting. Thus, my final model is SARIMA(2, 1, 2) * (0, 1, 1)[12], and it written as followed by:

$$(1 + 0.3058B - 0.2873B^2)(1 - B)(1 - B^{12})Xt = (1 - 0.6689B^2)(1 - 0.85211B^{12})Zt$$

## 7. Spectral Analysis

The periodogram of residuals for Model A in figure 19 looks like randomly located, meaning there are no clear patterns, which is good for model validity. Fisher's test with a high p-value of 0.8342183 also suggests the residuals don't have hidden patterns.

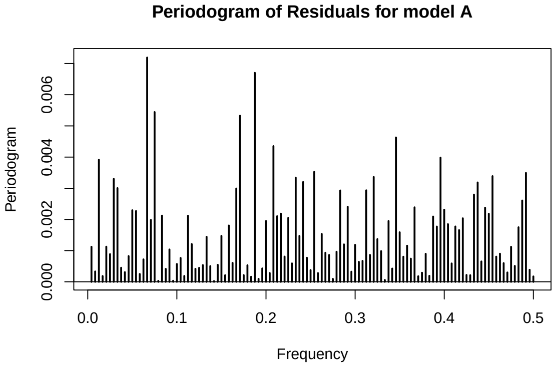

Figure 19: Periodogram of Residuals for Model A

Moreover, the Kolmogorov-Smirnov test's results in figure 20 are stay within the expected line, which means the residuals likely follow a normal distribution, supporting the model's adequacy.

```
## fisher.g.test p value: 0.8342183
```

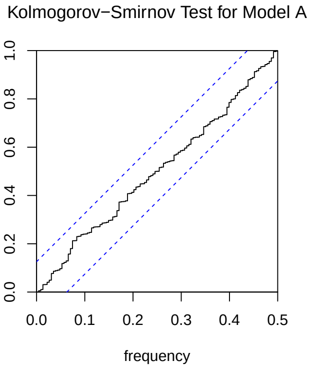

Figure 20: Kolmogorov-Smirnov Test for Model A

Overall, spectral analysis indicate that Model A's residuals are random, showing no patterns or cycles, which further suggests that the model fits the data well.

## 8. Forecasting

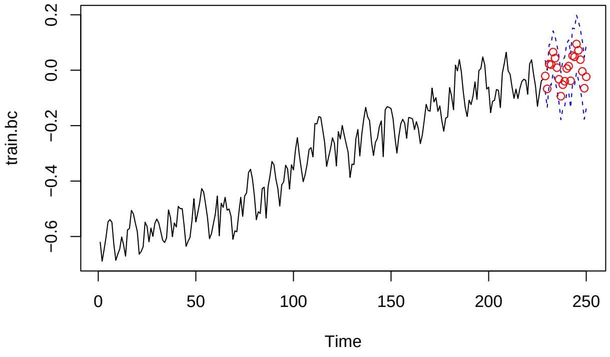

Figure 21: Forecast using the Box-Cox transformed data

The above figure is the forecast on the transformed data. The true values are within the confidence interval of the forecasting. For comparing with the true values of the last ten months, I convert the forecasting values back to the scale before box-cox transformation. This part shows how to convert the data back to the scale before box-cox transformation and compare the true values with predicted values.

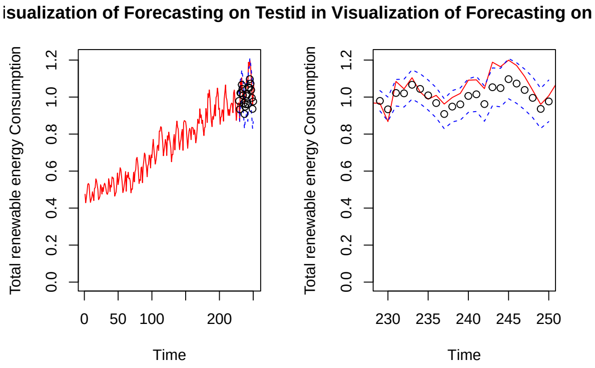

Figure 22: Visualization of Forecasting on Testing Set using original data

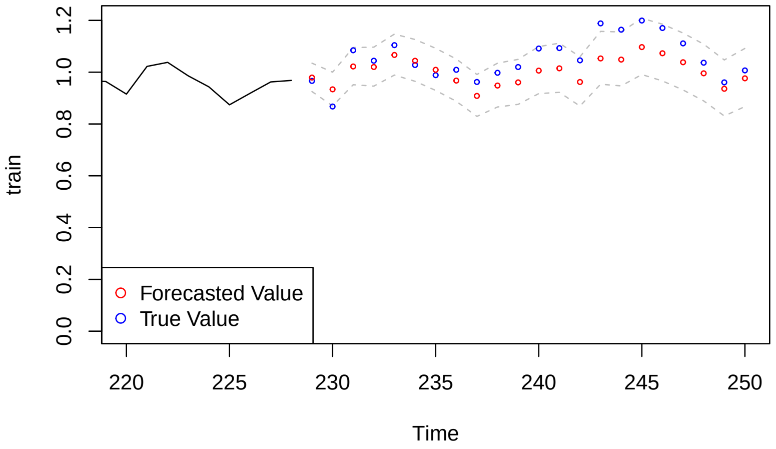

Figure 23: Zoomed plot including testing data and true value

The figure 22 display the forecast of total renewable energy consumption against actual data points. The left graph provides a broader view, showing the forecast in the context of the entire time series, while the right graph zooms in on the forecast horizon. The red line represents the forecasted values and the blue points are the actual, true values.

In the zoomed-in section from figure 23, we can see that the true values generally fall within the forecasted confidence intervals, indicating a reasonably accurate forecast. The dashed lines represent the upper and lower bounds of the confidence intervals, giving a sense of the potential variance in the predictions. It appear only on data outside the confident interval but is very close to CI. Additionally, here do have weakness in my data that it seems understimate my true value, but overall is fine since the majority of them fall inside the CI.

## Conclusion

In conclusion, the goals of the project were achieved by successfully applying time series techniques to model renewable energy consumption. The SARIMA(2,1,2)(0,1,1)[12] model, represented by the equation:

\$$(1 + 0.3058B - 0.2873B^2)(1 - B)(1 - B^{12})Xt = (1 - 0.6689B^2)(1 - 0.85211B^{12})Zt$\$.

The model performed well on the test set, with predicted values closely matching the actual observations, demonstrating its power. This was visually affirmed by the forecast plots where the true values mostly resided within the model's confidence intervals. Furthermore, spectral analysis corroborated the model's adequacy, showing no significant cycles or oscillations in the residuals, which suggests that the model's residuals are consistent with white noise.

I would like to begin by extending my sincere thanks to Dr. Raya Feldman for her guidance during office hours with model identification and suggestions on the format. Also, the lecture notes and lab are very well-organized and provide a super clear guideline for the project.I would also like to express my gratitude to Cosmin for his invaluable assistance at the project's outset, particularly in helping me grasp the appropriate data transformations. Finally, I would like to thank Assistant Lihao for providing additional office hours during the busy final week. His explanations were invaluable in interpreting the residual ACF and PACF, thereby solidifying my understanding of the model's diagnostics.

## References

lecture notes for week 1 to 9

labs from 1 to 7

lecture slice from week 1 to 9

https://www.kaggle.com/datasets/akhiljethwa/world-energy-statistics

## Appendix

```
# Load necessary libraries
rm(list=ls())
library(tidyverse)
library(lubridate)
library(forecast)
library(MASS)
library(astsa)
library(tsdl)
library(xts)
library(tibble)
library(ggplot2)

source("plot.roots.R", local=knitr::knit_global())
```

```
# Load the data set
energy <- read.csv("/Users/zejiegao/Desktop/PSTAT 274/final project/World Energy Overview.csv")

# Simplify the name of column
names(energy)[names(energy) == "Total.Renewable.Energy.Consumption"] <- "trec"

# Now select the Date and trec columns
renew_c <- energy[, c("Date", "trec")]

# Filter the data to include only rows after 2001-12-31
renew_c <- renew_c[renew_c$Date > as.Date("2001-12-31") & renew_c$trec < as.Date("2001-12-31"),]

# write to subset data to a new CVS file
write.csv(renew_c, file = "/Users/zejiegao/Desktop/PSTAT 274/final project/TREC.csv", row.names = FAL
```

```
trec_ts <- ts(renew_c[,2],start = c(2002,1), frequency = 12)
# Plotting the time series
plot.ts(trec_ts,
        main = "Monthly Renewable Energy Consumption from 2002 to 2022",
        xlab = 'Year',
        ylab = "Quadrillion British Thermal Units",
        type = 'l',
        col = 'black')

# Adding a linear trend line
# Fitting a linear model to the time series
trend <- lm(trec_ts ~ time(trec_ts))

# Adding the linear trend line to the plot
```

```
abline(trend, col = "red")  # Trend line in red

# Adding a horizontal line representing the mean of the series
mean_line <- mean(trec_ts)
abline(h = mean_line, col = "blue", lty = 2)  # Mean line in blue with dashed style
```

```
# Divide data into training and test data
# Select rows 1 to 228 for training data, first 19 years
train <- trec_ts[1:228]

# Select rows 229 to 250 for test data, last 2 year
test <- trec_ts[c(229:250)]
```

```
# Plot time series
plot(1:length(train),train, main =
   "Time series for training data", type = 'l',xlab='index', ylab = "Quadrillion British Thermal Unit
index = 1: length(train)
trend <- lm(train ~ index)
abline(trend, col="red")
abline(h=mean(train), col="blue")
```

```
# Analyze the plot for trend, seasonality, and sharp changes
```

```
x <- renew_c[1:228,]
x_ts <- ts(x[,2],start = c(2002,1), frequency = 12)

decomposed_data <- decompose(x_ts)

# Plot the decomposed components
plot(decomposed_data)
```

```
op <- par(mfrow = c(1,2))  # Save the original par settings and set up for side by side plots.

# Plot ACF of training data for 45 lags
acf(train, lag.max = 45, main = "ACF")

# Plot PACF of training data for 45 lags
pacf(train, lag.max = 45, main = "PACF")
```

```
par(op)  # Reset to original par settings after plotting.
```

```
# Using Box-Cox transformation to stabalize variance
t = 1:length(train)
fit = lm(train ~ t)
bcTransform = boxcox(train ~ t,plotit = TRUE)
```

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

```
op <- par(mfrow = c(1,2))
ts.plot(train,main = "Original data",ylab = expression(X[t]))
ts.plot(train.bc,main = "Box-Cox tranformed data", ylab = expression(Y[t]))
```

```
par(op)
```

From the two plat above, the transformed data has a more stable variance across time.

```
op <- par(mfrow = c(1,2))
hist(train, col="purple",main = "Original data",ylab = expression(X[t]))
hist(train.bc, col="purple",main = "Box-Cox transformed data",ylab = expression(X[t]))
```

```
par(op)
```

Compared with the histogram of original data, the transformed data is more Gaussian. Therefore, the transformed data is more appropriate than the original.

```
op = par(mfrow = c(1,2))
acf(train.bc,lag.max = 60,main = "")
pacf(train.bc,lag.max = 60,main = "")
title("Box-Cox Transformed Time Series", line = -1, outer=TRUE)
```

```
par(op)
```

Notice the cyclical behavior in the ACF of the transformed data. Also, notice that there are significant correlations with values moving proportionally every 12 lags. Therefore, we can see that the period of the seasonal component is given by s = 12.

Remove seasonal components by differencing the transformed time series using the diff() function.

```
# Differencing at lag 12 to reduce seasonal pattern
train.bc.diff12 = diff(train.bc,12)
plot(train.bc.diff12,type = "l", main = "De-seasonalized Time Series", ylab = expression(nabla~Y[t]))
abline(h = 0,lty = 2)
abline(trend, col="red")
abline(h=mean(train.bc.diff12) , col='blue')
```

```
var(train.bc.diff12)
```

After differencing to remove seasonal component, we need to assess whether we need to difference to remove trend.

From the two plots above, the regression line is quite horizontal, meaning that we may not have trend.

```
# Differencing at lag 1 to eliminate trend
train.bc.diff1.12 = diff(train.bc.diff12, 1)

# Plot the differenced time series as a line graph.
plot(train.bc.diff1.12, type = "l",main = "De-trended/seasonalized Time Series", ylab = "Differenced
abline(h = mean(train.bc.diff1.12), col = "blue")
```

```
# Differencing at lag 1 to eliminate trend
train.bc.diff1.12 = diff(train.bc.diff12, 1)

# Plot the differenced time series as a line graph.
plot(train.bc.diff1.12, type = "l",main = "De-trended/seasonalized Time Series", ylab = "Differenced
abline(h = mean(train.bc.diff1.12), col = "blue")
```

```
data_tibble <- tibble(Parameter = c('var(train.bc)', 'var(train.bc.diff12)', 'var(train.bc.diff1.12)'
                      Value = c(var(train.bc), var(train.bc.diff12), var(train.bc.diff1.12)))
data_tibble
```

From the variance information, variance of the de-trended/seasonalized data achieves the lowest variance. Therefore, we do need to de-trend the data.

## 5. Model Identification

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

- We applied one seasonal differencing so D = 1 at lag s = 12.
- The ACF shows a strong peak at h = 1s.
  A good choice for the MA part could be Q=1
- The PACF shows one strong peaks at h = 1s.also for 2s,3s.
  A good choice for the AR part could be P = 1, or 2,3

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

- We applied one differencing to remove the trend: d = 1
- The ACF seem to be significant at 1. A good choice for the MA part could be q = 2 and 0
- a good choice for the AR part could be p = 0,2.

Since we difference seasonality and trend both once, d=D=1

```
op <- par(mfrow = c(1,2))

# Plot acf and pacf of new data
acf(train.bc.diff1.12 ,lag.max = 45, main = "ACF")
pacf(train.bc.diff1.12,lag.max = 60, main = "PACF")
```

```
# Identify potential models based on ACF/PACF plots
par(op)  # Reset to original par settings after plotting.
```

```
# Create the histogram
hist_data <- hist(train.bc.diff1.12, col = "purple", freq = FALSE,
                  main = "Histogram with Normalized Curve",
                  xlab = "Differenced Values", xlim = c(-0.1, 0.2))

# Calculate the mean and standard deviation of the data
mean_val <- mean(train.bc.diff1.12)
sd_val <- sd(train.bc.diff1.12)

# Create a sequence of x values that covers the range of your data
x_values <- seq(min(train.bc.diff1.12), max(train.bc.diff1.12), length.out = 100)

# Calculate the density of the normal distribution with the same mean and sd as your data
normal_density <- dnorm(x_values, mean = mean_val, sd = sd_val)

# Overlay the normal curve on the histogram
lines(x_values, normal_density, col = "seagreen3", lwd = 2)
```

## 6. Model Fitting

```
# SARIMA(0,1,0)(1,1,1)[12]
fit_1 <- sarima(train.bc, p = 0, d = 1, q = 0, P = 1, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(0,1,0)(2,1,1)[12]
fit_2 <- sarima(train.bc, p = 0, d = 1, q = 0, P = 2, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(0,1,0)(3,1,1)[12]
fit_3 <- sarima(train.bc, p = 0, d = 1, q = 0, P = 3, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(0,1,2)(1,1,1)[12]
fit_4 <- sarima(train.bc, p = 0, d = 1, q = 2, P = 1, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(0,1,2)(2,1,1)[12]
fit_5 <- sarima(train.bc, p = 0, d = 1, q = 2, P = 2, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(0,1,2)(3,1,1)[12]
fit_6 <- sarima(train.bc, p = 0, d = 1, q = 2, P = 3, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(2,1,0)(1,1,1)[12]
fit_7 <- sarima(train.bc, p = 2, d = 1, q = 0, P = 1, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(2,1,0)(2,1,1)[12]
fit_8 <- sarima(train.bc, p = 2, d = 1, q = 0, P = 2, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(2,1,0)(3,1,1)[12]
fit_9 <- sarima(train.bc, p = 2, d = 1, q = 0, P = 3, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(2,1,2)(1,1,1)[12]
fit_10 <- sarima(train.bc, p = 2, d = 1, q = 2, P = 1, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(2,1,2)(2,1,1)[12]
fit_11 <- sarima(train.bc, p = 2, d = 1, q = 2, P = 2, D = 1, Q = 1, S = 12, details = FALSE)

# SARIMA(2,1,2)(3,1,1)[12]
fit_12 <- sarima(train.bc, p = 2, d = 1, q = 2, P = 3, D = 1, Q = 1, S = 12, details = FALSE)
```

```
AICc <- c(fit_1$AICc,fit_2$AICc,fit_3$AICc,fit_4$AICc,fit_5$AICc,fit_6$AICc,fit_7$AICc,fit_8$AICc,fit
AICc
# choose top 5 lower AICc models
# fit_4 (-4.172806)
# fit_10 (-4.165768)
# fit_5 (-4.163533)
# fit_11 (-4.160298)
# fit_6 (-4.155130)
```

Parameter Estimation

```
fit_4 # the 95% confidence intervals of sar1 include 0
fit.4_new <- sarima(train.bc, p = 0, d = 1, q = 2, P = 1, D = 1, Q = 1, S = 12,
                    details = FALSE,
                    fixed = c(NA, NA, 0, NA))
fit.4_new
```

```
fit_4$AICc # -4.172806
fit.4_new$AICc # -4.173044
# AICc decrease, so accept fit.4_new
```

```
fit_10 # the 95% confidence intervals of sar1,sar2 include 0
fit.10_new=sarima(xdata = train.bc, p = 2, d = 1, q = 0, P = 3, D = 1, Q = 1, S = 12, details=F, fixe
fit.10_new
fit_10$AICc # -4.165768
fit.10_new$AICc # -4.130981
# AICc increase, so not accept fit.10_new
```

```
fit_5 # the 95% confidence intervals of sar1,sar2 include 0
fit.5_new=sarima(xdata = train.bc, p = 0, d = 1, q = 2, P = 2, D = 1, Q = 1, S = 12, details=F, fixed
fit.5_new
fit_5$AICc # --4.163533
fit.5_new$AICc # -4.173044
# AICc decrease, so accept fit.5_new
```

```
fit_11 # the 95% confidence intervals of sar1,sar2,sar3 include 0
fit.11_new=sarima(xdata = train.bc, p = 2, d = 1, q = 2, P = 2, D = 1, Q = 1, S = 12, details=F, fixe
fit.11_new
fit_11$AICc # -4.160298
fit.11_new$AICc # -4.173827
# AICc decrease, so accept fit.11_new
```

```
fit_6 # the 95% confidence intervals of sar1,sar2,sar3 include 0
fit.6_new=sarima(xdata = train.bc, p = 0, d = 1, q = 2, P = 3, D = 1, Q = 1, S = 12, details=F, fixed
fit.6_new
fit_6$AICc # -4.15513
fit.6_new$AICc # -4.173044
# AICc decrease, so accept fit.6_new
```

```
AICc_new <- c(fit.4_new$AICc,fit_10$AICc,fit.5_new$AICc,fit.11_new$AICc,fit.6_new$AICc)
AICc_new
```

fit.11_new and (fit.4_new,fit.5_new,fit.6_new) have smallest AICc values. We will perform diagnostic check for these two models.fit.11_new and fit.4_new. We have two candidate model SARIMA(2, 1, 1) x (0, 1, 1)12 or SARIMA(0, 1, 2) x (0, 1, 1)12.

# 7. Model Diagnostics

## Model stationarity and invertibility.

model A: SARIMA(2, 1, 2) x (0, 1, 1)12, from train_bc (after box-cox transformation)

$$(1 + 0.3058B - 0.2873B^2)(1 - B)(1 - B^{12})Xt = (1 - 0.6689B^2)(1 - 0.85211B^{12})Zt$$

```
plot.roots(NULL,polyroot(c(1,0.3058,-0.2873)), main="Roots of the AR Part")
```

```
plot.roots(NULL,polyroot(c(1,0,-0.6689)), main="Roots of the MA Part")
```

model B : SARIMA(0, 1, 2) × (0, 1, 1)12

$$(1 - B)(1 - B12)Xt = (1 - 0.3146B - 0.2969B^2)(1 - 0.8515B^{12})Zt$$

```
plot.roots(NULL,polyroot(c(1,-0.3146,-0.2969)), main="Roots of the MA Part")
```

## Diagnostic check on residual

To check if the residuals of model A follow White Noise distribution, we perform several diagnostic tools in this section. We first check normality assumptions and get the plots below.

```
plot.ts(residuals(fit.11_new$fit))
```

```
resA <- residuals(fit.11_new$fit)
par(mfrow=c(2,2))
hist(resA,density=20,breaks=20, col="blue", xlab="", prob=TRUE,main="Histogram of res
iduals of model A")
m <- mean(resA)
std <- sqrt(var(resA))
curve( dnorm(x,m,std), add=TRUE )
plot.ts(resA,ylab= "residuals of model A",main="Residuals plot of model A")
fitt <- lm(resA ~ as.numeric(1:length(resA)))
abline(fitt, col="red")
abline(h=mean(resA), col="blue")
qqnorm(resA,main= "Normal Q-Q Plot for Model A")
qqline(resA,col="blue")
```

```
h = round(sqrt(228))
h # the lag we will use in the later tests

# Shapiro test for normality
shapiro.test(resA)

#Box-Pierce test
Box.test(resA, type = c("Box-Pierce"), lag = h, fitdf = 4)

#Ljung-Box test
Box.test(resA, type = c("Ljung-Box"), lag = h, fitdf = 4)

#McLeod-Li test
Box.test(resA^2, type = c("Ljung-Box"), lag = h, fitdf= 0)
```

We can see that the p-values are larger than 0.05 for all the tests. Therefore, model B passes all tests.

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

```
par(mfrow=c(2,1))
acf(resA, lag.max=30,main="")
title("ACF of the residuals of Model A")
pacf(resA, lag.max=30,main="")
title("PACF of the residuals of Model A")
```

To check if the residuals of model B follow White Noise distribution, we perform several diagnostic tools in this section. We first check normality assumptions and get the plots below.

```
plot.ts(residuals(fit.4_new$fit))
```

```
resB <- residuals(fit.4_new$fit)
par(mfrow=c(2,2))
hist(resB,density=20,breaks=20, col="blue", xlab="", prob=TRUE,main="Histogram of res
iduals of model B")
m <- mean(resB)
std <- sqrt(var(resB))
curve( dnorm(x,m,std), add=TRUE )
plot.ts(resB,ylab= "residuals of model B",main="Residuals plot of model B")
fitt <- lm(resB ~ as.numeric(1:length(resB)))
abline(fitt, col="red")
abline(h=mean(resB), col="blue")
qqnorm(resB,main= "Normal Q-Q Plot for Model B")
qqline(resB,col="blue")
```

```
h = round(sqrt(228))
h # the lag we will use in the later tests

# Shapiro test for normality
shapiro.test(resB)

#Box-Pierce test
Box.test(resB, type = c("Box-Pierce"), lag = h, fitdf = 4)

#Ljung-Box test
Box.test(resB, type = c("Ljung-Box"), lag = h, fitdf = 4)

#McLeod-Li test
Box.test(resB^2, type = c("Ljung-Box"), lag = h, fitdf= 0)
```

We can see that the p-values are larger than 0.05 for all the tests. Therefore, model B passes all tests.

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

```
par(mfrow=c(2,1))
acf(resB, lag.max=30,main="")
title("ACF of the residuals of Model B")
pacf(resB, lag.max=30,main="")
title("PACF of the residuals of Model B")
```

since model A have less AICc anf it also pass all the test, we can use model A to forecast the total rennewable value. # 8.Forecasting

```
# SARIMA(2,1,2)(2,1,1)[12] for fit_11
model_A <- arima(train.bc, order=c(2,1,2), method="ML", seasonal = list(order = c(2,1,1), period = 12
# SARIMA(0,1,2)(1,1,1)[12] for fit_4
model_B <- arima(train.bc, order=c(0,1,2), method="ML", seasonal = list(order = c(1,1,1), period = 12

# Assuming model_A is a fitted Arima model
pred.trec <- forecast(model_A, h = length(test))

# Extracting upper and lower confidence intervals
U.tr <- pred.trec$upper[,2]  # Upper confidence interval
```

```r
L.tr <- pred.trec$lower[,2]  # Lower confidence interval

# Assuming 'train.bc' is your original time series data
ts.plot(train.bc, xlim=c(1, length(train.bc) + length(test)), ylim=c(min(train.bc, L.tr), max(train.b
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(train.bc) + 1):(length(train.bc) + length(test)), pred.trec$mean, col="red")
```

The above figure is the forecast on the transformed data. The true values are within the confidence interval of the forecasting. If we want to compare with the true values of the last ten months, we need to convert the forecasting values back to the scale before box-cox transformation. This part shows how to convert the data back to the scale before box-cox transformation and compare the true values with predicted values.

```r
# Correcting the inverse Box-Cox transformation
pred.orig <- (pred.trec$mean * lambda + 1)^(1/lambda)
U <- (U.tr * lambda + 1)^(1/lambda)
L <- (L.tr * lambda + 1)^(1/lambda)

# Original time series plot with forecasts
par(mfrow=c(2,1))

ts.plot(as.numeric(trec_ts), ylim = c(0, max(U)), col = "red", ylab = "Quadrillion British Thermal Un
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(train)+1):(length(train)+length(test)), pred.orig, col="black")
```

```r
ts.plot(as.numeric(trec_ts), xlim = c(229, length(train) + length(test)), ylim = c(0, max(U)), col="r
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(train)+1):(length(train)+length(test)), pred.orig, col="black")
```

```r
# Zoomed plot
ts.plot(train, xlim = c(220, length(train) + length(test)), ylim = c(0, max(U)))
lines(U, col="grey", lty="dashed")
lines(L, col="grey", lty="dashed")

# Ensure that 'test' and 'pred.original' have the same length and match the forecast horizon
points((length(train) + 1):(length(train) + length(test)), test, col="blue", cex=0.5)  # True Value
points((length(train) + 1):(length(train) + length(pred.orig)), pred.orig, col="red", cex=0.5)  # For

legend("bottomleft", pch = 1, col = c("red", "blue"), legend = c("Forecasted Value", "True Value"))
```

## 10. Spectral analysis

```r
# install.packages("TSA")
TSA::periodogram(resA, main="Periodogram of Residuals for Model A")
abline(h=0)
```

```r
library(GeneCycle)
fisher.g.test(resA)
cpgram(resA, main = expression("Kolmogorov-Smirnov Test for Model A"))
```