Electricity Sales in California: A Time Series Analysis Approach

Alison Wong, Jackson Sousa, Victor Lu

7th June 2023

Abstract

This time series analysis project aims to analyze and forecast electricity sales using historical data from 2010 to 2023. The project utilizes electricity sales data from the U.S Energy Information Administration (EIA) and aims to provide valuable insights for forecasting and making inferences. The analysis involves several key steps: time series decomposition, residual analysis, ARMA model fitting, and predicting future values. We accurately forecast values and find that there exists strong annual and semi-annual components in the data.

1 Introduction

The frequency of heatwaves in California has been increasing over the past two decades [4]. This trend not only applies to the frequency of heat waves but also to intensity and duration. These types of climate events are hazardous to California's populations, especially vulnerable populations such as children and the elderly. Hulley's paper specifically notes that inland urban areas, such as Los Angeles, have the highest increase in heatwave frequency, intensity, and duration. Negative consequences of heatwaves have recently been seen through blackouts ranging from cities like Los Angeles and Sacramento. Specifically, after the worst heatwave since 1985, California suffered its first large-scale power outage on August 14 to 15, 2020 [7]. This event, in part caused by a surge in energy demand, affected more than 420,000 people.

This paper seeks to address the issue of over-consumption and under-supply of energy, especially during times of high demand like summer and winter. We examine the total amount of megawatt-hours being sold per month within California. While it may seem reasonable to investigate this issue on a nationwide basis since many states rely on the same electrical grid, weather metrics vary greatly across regions as a result of complex atmosphere interactions [4]. By conducting a time series analysis, findings from this paper may aid California in ensuring that utility companies are better able to prepare and supply enough energy for the state during times of high demand in the future.

The ensuing parts of the paper describe the data set and focus on exploratory data analysis through time series methods.

1.1 Data Set Background

The data used for this project is from the EIA, an agency of the US Federal Statistical System that is responsible for collecting, analyzing, and spreading energy information. Specifically, the data set provided by the EIA is Form EIA-861M, which details figures from utility companies and power marketers measured monthly. EIA collects the data for this form from the following surveys: Form EIA-860, Annual Electric Generator Report, Form EIA-923, and Power Plant Operations Report. Data for all 50 states are included in the data set.

From this data set, we are given the amount of revenue per thousand dollars, sales of megawatt-hours, customer count, and price in cents/kWh. These figures are categorized into 5 sections: Residential, Commercial, Industrial, Transportation, and Total, all of which range from January 2020 through February 2023. Since the goal of this paper is to analyze California's total energy needs, we focus on the total amount of energy sold from the total category. Notably, since this data set reports energy sales, it does not accurately measure the energy generated and used by privately owned solar panels.

California as a state has been encouraging the purchase of solar panels through tax credits and requiring all homes built during and after 2020 to include solar panels, as noted by California's Title 24. These details may explain the slight downward trend in energy sales after 2018 which is seen later in the paper. However, it should be noted that extra unused power is fed back out into the power grid and subsequently sold, so these specific figures are captured by the data set.

2 Data Preparation

As mentioned previously, the data set provided by the EIA contains 5 different categories. However, we only seek to focus on total sales of megawatt-hours per month. So, before any analysis is done, we subset the raw data to filter only for this single column of interest. Since January 2010 through February 2023 is 158 months, we ensure that no data was lost by checking that the raw data has 158 data points for CA and that the filtered data has 158 data points too. However, since the data set only measures 2 months of data for 2023 and the data status for these months are "preliminary," we ignore these values for our analysis.

To ensure an accurate assessment of our model's accuracy and prediction errors, we have excluded the year 2022 from the dataset during the model fitting process. By doing so, we can utilize the actual values of the excluded year for comparison.

Lastly, since the magnitude of the megawatt-hours sold is 7 figures, we reduced these values by a factor of 10^6 so that our values are easier to interpret visually.

3 Exploratory Data Analysis

3.1 Decompose Time Series

We will first define our variables:

$$Y_t = m_t + s_t + X_t$$

- Y_t: the data, represents a combination of the trend, seasonality, and residuals.
- m_t: trend component, represents the long-term pattern in the time series.
- s_t: seasonality component, represents recurring patterns within a fixed time interval.
- X_t: residual component, represents random fluctuations in the time series.

We will perform a trend and seasonality analysis and ensure the resulting residuals pass as a stationary time series for inference and forecasting.

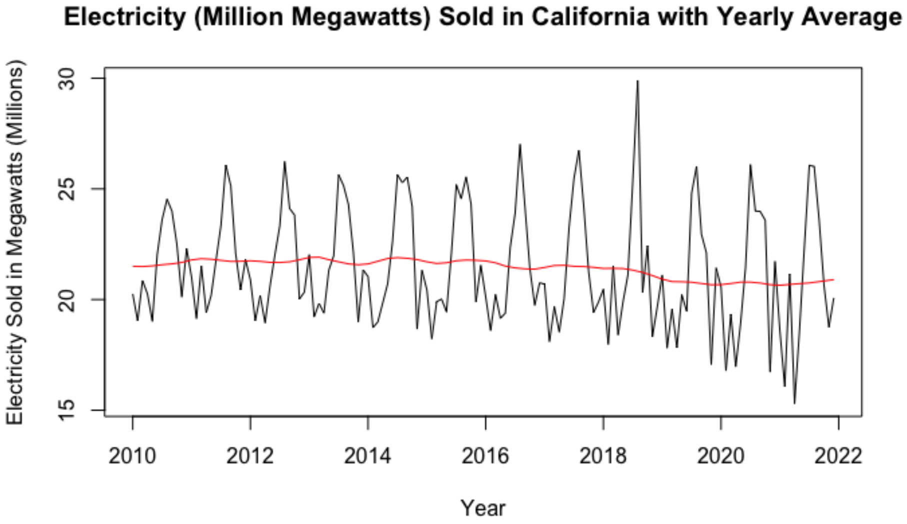

Figure 1: Time Series Plot

When we plot the time series data, we see some distinct patterns of seasonality: Sales are low at the start and end of the year, which corresponds to the winter months when electricity is lower in demand. There is a peak in electricity sold in the months around the summer of 2018 and a drop in the minimum electricity sales from 2020 to 2021.

3.2 ARMA (Autoregressive Moving Average) model

An ARMA model is a combination of the autoregressive (AR) component and the moving average (MA) component. The AR component represents the dependence of the current value on past values, while the MA component represents the dependence of the current value on past error terms. A weakly stationary process can be an ARMA time series ARMA(p,q) if it satisfies the difference equations [1]:

$$X_t = \phi_1 X_{t-1} + \ldots + \phi_p X_{t-p} + Z_t + \theta_1 Z_{t-1} + \ldots + \theta_q Z_{t-q}, \quad t \in Z$$

We will use the Akaike Information Criterion (AIC) as the statistical measure used for our ARMA model selection. The most appropriate model minimizes the AIC which balances its ability to fit and represent the data while avoiding overfitting [2].

To determine the optimal $ARMA(p,q)$ model that minimizes the AIC, we developed a function that fits each combination of p and q to the given residuals and identifies the combination with the lowest AIC.

3.3 Analyze "Smooth" Component

We start by analyzing the "smooth" component of our time series data - trend (m_t) and seasonality (s_t). We performed seasonal and trend decomposition using LOWESS (locally weighted regression and scatter plot smoothing) [3], which focuses on fitting a regression model to local subsets of the data. This method is preferred over the two-sided moving average as it does not leave incomplete averages at the edges of the time series data points, which we need for forecasting.

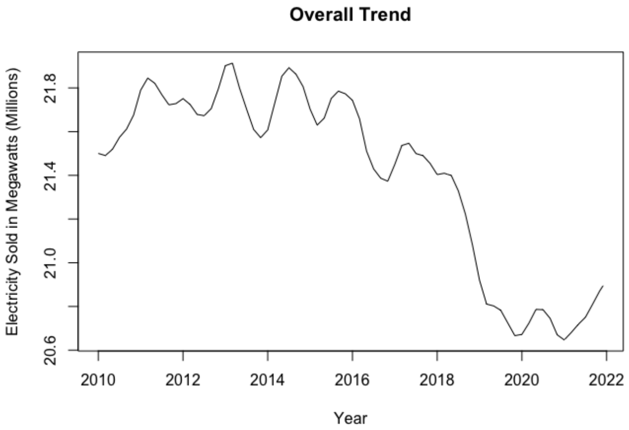

Figure 2: Trend

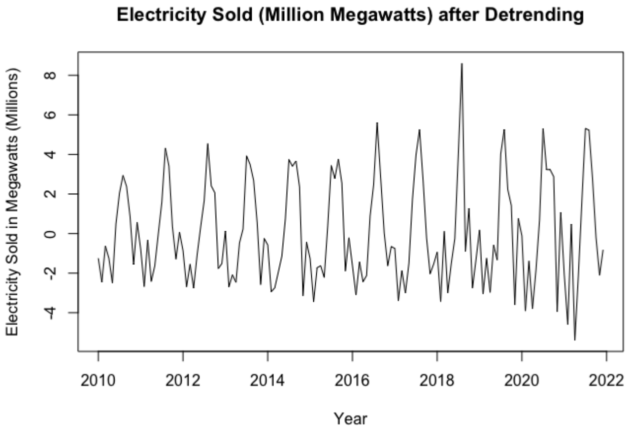

Figure 3: Detrend

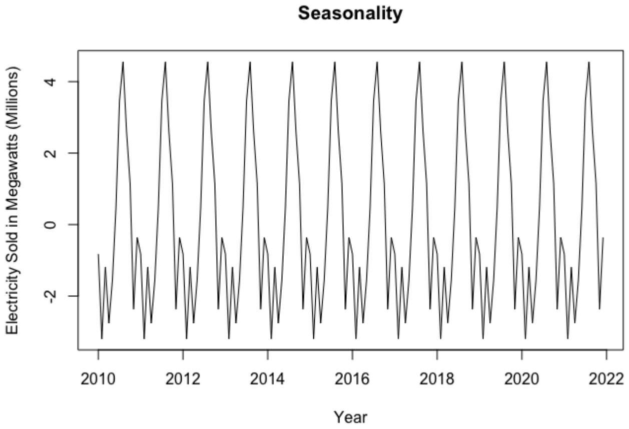

Figure 4: Seasonality

From 2010 to 2022, the trend (m_t) of electricity sales remained relatively stable. It is important to note that the y-axis scale in the plot is highly zoomed in, making it difficult to accurately interpret the exact values. There was a noticeable peak in sales between 2013 and 2015, reaching approximately 21.9 megawatts million. Subsequently, a downward trend emerged, with sales declining from 21.4 megawatts million in 2018 to 20.7 megawatts million in 2020.

Detrending involves

$$Y_t - m_t = s_t + X_t,$$

which leaves us with seasonality and residuals. Next, we will isolate the seasonality to get our residuals.

3.4 Analysis of Residuals

$$X_t = Y_t - m_t - s_t$$

We will look at the residual diagnostics plots below to analyze their distribution and stationarity.

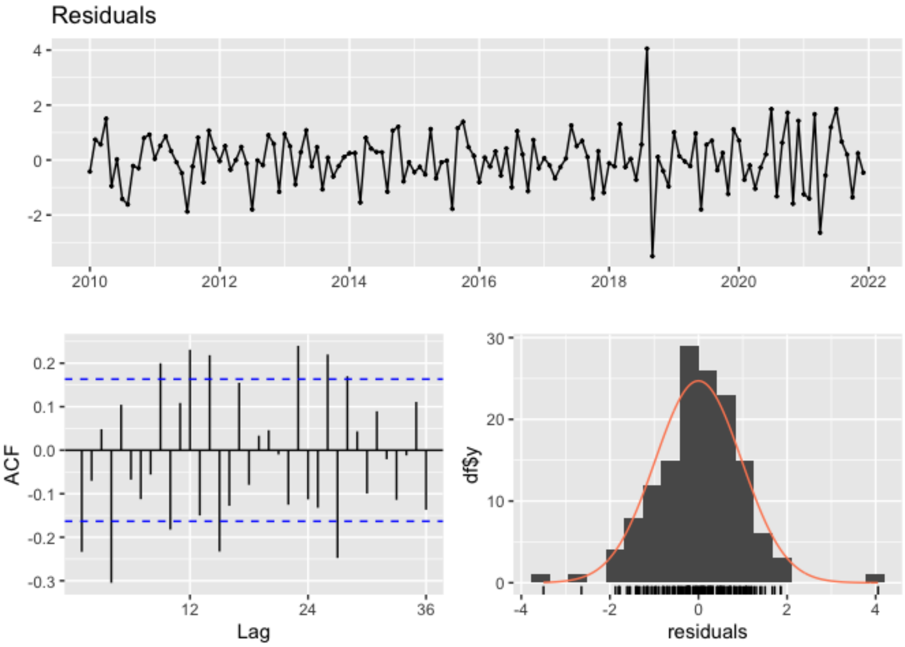

Figure 5: Residual Diagnostics

In the histogram, we can see that there are two major outliers in our residuals that may affect our analysis: August 2018 and September 2018. To ensure that our ARMA models are representative of the underlying pattern of our data, we have to address our outliers.

3.5 Treatment of Outliers

To address the outliers, we divide the dataset into two periods: January 2010 through July 2018 and October 2018 through December 2022. Subsequently, we fit separate ARMA models to each of these periods' residual datasets to forecast and backcast the outliers. By combining the forecasted and backcasted values, we calculate the average to replace the outliers effectively.

Through the function we created listed in Section 3.2, we determined that the suitable $ARMA(p, q)$ models that minimize the AIC for the two residual datasets are $ARMA(4, 5)$ and $ARMA(6, 3)$, with AICs of 177.667 and 83.945 respectively (B.1).

After forecasting and backcasting the residuals, our new predicted residuals are 0.601 for August 2018 and -0.530 for September 2018.

By addressing the outliers, we can ensure that our analysis accurately captures the underlying pattern of the data and enables reliable forecasting and inference.

3.6 Hypothesis testing for stationarity

After addressing the outliers, we used the Augmented Dickey–Fuller test for stationarity [5]. Since the p-value (0.01, B.2) is smaller than a significance level set to 0.05, we reject the null hypothesis of non-stationarity and conclude that our residuals are stationary. We can now use these residuals for the model fitting process. The KPSS Test for Level Stationarity also supports our outcome as the p-value (0.1, B.2) is more than the significance level of 0.05, so we accept the null hypothesis of trend stationary and conclude that the time series is stationary [6].

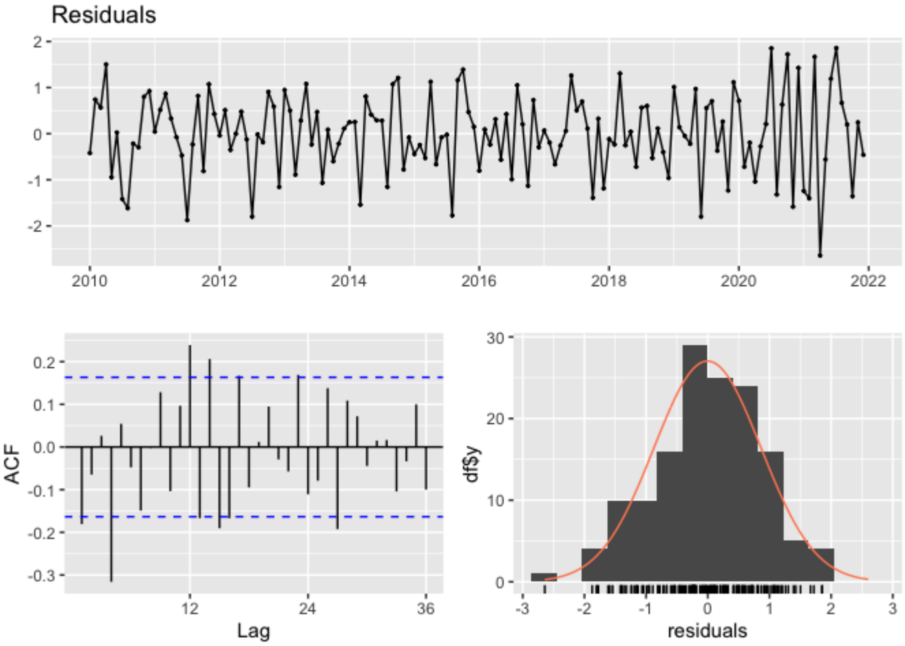

Figure 6: Residual Diagnostics

The residual plot also supports evidence of stationarity as the points are random and in a zig-zag motion. The ACF plot follows stationarity closer with fewer lags outside of the 95% confidence interval.

4 Analyzing the "rough" component

4.1 Model Fitting

Now that our residuals are stationary, we can fit an ARMA model for inference and forecasting.

As stated before, to find the optimal ARMA model, we used the same function mentioned in Section 3.2 to find the p, q values that minimize the AIC.

	1	2	3	4
1	337.4124	338.6755	339.7583	331.1321
2	338.2866	338.6718	340.6673	329.4787
3	336.2805	340.6586	328.4973	331.3229
4	322.9641	324.6465	319.3487	294.6362

We found that $ARMA(4, 4)$ is the most optimal model with an AIC of 294.636.

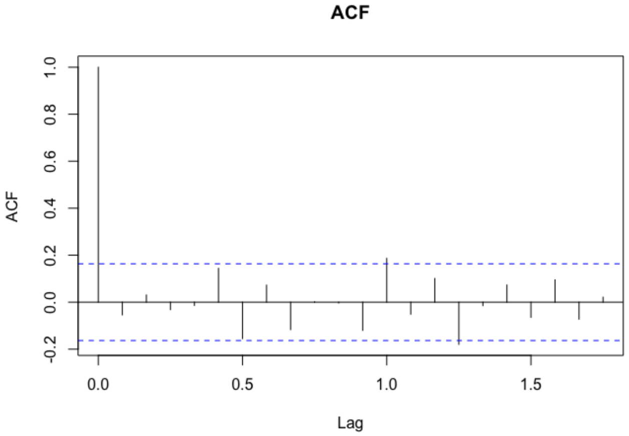

Figure 7: ACF

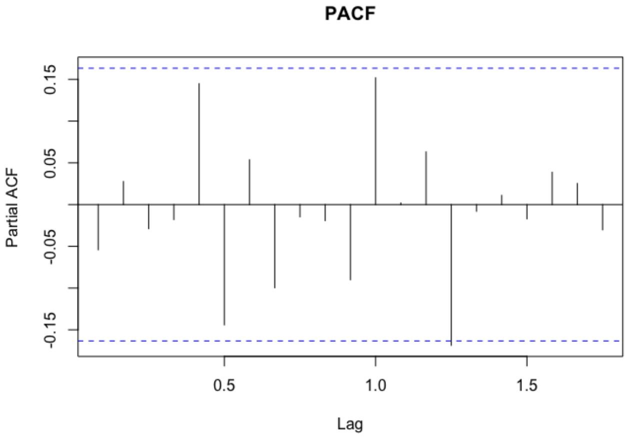

Figure 8: PACF

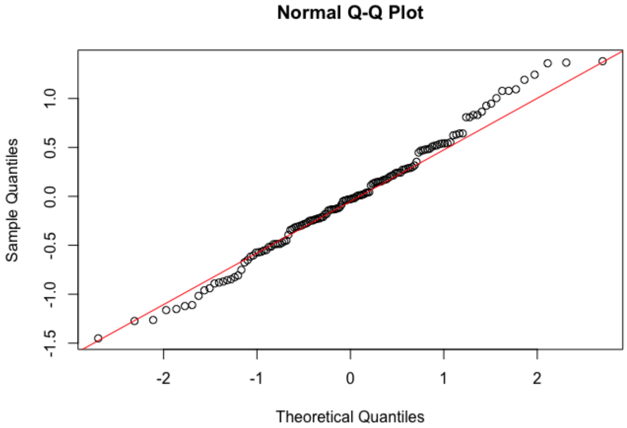

Figure 9: Normal Q-Q Plot

The analysis of the ACF and PACF indicates that our residuals are not significant and conform to the independence assumption. The normal Q-Q plot reveals a generally normal distribution, although a slight heavy tail is observed.

To further validate the independence of our residuals, the Ljung-Box test can be used as our residuals conform to normality [8]. As the p-value (0.5117, B.3) is larger than the significance level of 0.05, we cannot reject our null hypothesis of the residuals being independently distributed. This can infer that our residuals have white noise characteristics.

4.2 Trend Fitting

We will fit a polynomial to the trend component ($\hat{m}_t$) to be used for forecasting. We found that a 4th-degree polynomial fits the trend line well with an adjusted R-squared value of 0.9256, which means that 92.56% of the variability is explained by the 4th-degree polynomial model (B.4).

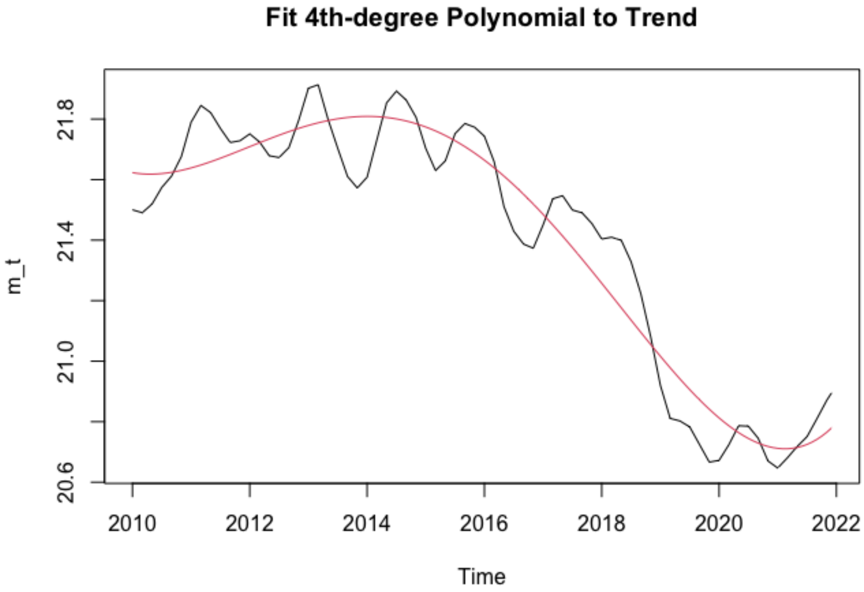

Figure 10: Trend Fitting

The plot above reveals the fitted line effectively captures the curvature and shape of the underlying trend, indicating a strong fit between the two.

4.3 Inference

In the interpretation of our ARMA model, we have conducted a significance test for the coefficients. The p-values for the ϕ_2 (0.475) and intercept (0.425) coefficients are larger than the significance level of 0.05. This indicates that these coefficients are not statistically significant and therefore will be dropped from our final model.

Below is the equation of our final $ARMA(4,4)$ model (B.5):

$$X_t = 0.338X_{t-1} + 0.638X_{t-3} - 0.730X_{t-4} + Z_t - 0.856Z_{t-1} - 0.215Z_{t-2} - 0.856Z_{t-3} + Z_{t-4}$$

In the interpretation of our trend model, we looked at the p-values of our estimated coefficients. Since the p-value (0.427) for the coefficient of X^1 is larger than the significance level of 0.05, it is not statistically significant and will be dropped from our model.

Below is the equation of the polynomial model of our trend component (B.4):

$$m_t = 21.630 + 0.000452X^2 - 0.00000761X^3 + 0.0000000303X^4$$

5 Forecasting

5.1 Predict Residuals

Our objective is to forecast both the residuals and the true values of electricity sales for the year 2022. By having the actual values of 2022 available, we can compare them with our predictions and calculate the prediction error of our model. This allows us to assess the accuracy of our ARMA model.

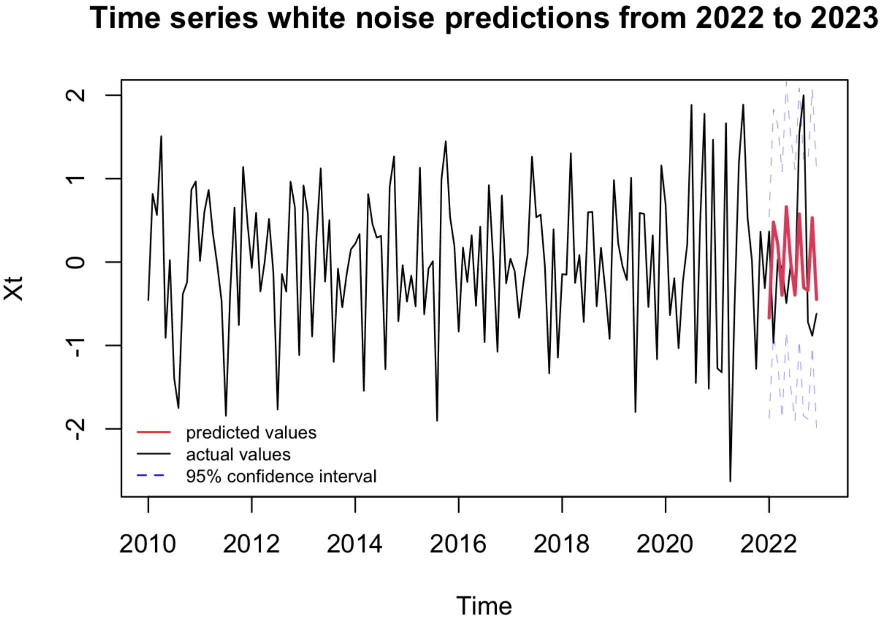

Figure 11: White Noise Predictions

We first predicted the residuals ($\hat{X}_t$) and computed its 95% confidence interval. The prediction error of $\hat{X}_t$ is 19.32.

Compared to the MAPE, mean absolute percentage error, of 253%, the prediction error is significantly smaller than the MAPE. This suggests that the prediction has a substantially smaller deviation from the expected average percentage error. This can be considered a positive indication of our model. However, we cannot solely rely on this single metric to evaluate the accuracy of our model.

There is also a large spike in the actual residuals in September 2022 that our model did not predict, which can be improved to capture more specific patterns effectively.

5.2 Predict Electricity Sales in 2022

Now, we will forecast the predictions for electricity sales in 2022 ($\hat{Y}_t$). We will obtain $\hat{Y}_t$ by:

$$\hat{Y}_{2022} = \hat{m}_{2022} + \hat{s}_{2022} + \hat{X}_{2022}$$

We obtained the predicted trend and residuals from our fitted models above. The seasonal component remains the same for each period since it is constant. By summing the predicted trend, residuals, and the fixed seasonal component, we obtain the complete predicted time series $\hat{Y}_t$.

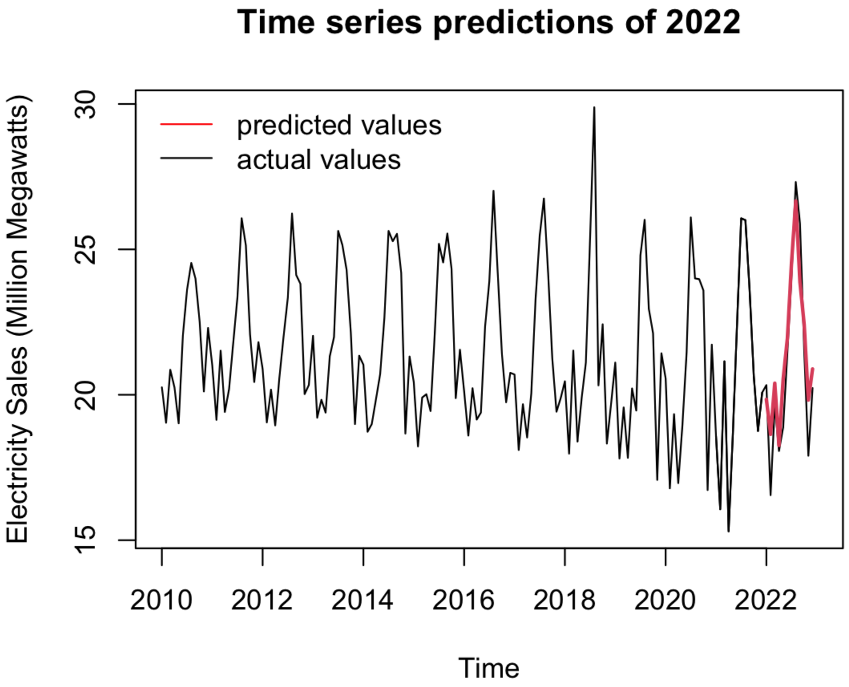

Figure 12: Time Series Predictions

As we can see from the plot, our predictions visually look pretty close to the actual values. The prediction error of $\hat{Y}_t$ is 1.64 which is smaller than $var(\hat{Y}_t)$ of 6.31. Compared to the MAPE of 5.08%, the prediction error is significantly smaller than the MAPE. This suggests that the prediction has a substantially smaller deviation from the expected average percentage error.

One possible explanation for the small prediction error observed in our analysis is that we are only forecasting the data for a relatively short period of one year. The prediction error would likely increase for predictions of over a year.

6 Spectral Analysis

Put simply, spectral analysis is a time series method that decomposes time series data into its frequency components. By decomposing the data in such a way, spectral analysis is instrumental in identifying underlying periodic patterns.

Intuitively, one may believe that an annual cycle of energy sales exists as the amount of energy used in a given month often corresponds to the time of year. This is seen as higher amounts of energy are sold during summer months and lower amounts of sales happen during winter months. So, an annual cycle may have a high degree of explanatory power. Plotting the periodogram of the raw data appears to show a significant contribution of an annual and semi-annual cycle since the data points are high at a frequency of 1/12 and 1/6, respectively.

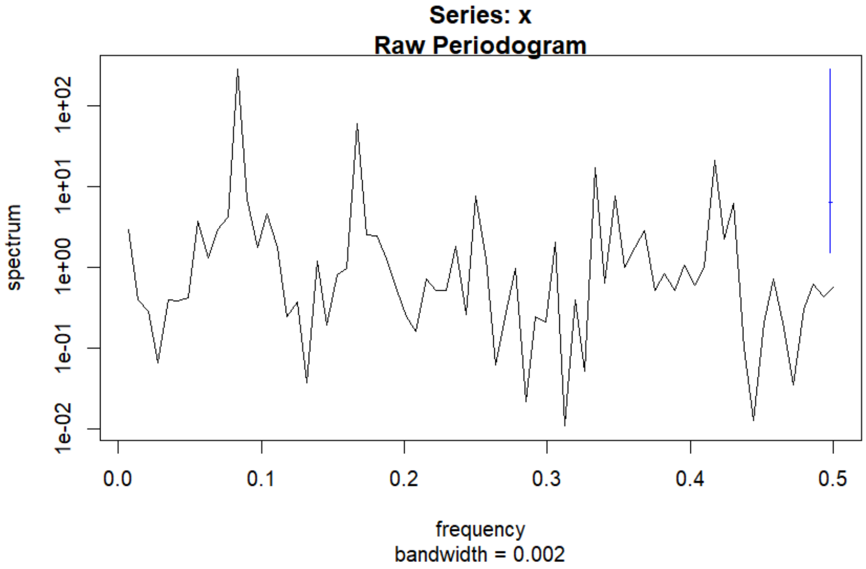

Figure 13: Raw Periodogram

This same detail is further illustrated by the smoothed periodogram. This plot removes much of the noise from the data so that significant data points can more easily be viewed. It should be noted that while other frequencies seem to be significant, they are much smaller in scale and may be still be subject to noise.

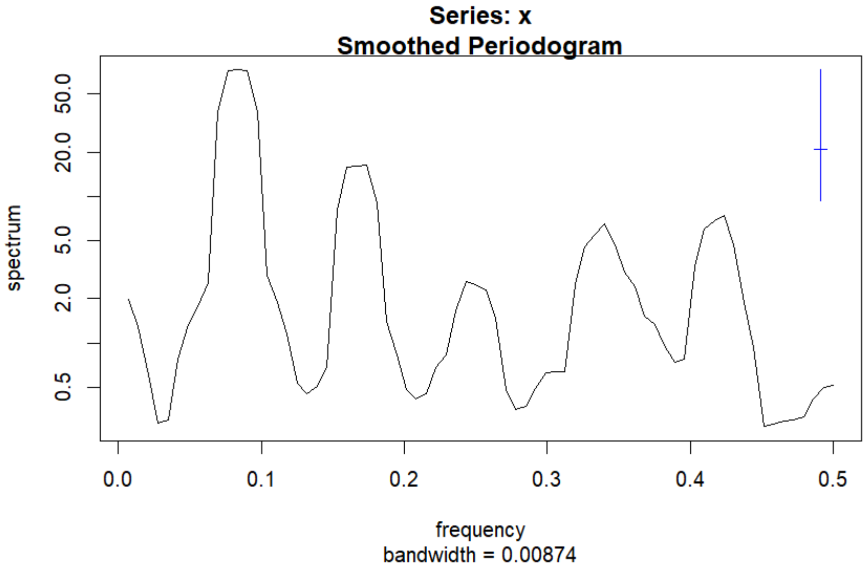

Figure 14: Smoothed Periodogram

However, these plots are based on the raw data, which have yet to be deasonalized and detrended. This makes interpretation of the periodogram difficult as attributes of non-stationarity data may contribute to the spectrum values at the annual and semi-annual data points. Thus, the removal of trend and seasonality is required in order to more accurately understand the periodicity of the data.

6.1 12-Month Cycle

After subtracting out the trend component, we must remove the 6-month cycle by subtracting out the regression coefficients for the sine and cosine functions at a frequency of 1/6:

$$\hat{Y}_t - (-0.320134cos(2\pi time/6) - (1.330287sin(2\pi time/6)$$

Our data is now believed to be stationary according to both the ADF and KPSS test (B.6). This allows us to more accurately view the annual cycle.

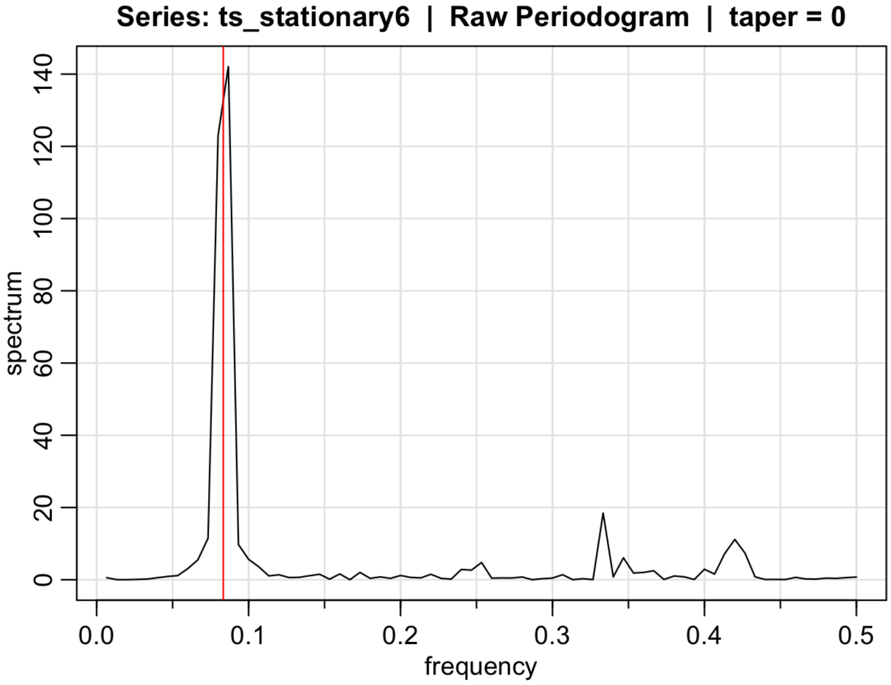

Figure 15: Periodogram with 6-month cycle removed

From this, we find that 39.1% of the variation is explained by the annual cycle. This provides further evidence to suggest that there is a strong annual seasonal component in our data.

6.2 6-Month Cycle

Similarly, we must remove the 12-month cycle by subtracting out the regression coefficients for the sine and cosine functions at at frequency of 1/12:

$$\hat{Y}_t - (-1.166773cos(2\pi time/12) - (-2.608522sin(2\pi time/12)$$

Our data is now believed to be stationary according to both the ADF and KPSS test (B.7). This allows us to more accurately view the semi-annual cycle.

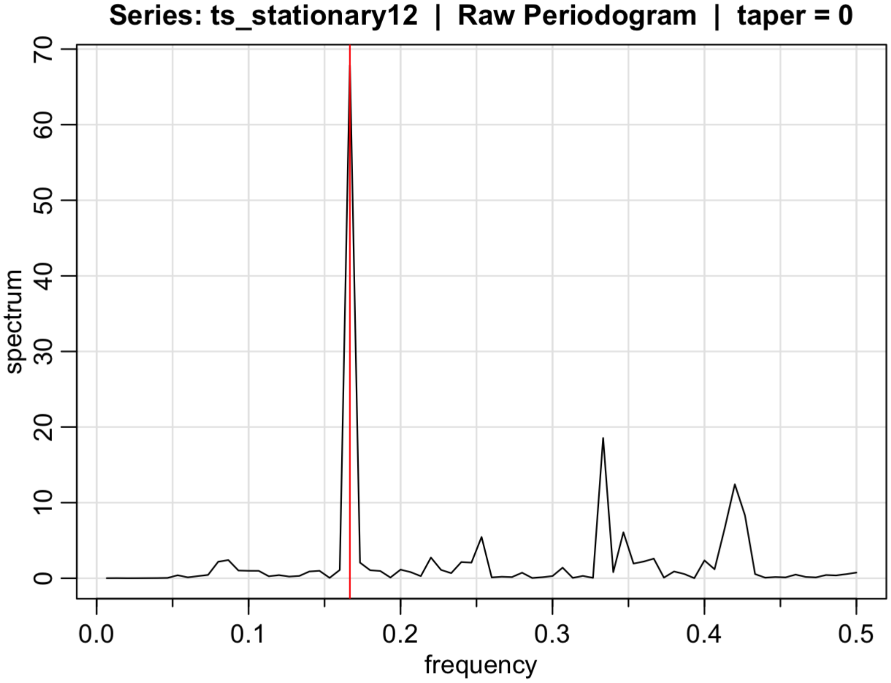

Figure 16: Periodogram with 12-month cycle removed

From this, we find that 34.9% of the variation is explained by the semi-annual cycle. Through the same reasoning as before, this provides further evidence to suggest that there is a strong semi-annual seasonal component in our data.

7 Conclusion and Discussion

Overall, after conducting a time series analysis, this paper brings valuable insight into understanding electricity sales in California. Specifically, we find that the data has both a significant annual and semi-annual component. Furthermore, our model for forecasting electricity sales for 2022 produces a MAPE of 5.08%, which is relatively small. It can then be argued that this ARMA model is powerful in predicting future values. In other words, California's utility companies and power marketers may be able to leverage valuable information from this analysis and forecast energy sales for the summer of 2023 with our model.

However, this analysis is not completely without its faults. For instance, the error was quite high when forecasting the residuals. This is likely the result of a large spike in the actual residuals towards the end of 2022. Unfortunately, our residual model did not accurately predict this spike. Importantly, however, this spike in actual residuals still lies within the 95% confidence interval. Despite the large error, this fact brings some credence to the accuracy of our model since the actual residuals do not lie outside of the confidence interval. Moreover, while the contribution of the annual and semi-annual cycles was investigated, the exact causes of these cycles cannot be stated unless further examination is conducted. These types of patterns could be a result of seasonality due to temperatures at certain times of the year, but could also be a result of business cycles, or even policy changes.

References

[1] Alexander Aue. Introduction to autoregressive moving average (arma) processes. `https://stats.libretexts.org/Bookshelves/Advanced_Statistics/Time_Series_Analysis_(Aue)/3%3A_ARMA_Processes/3.1%3A_Introduction_to_Autoregressive_Moving_Average_(ARMA)_Processes/`, 2022.

[2] Rebecca Bevans. Akaike information criterion. `https://www.scribbr.com/statistics/akaike-information-criterion/`, 2022.

[3] Krois J. Waske B Hartmann, K. Stl decomposition. `https://www.geo.fu-berlin.de/en/v/soga/Geodata-analysis/time-series-analysis/Seasonal-decompositon/STL-decomposition/index.html`, 2018.

[4] Glynn C. Hulley, Benedicte Dousset, and Brian H. Kahn. Rising trends in heatwave metrics across southern california. *Earth's Future*, 8(7):e2020EF001480, 2020.

[5] Jim. Augmented dickey-fuller test in r. `https://www.r-bloggers.com/2022/06/augmented-dickey-fuller-test-in-r/`, 2022.

[6] Statology. Kpss test in r. `https://www.statology.org/kpss-test-in-r/`, 2022.

[7] Ying Wang, Jiechen Wu, and Haiqiong Yi. Analysis on several blackouts caused by extreme weather and its enlightenment. In *IOP Conference Series: Earth and Environmental Science*, volume 766. IOP Publishing, 2021.

[8] Zach. Ljung-box test. `https://www.statology.org/ljung-box-test/`, 2020.

Appendix

A Raw R Code

R code

```
# Load Packages
library(tidyverse)
library(forecast)
library(lmtest)
library(tseries)
library(astsa)

# Read data
energy <- read.csv("energy-ca.csv")

# Subset data to select columns we need: Year, Month, State, Megawatts (sold)
energy_df <- subset(energy, select = c("X", "X.1", "X.2", "X.16"))
# Remove first two rows
energy_df <- energy_df[-c(1,2),]

# Add column names
colnames(energy_df) <- c("Year", "Month", "State", "Megawatts")
# Check the amount of times "CA" appears in the "X.2" column
# corresponds to the respective state
table(grepl("CA", energy_df$State))
# Subset data to select California
energy_df <- energy_df[energy_df$State == "CA",]

# Get rid of empty rows
row.names(energy_df) <- NULL

# Check the amount of times in which the State column equals "CA".
# Since this number, 158, matches with the earlier number,
# we have support to say no important data was lost
table(grepl("CA", energy_df$State))

# Change to integers
energy_df$Year <- as.integer(energy_df$Year)
energy_df$Month <- as.integer(energy_df$Month)
energy_df$Megawatts <- as.integer(gsub(",", "", energy_df$Megawatts))

# Arrange energy_df in ascending order of time
energy_df <- energy_df %>%
  arrange(Year, Month)

# Remove the 2022 and 2023 year as only Jan and Feb data is present
energy_df2022 <- energy_df[133:158,]
energy_df1 <- energy_df[energy_df$Year != 2023 & energy_df$Year != 2022,]
energyall <- energy_df[energy_df$Year != 2023,]
```

```
# Time series object
energy_ts <- ts(energy_df1[,4], start= 2010, frequency = 12)
energy2022 <- ts(energy_df2022[,4], start= 2021, frequency = 12)
energyall <- ts(energyall[,4], start= 2010, frequency = 12)

# Divide data by a million to standardize units in a more perceivable way
energy_ts <- energy_ts/(10^6)
energy2022 <- energy2022/(10^6)
energyall <- energyall/(10^6)

# Seasonal and Trend decomposition using Loess
model <- stl(energy_ts, s.window = "periodic")

# Get m_t
m_t <- model$time.series[, "trend"]

# Plot time series data with yearly average
ts.plot(energy_ts,
        xlab = "Year",
        ylab = "Electricity Sold in Megawatts (Millions)",
        main = "Electricity (Million Megawatts) Sold in California with Yearly Average")
lines(m_t, col = "red")

# Overall trend
ts.plot(m_t,
        xlab = "Year",
        ylab = "Electricity Sold in Megawatts (Millions)",
        main = "Overall Trend")

# Detrend only
detrend <- energy_ts - m_t

# Plot new time series that clearly exposes seasonality
ts.plot(detrend,
        xlab = "Year",
        ylab = "Electricity Sold in Megawatts (Millions)",
        main = "Electricity Sold (Million Megawatts) after Detrending")

# Get seasonality
s_t <- model$time.series[, "seasonal"]
ts.plot(s_t,
        xlab = "Year",
        ylab = "Electricity Sold in Megawatts (Millions)",
        main = "Seasonality")

# Extract residuals
```

```
z_t <- energy_ts - m_t - s_t

# Plot time series, ACF, histogram, QQ-plot of white noise
checkresiduals(z_t)
qqnorm(z_t)
qqline(z_t, col = "red")

# Jan 2010 - Jul 2018
df1 <- energy_df1[1:103,]
df1_ts <- ts(df1[,4], start= 2010, frequency = 12)
df1_ts <- df1_ts/(10^6)

# Oct 2018 - Dec 2021
df2 <- energy_df1[106:144,]
df2_ts <- ts(rev(df2[,4]), start= c(2018, 10), frequency = 12) # reversed order
df2_ts <- df2_ts/(10^6)

# Decompose both ts
decom_df1 <- stl(df1_ts, s.window = "periodic")
decom_df2 <- stl(df2_ts, s.window = "periodic")

# Function to find optimal p, q values
find_arima <- function(X, p_values, d_values, q_values) {

  aic_matrix <- matrix(NA, nrow = length(p_values), ncol = length(q_values))

  # Loop over all combinations of p, d, and q values
  for (i in 1:length(p_values)) {
    for (j in 1:length(d_values)) {
      for (k in 1:length(q_values)) {

        # Try fitting the ARMA model with current p, d, q values
        tryCatch({
          arma_model <- arima(X, order = c(p_values[i], d_values[j], q_values[k]))
          # Calculate the AIC value for the current model
          aic <- AIC(arma_model)
          # Store the AIC value in the matrix
          aic_matrix[i, k] <- aic
        }, error = function(e) {
          # If an error occurs, assign -1 to the matrix element
          aic_matrix[i, k] <- -1
        })

      }
    }
  }
  print(aic_matrix)

  # Find the indices of the minimum AIC value in the matrix
  min_indices <- which(aic_matrix == min(aic_matrix), arr.ind = TRUE)
  # Extract the corresponding p, d, q values with the minimum AIC
  best_p <- p_values[min_indices[, 1]]
  best_d <- d_values[min_indices[, 2]]
  best_q <- q_values[min_indices[, 2]]

  # Print the best p, d, q values
  cat("Best p:", best_p, "\n")
  cat("Best d:", best_d, "\n")
  cat("Best q:", best_q, "\n")
  cat("AIC:", min(aic_matrix, na.rm = TRUE), "\n")
}

# Define the range of values to consider for p, d, and q
p_values <- 0:6
d_values <- 0
q_values <- 0:6

find_arima(decom_df1$time.series[, "remainder"], p_values, d_values, q_values)
find_arima(decom_df2$time.series[, "remainder"], p_values, d_values, q_values)
arma1 <- arima(decom_df1$time.series[, "remainder"], order = c(4, 0, 5))
arma2 <- arima(decom_df2$time.series[, "remainder"], order = c(6, 0, 3))

# Forecast the residuals w/ ARMA(5,6)
forecast_zt <- predict(arma1, n.ahead = 2)$pred

# Backcast the residuals w/ ARMA(5,6)
backcast_zt <- rev(predict(arma2, n.ahead = 2)$pred)

# Average
avg_zt <- (forecast_zt + backcast_zt)/2
avg_zt

# Indices and values to replace
indices <- c(104, 105)
new_values <- c(avg_zt)
z_t[indices] <- new_values
zt_all <- model_all$time.series[, "remainder"]
zt_all[indices] <- new_values

# Residual Diagnostics
checkresiduals(z_t)
```

```
# Augmented Dickey{Fuller test for stationarity
adf.test(z_t)

# KPSS test
kpss.test(z_t)

find_arima(z_t, 1:4, 0, 1:4)

# Fit an ARMA model to the residuals
arma_model <- arima(z_t, order = c(4, 0, 4))
arma_model

# White Noise
acf(arma_model$residuals, na.action = na.pass, main = "ACF")
pacf(arma_model$residuals, na.action = na.pass, main = "PACF")

# Normality
qqnorm(arma_model$residuals, main = "Q-Q Plot") # Heavy tail
qqline(arma_model$residuals, col = "red")

Box.test(arma_model$residuals, type = "Ljung-Box")
coeftest(arma_model)

# Forecasting next year
xt_hat22 <- predict(arma_model, n.ahead = 12)

# Plot the time series with the 2022 forecast and 95% prediction intervals
plot(zt_all, xlim = c(2010, 2023),
     main = "Time series white noise predictions from 2022 to 2023", ylab = "Xt")
points(xt_hat22$pred, type = "l", col = 2, lwd = 2)
points(xt_hat22$pred - 2*xt_hat22$se, type = "l", col = "blue", lty = 2, lwd = 0.2)
points(xt_hat22$pred + 2*xt_hat22$se, type = "l", col = "blue", lty = 2, lwd = 0.2)
legend("bottomleft", lty=c(1, 1, 2), bty="n", col=c("red", "black", "blue"),
       legend=c("predicted values", "actual values", "95% confidence interval"), cex=0.7)

# Get true Xt’s in 2022
model2022 <- stl(energy2022, s.window = "periodic")
xt_22 <- model2022$time.series[, "remainder"]

# Get prediction error of Xt’s in 2022
pred_error <- sum((xt_22 - xt_hat22$pred)^2/var(xt_22[13:24]))
pred_error
accuracy(xt_hat22$pred, xt_22[13:24])

# To predict Yt - fit a polynomial to trend component
```

```
t <- 1:length(m_t)
t2 <- tˆ2
t3 <- tˆ3
t4 <- tˆ4

# Fit model
trend_model <- lm(m_t ˜ t + t2 + t3 + t4)
summary(trend_model)
g <- ts(predict(trend_model), start = 2010, frequency = 12)

# Plot 4th degree polynomial on the trend line
plot(m_t, main = "Fit 4th-degree Polynomial to Trend")
points(g, type = "l", lwd = 1, col = 2)

# Predict the trend for the 2022 year by fitting the next 12 data points to the model
x <- 145:156
m2022 <- 21.63 + 0.0004515*(x^2) - 0.000007607*(x^3) + 0.00000003032*(x^4)

# Add the decomposed portion back for 2022
Yt <- xt_hat22$pred + s_t[1:12] + m2022

# Plot
plot(energy_ts, xlim = c(2010, 2023),
     main = "Time series predictions of 2022",
     ylab = "Electricity Sales (Million Megawatts)")
points(energy2022, type = "l", lwd = 1)
points(Yt, type = "l", col = 2, lwd = 2)
legend("topleft", lty=1, bty = "n", col=c("red","black"),
    c("predicted values","actual values"))

pred_error <- sum((energy2022 - Yt)^2/var(energy2022[13:24]))
pred_error
accuracy(Yt, energy2022[13:24])

# Spectral Analysis
Ynew <- z_t + m_t + s_t

# Remove trend
ts_for_spectral <- Ynew %>%
  - m_t %>%
  as.data.frame()

# Add time column in months
ts_for_spectral$time = seq(1, 144, by = 1) %>% as.numeric()

# Raw periodogram
```

```
spectrum(Ynew)

# Smooth periodogram
spectrum(Ynew, span = 5)

# Remove 6-month cycle
reg_spec6 <- lm(ts_for_spectral$x ~ cos(2*pi*time/6) + sin(2*pi*time/6),
                data = ts_for_spectral)
reg_spec6

ts_stationary6 <- (ts_for_spectral$x - (-0.320134*cos(2*pi*ts_for_spectral$time/6)) - (1.330
  unlist()

# Add time column in months
ts_stationary6$time <-seq(1, 144, by = 1) %>%
  as.numeric()
ts_stationary6 <- as.data.frame(ts_stationary6)
ts_stationary6 <- t(ts_stationary6[1,])

# Test for stationarity
adf.test(ts_stationary6)
kpss.test(ts_stationary6)

# Plot 6-month removed periodogram
specvalues6 <- mvspec(ts_stationary6, log = ’no’)
mvspec(ts_stationary6, log = ’no’)
  abline(v = 1/12, col = ’red’)

specvalues6 <- as.data.frame(specvalues6$details)$spectrum

# Variance
max(specvalues6) / sum(specvalues6)
# 34.9% of variation is explained by the semi-annual cycle

# Remove 12-month cycle
reg_spec12 <- lm(ts_for_spectral$x ~ cos(2*pi*time/12) + sin(2*pi*time/12),
                 data = ts_for_spectral)
reg_spec12

ts_stationary12 <- (ts_for_spectral$x -
(-1.166773*cos(2*pi*ts_for_spectral$time/12)) - (-2.608522*sin(2*pi*ts_for_spectral$time/12)
  unlist()

# Add time column in months
ts_stationary12$time <- seq(1, 144, by = 1) %>% as.numeric()
ts_stationary12 <- as.data.frame(ts_stationary12)
```

```
ts_stationary12 <- t(ts_stationary12[1,])

# Test for stationarity
adf.test(ts_stationary12)
kpss.test(ts_stationary12)

# Plot 12-month removed periodogram
specvalues12 <- mvspec(ts_stationary12, log = 'no')
mvspec(ts_stationary12, log = 'no')
  abline(v = 1/6, col = 'red')

specvalues12 <- as.data.frame(specvalues12$details)$spectrum

# Variance
max(specvalues12) / sum(specvalues12)
# 34.9% of variation is explained by the semi-annual cycle
```

B R Code Outputs

B.1 AICs of ARMA(p, q) for treatment of outliers

```
         [,1]     [,2]     [,3]     [,4]     [,5]     [,6]
[1,] 209.2759 209.7890 210.5863 208.5279 197.0930 211.0968
[2,] 208.8985 186.7521 184.0650 208.8150 199.3339 200.5950
[3,] 206.1623 185.2677 210.6292 209.3330 199.8704 194.4386
[4,] 199.7025 185.7590 200.0202 187.5871 177.6665 179.5514
[5,] 200.7929 187.7262 188.4853 177.7545 177.9557 179.8225
[6,] 200.7805 204.7918 195.1796 179.7266 181.3450 181.6694
Best p: 4
Best q: 5
AIC: 177.6665
```

```
          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]
[1,]  92.62386 91.35960 93.35059 94.90999 92.50690 93.86412
[2,]  98.57093 93.34448 92.67357 87.43528 87.42289 90.22785
[3,] 100.55754 94.58201 91.97182 89.33916 88.68472 88.96863
[4,]  90.00232 91.68912 89.34313 88.04496 88.70298 87.52728
[5,]  91.99246 93.55198 94.34543 89.72021 91.81324 88.43952
[6,]  88.25243 89.39336 83.94458 87.99452 84.93342 86.46731
Best p: 6
Best q: 3
AIC: 83.94458
```

B.2 Stationarity Tests

```
Augmented Dickey-Fuller Test

data:  z_t
Dickey-Fuller = -7.4002, Lag order = 5, p-value = 0.01
alternative hypothesis: stationary

KPSS Test for Level Stationarity

data:  z_t
KPSS Level = 0.013708, Truncation lag parameter = 4, p-value = 0.1
```

B.3 Box-Ljung Test

```
Box-Ljung test

data:  arma_model$residuals
X-squared = 0.43051, df = 1, p-value = 0.5117
```

B.4 Trend Summary

```
Call:
lm(formula = m_t ~ t + t2 + t3 + t4)

Residuals:
     Min       1Q   Median       3Q      Max
-0.23566 -0.09655  0.01573  0.08098  0.22314

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.163e+01  4.891e-02 442.154  < 2e-16 ***
t           -3.696e-03  4.640e-03  -0.797 0.427051
t2           4.515e-04  1.295e-04   3.487 0.000654 ***
t3          -7.607e-06  1.339e-06  -5.679 7.61e-08 ***
t4           3.032e-08  4.583e-09   6.616 7.38e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1125 on 139 degrees of freedom
Multiple R-squared:  0.9276,Adjusted R-squared:  0.9256
F-statistic: 445.5 on 4 and 139 DF,  p-value: < 2.2e-16
```

B.5 ARMA Summary

```
z test of coefficients:

            Estimate Std. Error  z value  Pr(>|z|)    
ar1        0.3375542  0.0562559   6.0003 1.969e-09 ***
ar2        0.0261861  0.0366217   0.7150    0.4746    
ar3        0.6383080  0.0359589  17.7510 < 2.2e-16 ***
ar4       -0.7303466  0.0567039 -12.8800 < 2.2e-16 ***
ma1       -0.8564889  0.0531204 -16.1235 < 2.2e-16 ***
ma2       -0.2146981  0.0402335  -5.3363 9.486e-08 ***
ma3       -0.8564984  0.0539936 -15.8630 < 2.2e-16 ***
ma4        0.9999877  0.0574441  17.4080 < 2.2e-16 ***
intercept -0.0039479  0.0049480  -0.7979    0.4249    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

B.6 6-month Test for Stationarity

```
Augmented Dickey-Fuller Test

data:  ts_stationary6
Dickey-Fuller = -14.118, Lag order = 5, p-value = 0.01
alternative hypothesis: stationary

KPSS Test for Level Stationarity

data:  ts_stationary6
KPSS Level = 0.02058, Truncation lag parameter = 4, p-value = 0.1
```

B.7 12-month Test for Stationarity

```
Augmented Dickey-Fuller Test

data:  ts_stationary12
Dickey-Fuller = -7.9711, Lag order = 5, p-value = 0.01
alternative hypothesis: stationary

KPSS Test for Level Stationarity

data:  ts_stationary12
KPSS Level = 0.025929, Truncation lag parameter = 4, p-value = 0.1
```