Key Factors Driving Load Growth in PJM and ERCOT

https://github.com/jazpritch/KarneiKogaWargoPritchett_ENV872_EDA_FinalProject.git

Hanna Karnei, Haru Koga, Jazmine Pritchett, Jaimie Wargo

Fall 2023

Contents


```
# Set your ggplot theme
mytheme <- theme_gray()+
  theme(plot.title = element_text(size = 16, hjust= 0),
        axis.title = element_text(size = 13),
        legend.position = 'right')

theme_set(mytheme)
```

Load in data

```
# Read the CSV files
x2010<- read_excel("Data/Raw/ERCOT_Source_Data/2010_ercot_hourly_load_data.xls") %>%
  rename(Hour_Ending = Hour_End) %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2011<- read_excel("Data/Raw/ERCOT_Source_Data/2011_ercot_hourly_load_data.xls") %>%
  rename(Hour_Ending = Hour_End) %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2012<- read_excel("Data/Raw/ERCOT_Source_Data/2012_ercot_hourly_load_data.xls") %>%
  rename(Hour_Ending = Hour_End) %>%
```

```
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2013<- read_excel("Data/Raw/ERCOT_Source_Data/2013_ercot_hourly_load_data.xls") %>%
  rename(Hour_Ending = Hour_End) %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2014<- read_excel("Data/Raw/ERCOT_Source_Data/2014_ercot_hourly_load_data.xls") %>%
  rename(Hour_Ending = Hour_End) %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2015<- read_excel("Data/Raw/ERCOT_Source_Data/native_load_2015.xls") %>%
  rename(Hour_Ending = Hour_End) %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2016<- read_excel("Data/Raw/ERCOT_Source_Data/native_Load_2016.xlsx") %>%
  rename(Hour_Ending = Hour_End) %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2017<- read_excel("Data/Raw/ERCOT_Source_Data/native_Load_2017.xlsx") %>%
  rename(Hour_Ending = "Hour Ending") %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2018<- read_excel("Data/Raw/ERCOT_Source_Data/Native_Load_2018.xlsx") %>%
  rename(Hour_Ending = "HourEnding") %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2019<- read_excel("Data/Raw/ERCOT_Source_Data/Native_Load_2019.xlsx") %>%
  rename(Hour_Ending = "HourEnding") %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2020<- read_excel("Data/Raw/ERCOT_Source_Data/Native_Load_2020.xlsx") %>%
  rename(Hour_Ending = "HourEnding") %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2021<- read_excel("Data/Raw/ERCOT_Source_Data/Native_Load_2021.xlsx") %>%
  rename(Hour_Ending = "Hour Ending") %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))
x2022<- read_excel("Data/Raw/ERCOT_Source_Data/Native_Load_2022.xlsx") %>%
  rename(Hour_Ending = "Hour Ending") %>%
  mutate(Hour_Ending = as.Date(Hour_Ending, format = "%m/%d/%Y"))

# Merge the data frames based on a common column
ercot <- bind_rows(x2010, x2011, x2012, x2013, x2014, x2015, x2016, x2017, x2018, x2019, x2020, x2021, x

ercot_load <- ercot %>%
  select(Hour_Ending,ERCOT)
```

Rationale and Research Questions

Dataset Information

Exploratory Analysis

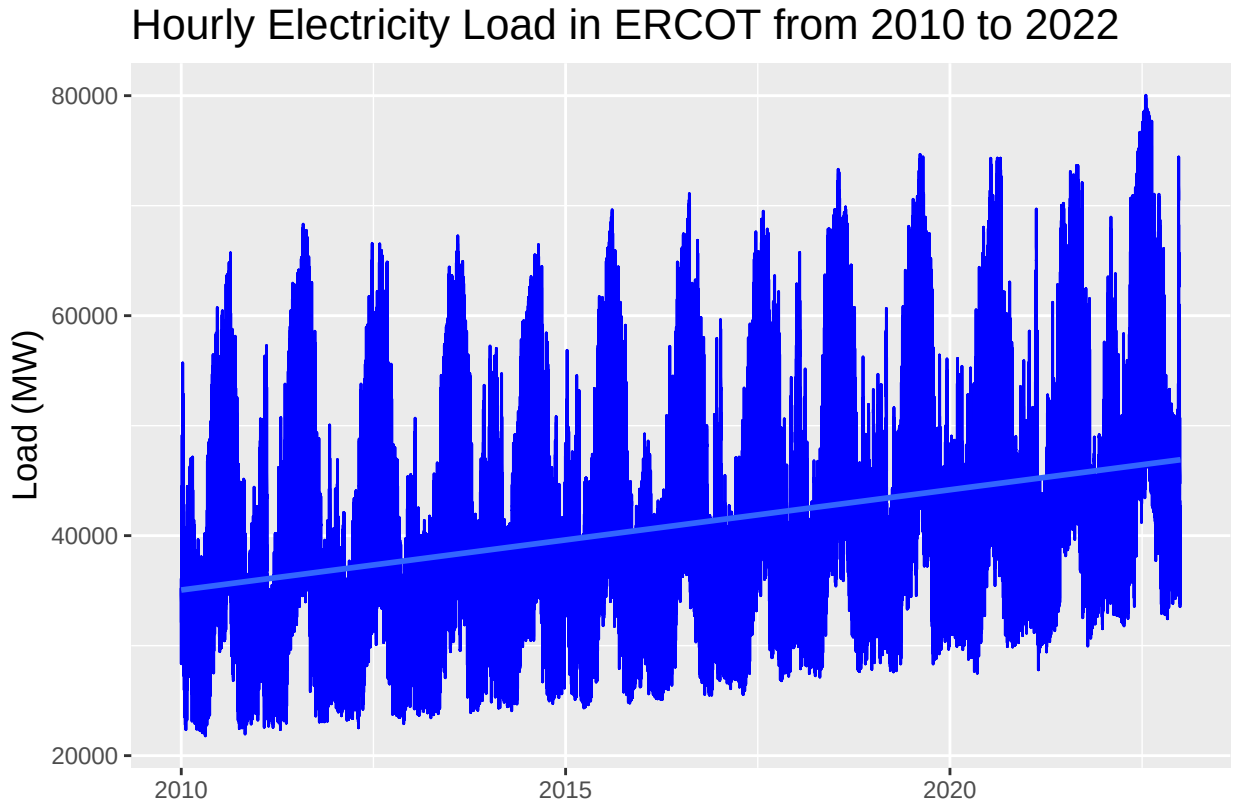
Distribution of ERCOT hourly load data, as well as aggregated by day and month.

```
# Plot hourly data

head(ercot_load)
```

```
## # A tibble: 6 x 2
##   Hour_Ending  ERCOT
##   <date>       <dbl>
## 1 2010-01-01  32094.
## 2 2010-01-01  32171.
## 3 2010-01-01  32242.
## 4 2010-01-01  32459.
## 5 2010-01-01  33124.
## 6 2010-01-01  34315.
```

```
ggplot(ercot_load, aes(x = Hour_Ending, y = ERCOT)) +
  geom_line(color = "blue") +
  geom_smooth(method = "lm") +
  ggtitle('Hourly Electricity Load in ERCOT from 2010 to 2022') +
  labs(x = "", y = "Load (MW)")
```

```
## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 2 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 1 row containing missing values (`geom_line()`).
```




```
#Aggregate data by year

ercot_load_yearly <- ercot_load %>%
  mutate(
    year = year(Hour_Ending)
  ) %>%
  group_by(year) %>%
  summarise(
    Mean_Load = mean(ERCOT, na.rm = TRUE)
  ) %>%
  drop_na()

ercot_load_yearly
```

```
## # A tibble: 13 x 2
##     year Mean_Load
##    <dbl>     <dbl>
##  1  2010    36336.
##  2  2011    38127.
##  3  2012    37000.
##  4  2013    37869.
##  5  2014    38828.
##  6  2015    39669.
##  7  2016    40006.
##  8  2017    40782.
##  9  2018    42949.
```

```
## 10  2019    43818.
## 11  2020    43463.
## 12  2021    44829.
## 13  2022    49074.
```

```
#Aggregate data by month

ercot_load_monthly <- ercot_load %>%
  mutate(
    Year = year(Hour_Ending),
    Month = month(Hour_Ending)
  ) %>%
  group_by(Year, Month) %>%
  summarise(
    Mean_Load = mean(ERCOT, na.rm = TRUE)
  )
```

```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

```
ercot_load_monthly$NewDate <-
  as.Date(paste(ercot_load_monthly$Year, ercot_load_monthly$Month, "01", sep = "-"))

head(ercot_load_monthly)
```

```
## # A tibble: 6 x 4
## # Groups:   Year [1]
##    Year Month Mean_Load NewDate
##   <dbl> <dbl>     <dbl> <date>
## 1  2010     1    34884. 2010-01-01
## 2  2010     2    35240. 2010-02-01
## 3  2010     3    29550. 2010-03-01
## 4  2010     4    29856. 2010-04-01
## 5  2010     5    36547. 2010-05-01
## 6  2010     6    44000. 2010-06-01
```

```
#Aggregate data by day

ercot_load_daily<- ercot_load %>%
  mutate(
    Year = year(Hour_Ending),
    Month = month(Hour_Ending),
    Day = day (Hour_Ending)
  ) %>%
  group_by(Year, Month, Day) %>%
  summarise(
    Mean_Load = mean(ERCOT, na.rm = TRUE)
  )
```

```
## `summarise()` has grouped output by 'Year', 'Month'. You can override using the
## `.groups` argument.
```

```r
ercot_load_daily$NewDate <-
  as.Date(paste(ercot_load_daily$Year, ercot_load_daily$Month,ercot_load_daily$Day, sep = "-"))

head(ercot_load_daily)
```

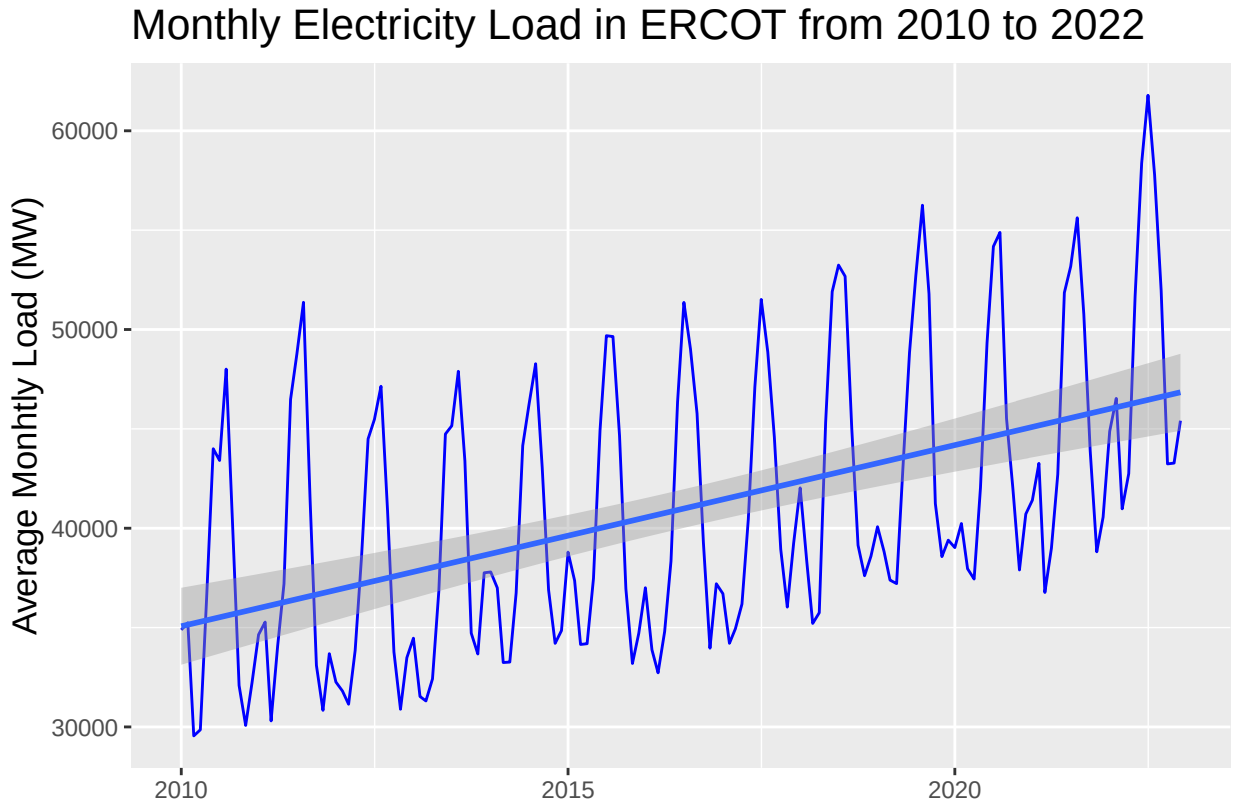
```
## # A tibble: 6 x 5
## # Groups:   Year, Month [1]
##    Year Month   Day Mean_Load NewDate
##   <dbl> <dbl> <int>     <dbl> <date>
## 1  2010     1     1    33080. 2010-01-01
## 2  2010     1     2    33780. 2010-01-02
## 3  2010     1     3    33892. 2010-01-03
## 4  2010     1     4    38857. 2010-01-04
## 5  2010     1     5    40971. 2010-01-05
## 6  2010     1     6    39005. 2010-01-06
```

```r
graph1 <- ggplot(ercot_load_monthly, aes(x = NewDate, y = Mean_Load)) +
  geom_line(color = "blue") +
  geom_smooth(method = "lm") +
  ggtitle('Monthly Electricity Load in ERCOT from 2010 to 2022') +
  labs(x = "", y = "Average Monhtly Load (MW)")

graph1
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 1 row containing missing values (`geom_line()`).
```

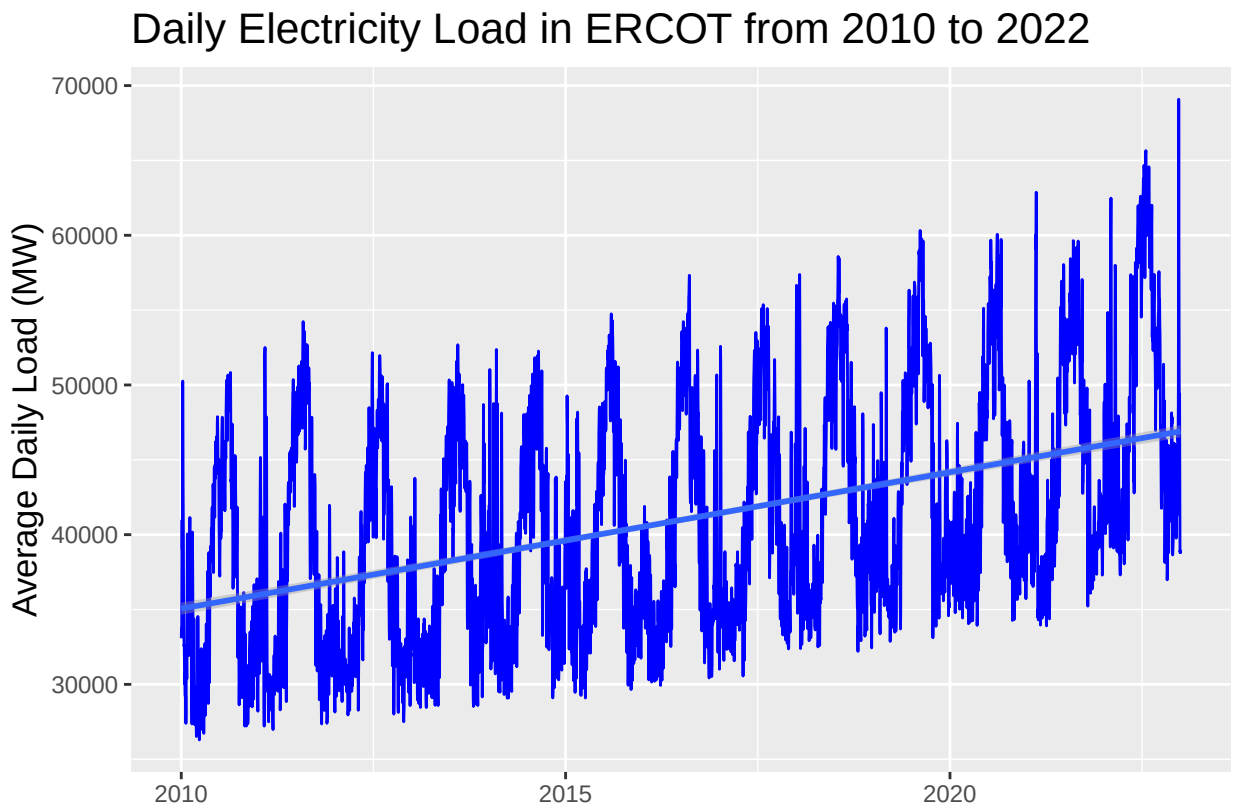



```
graph2 <- ggplot(ercot_load_daily, aes(x = NewDate, y = Mean_Load)) +
  geom_line(color = "blue") +
  geom_smooth(method = "lm") +
  ggtitle('Daily Electricity Load in ERCOT from 2010 to 2022') +
  labs(x = "", y = "Average Daily Load (MW)")

graph2
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_smooth()`).
```

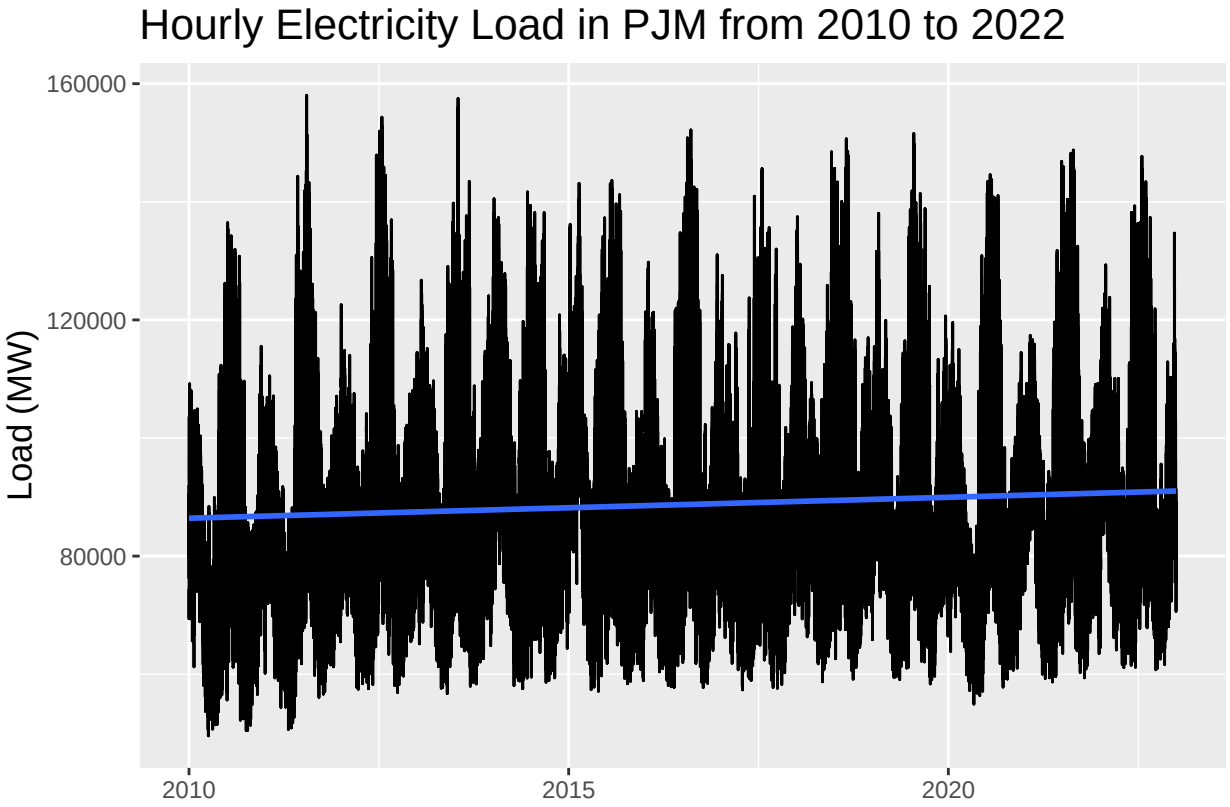
```
## Warning: Removed 1 row containing missing values (`geom_line()`).
```



Distribution of PJM hourly load data, as well as aggregated by day and month.

```
# Plot hourly data

ggplot(PJM_load, aes(x = Date, y = Load_MW)) +
  geom_line(color = "black") +
  geom_smooth(method = "lm") +
  ggtitle('Hourly Electricity Load in PJM from 2010 to 2022') +
  labs(x = "", y = "Load (MW)")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
#Aggregate data by month

PJM_load_monthly <- PJM_load %>%
  mutate(
    Year = year(Date),
    Month = month(Date)
  ) %>%
  group_by(Year, Month) %>%
  summarise(
    Mean_Load = mean(Load_MW, na.rm = TRUE)
  )
```

```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

```
PJM_load_monthly$NewDate <-
  as.Date(paste(PJM_load_monthly$Year, PJM_load_monthly$Month, "01", sep = "-"))

head(PJM_load_monthly)
```

```
## # A tibble: 6 x 4
## # Groups:   Year [1]
##    Year Month Mean_Load NewDate
##   <dbl> <dbl>     <dbl> <date>
## 1  2010     1    88058. 2010-01-01
```

```
## 2  2010     2    86967. 2010-02-01
## 3  2010     3    74220. 2010-03-01
## 4  2010     4    67861. 2010-04-01
## 5  2010     5    73517. 2010-05-01
## 6  2010     6    88797. 2010-06-01
```

```r
#Aggregate data by day

PJM_load_daily<- PJM_load %>%
  mutate(
    Year = year(Date),
    Month = month(Date),
    Day = day(Date)
  ) %>%
  group_by(Year, Month, Day) %>%
  summarise(
    Mean_Load = mean(Load_MW, na.rm = TRUE)
  )
```

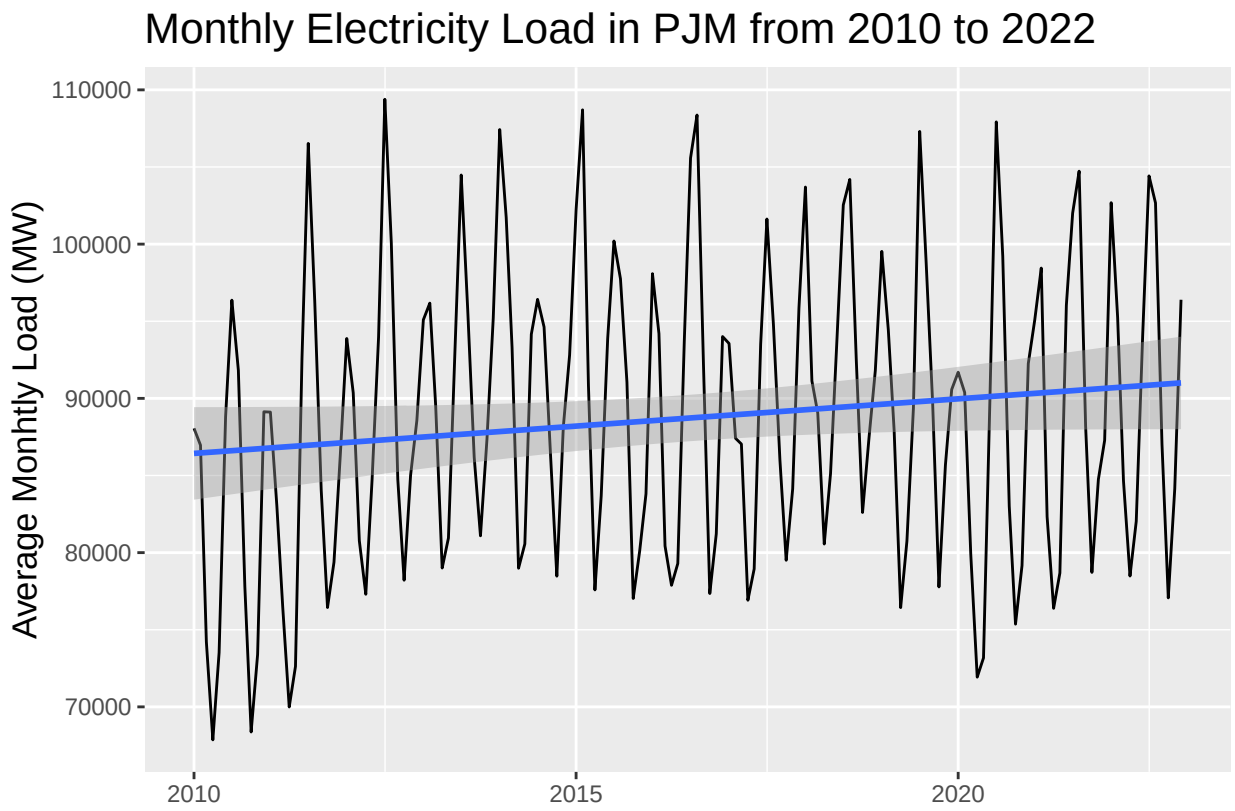
```
## `summarise()` has grouped output by 'Year', 'Month'. You can override using the
## `.groups` argument.
```

```r
PJM_load_daily$NewDate <-
  as.Date(paste(PJM_load_daily$Year, PJM_load_daily$Month,PJM_load_daily$Day, sep = "-"))

head(PJM_load_daily)
```

```
## # A tibble: 6 x 5
## # Groups:   Year, Month [1]
##    Year Month   Day Mean_Load NewDate
##   <dbl> <dbl> <int>     <dbl> <date>
## 1  2010     1     1    79483. 2010-01-01
## 2  2010     1     2    89741. 2010-01-02
## 3  2010     1     3    95061. 2010-01-03
## 4  2010     1     4    98474. 2010-01-04
## 5  2010     1     5    97021. 2010-01-05
## 6  2010     1     6    95613. 2010-01-06
```

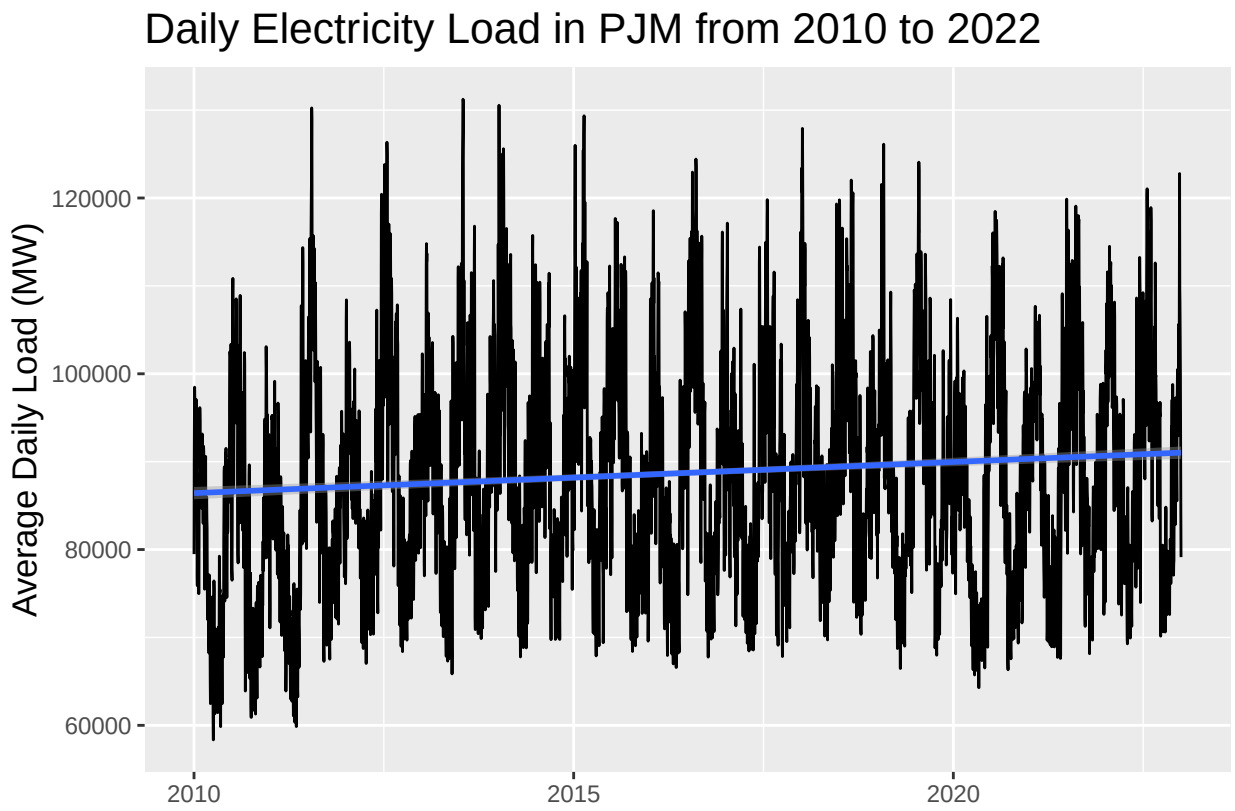
```r
PJMgraph_month<- ggplot(PJM_load_monthly, aes(x = NewDate, y = Mean_Load)) +
  geom_line(color = "black") +
  geom_smooth(method = "lm") +
  ggtitle('Monthly Electricity Load in PJM from 2010 to 2022') +
  labs(x = "", y = "Average Monhtly Load (MW)")

PJMgraph_month
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



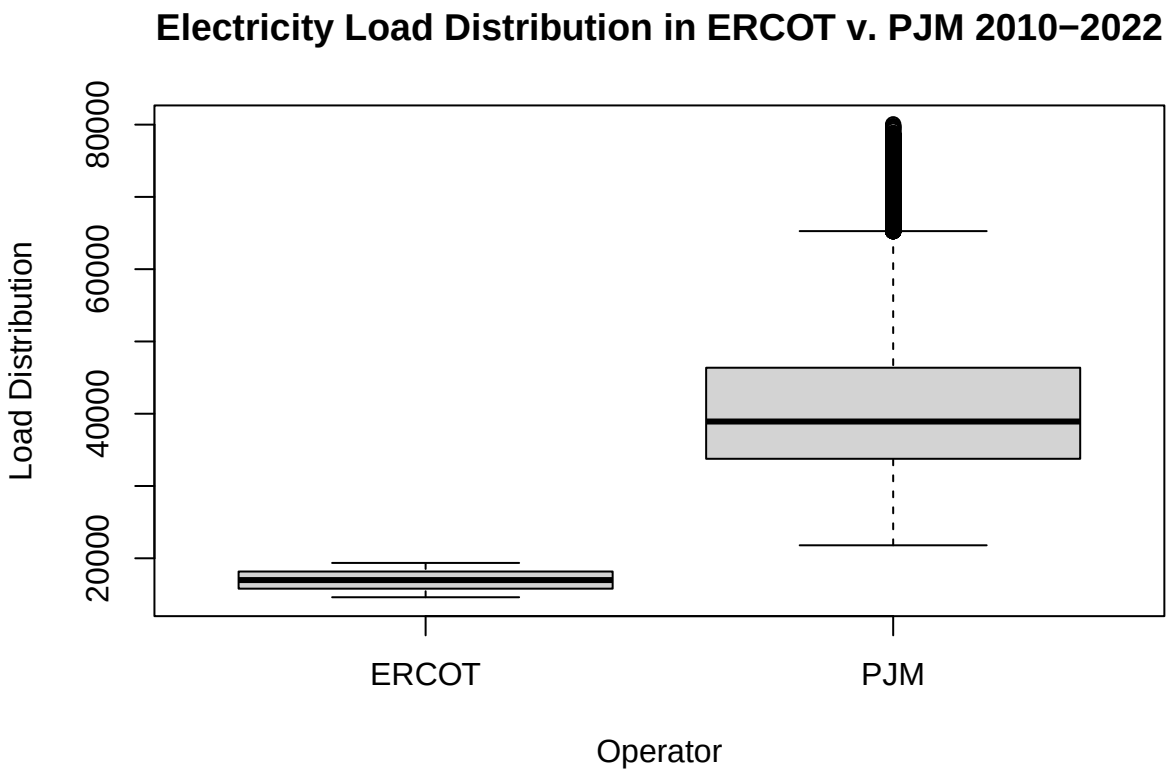

```
PJMgraph_daily <- ggplot(PJM_load_daily, aes(x = NewDate, y = Mean_Load)) +
  geom_line(color = "black") +
  geom_smooth(method = "lm") +
  ggtitle('Daily Electricity Load in PJM from 2010 to 2022') +
  labs(x = "", y = "Average Daily Load (MW)")

PJMgraph_daily
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Using a boxplot to compare electricity load distribution between ERCOT versus PJM.

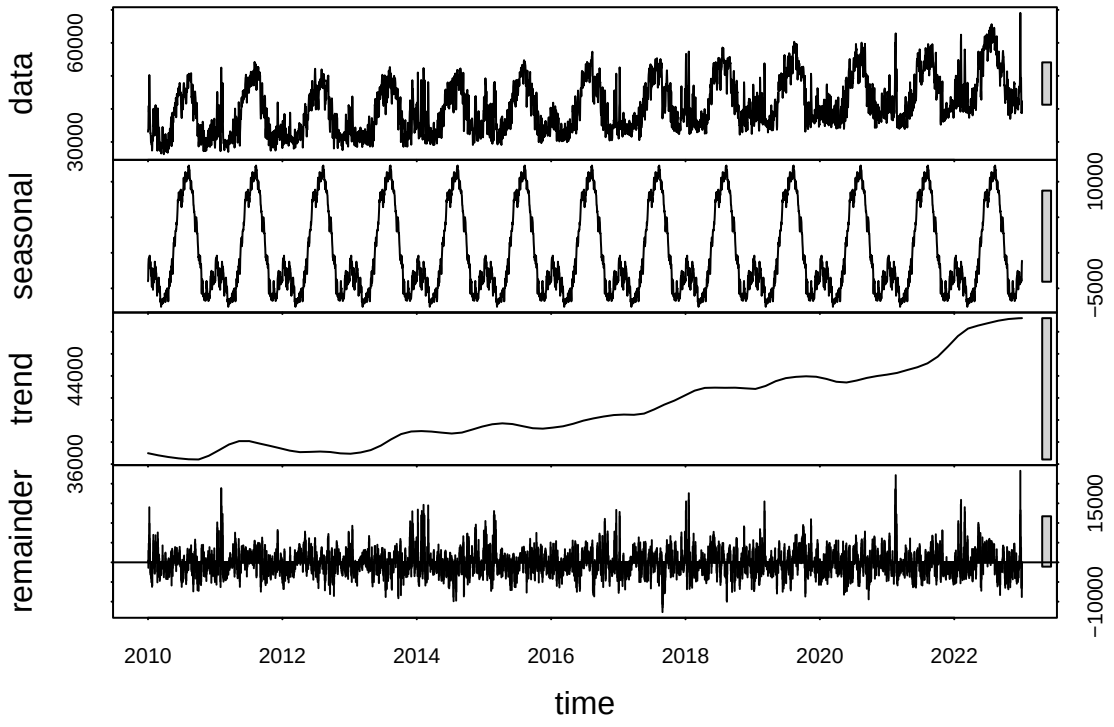
```
boxplot(ercot_load, PJM_load, names = c("ERCOT", "PJM"), xlab = "Operator", ylab = "Load Distribution",
```


Electricity Load Distribution in ERCOT v. PJM 2010−2022
Load Distribution
20000
40000
60000
80000
ERCOT
PJM
Operator

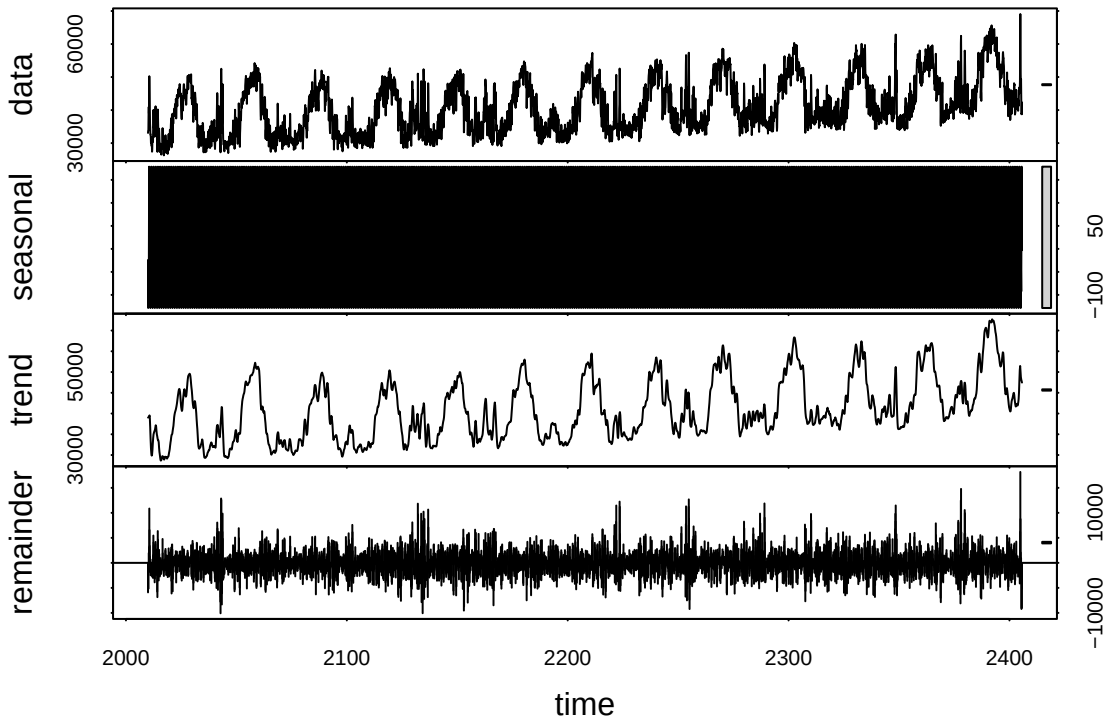
#Time Series Analysis The null hypothesis: Electricity load in ERCOT remained the same from 2010 to 2022. Alternative hypothesis: Electricity load in ERCOT has changed from 2010 to 2022.

```
#Generate ts dataframes

ercot_load_daily_ts <- ts(
  ercot_load_daily$Mean_Load,
  start = c(2010,1), frequency = 365
)

ercot_load_monthly_ts <- ts(
  ercot_load_daily$Mean_Load,
  start = c(2010,1),
  frequency = 12
)
```

```
#11 Decompose ts dfs and plot them

daily_decomposed <- stl(ercot_load_daily_ts, s.window="periodic")
plot(daily_decomposed)
```




```
monthly_decomposed <- stl(ercot_load_monthly_ts, s.window="periodic")
plot(monthly_decomposed)
```




```
monthly_load_trend <- Kendall::SeasonalMannKendall(ercot_load_monthly_ts)
summary(monthly_load_trend)
```

```
## Score =  287577 , Var(Score) = 82954344
## denominator =  937335.1
## tau = 0.307, 2-sided pvalue =< 2.22e-16
```

The seasonal Mann-Kendall is the most appropriate test because it is designed to analyze seasonal, non-parametric data, and there is a clear seasonal trend in our data. All other tests we learned about could only be used for non-seasonal data. A positive score (361963) and tau (0.341) indicate an increasing trend in the time series data. The p-value smaller than 0.05 allows us to reject the null hypothesis. Therefore, we can conclude that electricity demand has changed over time in ERCOT.

```
# Extract seasonal component from the time series

monthly_decomposed_components <- as.data.frame(monthly_decomposed$time.series[,1:3])

monthly_decomposed_deseasoned <-
  ercot_load_monthly_ts-monthly_decomposed_components$seasonal

# Run the Mann Kendall test on the non-seasonal data

monthly_load_trend_deseason <- Kendall::MannKendall(monthly_decomposed_deseasoned)
summary(monthly_load_trend_deseason)
```

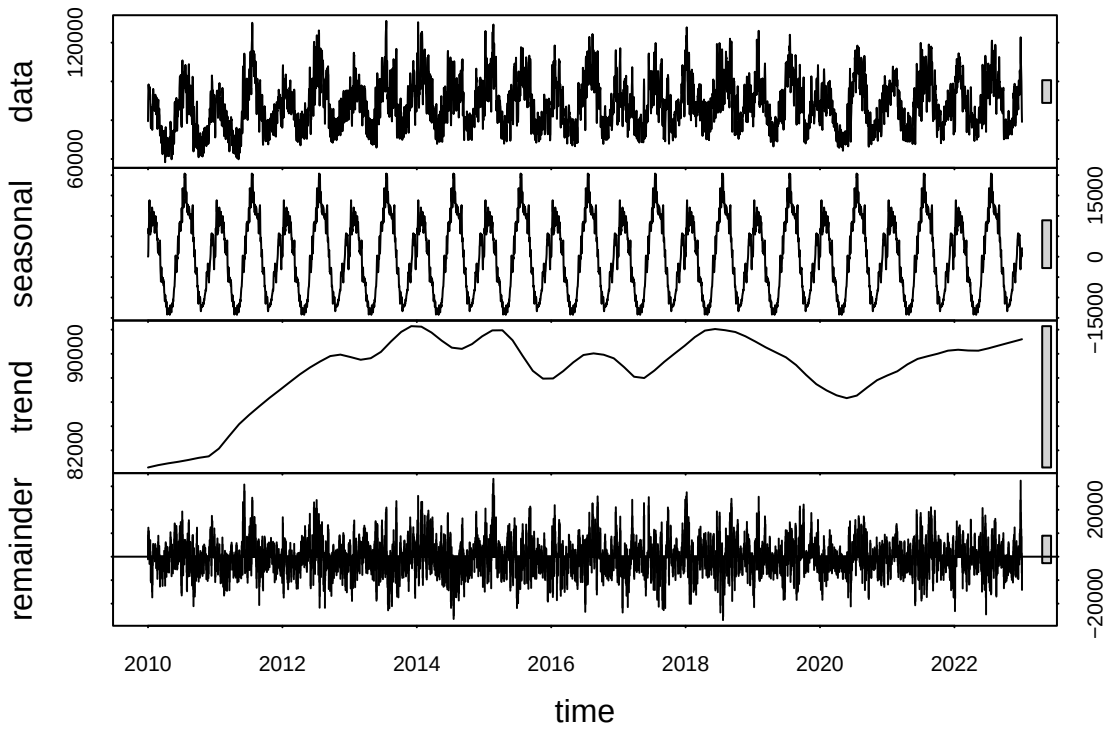
```
## Score =  3444826 , Var(Score) = 11904225280
## denominator =  11274125
## tau = 0.306, 2-sided pvalue =< 2.22e-16
```

Based on the results of the Mann Kendall test, we are able to reject the null hypothesis ($p<0.05$) and identify an increasing trend in the data (tau=0.34, score=4339158). Hence, the load in ERCOT has indeed changed over time; it was not simply a seasonal change. A p-value <0.05 in the seasonal Mann Kendall test also led us to the same conclusion.

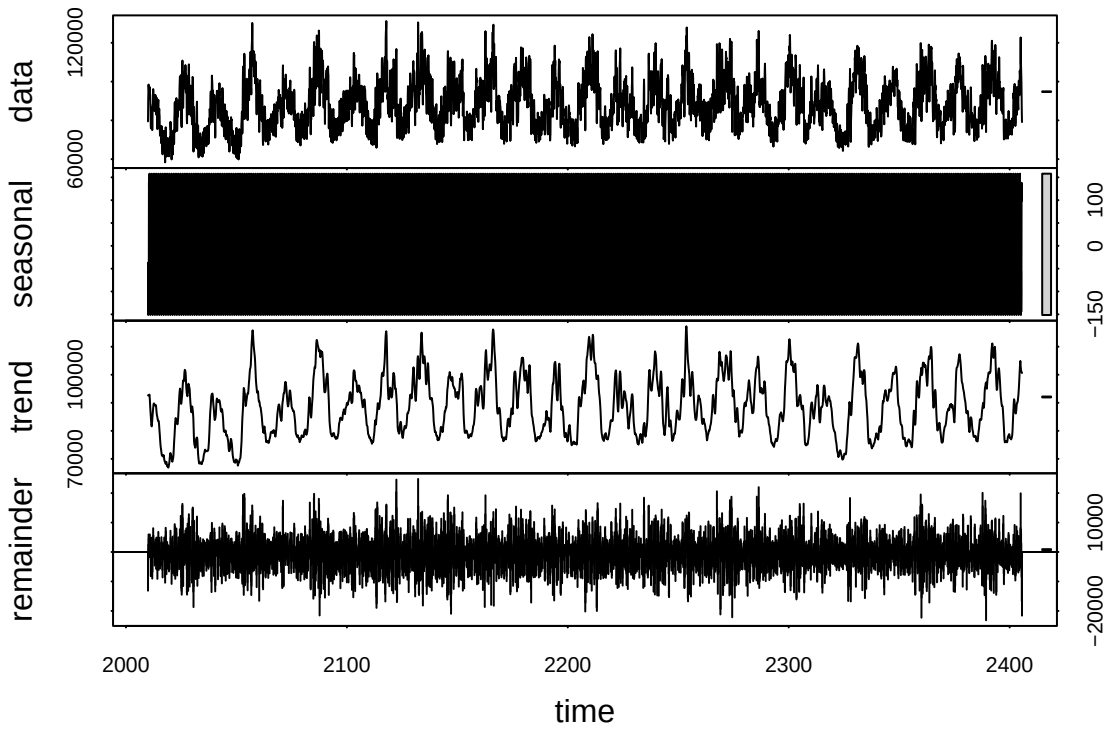
The null hypothesis: Electricity load in PJM remained the same from 2010 to 2022. Alternative hypothesis: Electricity load in PJM has changed from 2010 to 2022.

```
#Generate ts dataframes

PJM_load_daily_ts <- ts(
  PJM_load_daily$Mean_Load,
  start = c(2010,1), frequency = 365
)

PJM_load_monthly_ts <- ts(
  PJM_load_daily$Mean_Load,
  start = c(2010,1),
  frequency = 12
)
```

```
#11 Decompose ts dfs and plot them

daily_decomposed_PJM <- stl(PJM_load_daily_ts, s.window="periodic")
plot(daily_decomposed_PJM)
```




```
monthly_decomposed_PJM <- stl(PJM_load_monthly_ts, s.window="periodic")
plot(monthly_decomposed_PJM)
```




```
monthly_load_trend_PJM <- Kendall::SeasonalMannKendall(PJM_load_monthly_ts)
summary(monthly_load_trend_PJM)
```

```
## Score =  63799 , Var(Score) = 82902071
## denominator =  936939.6
## tau = 0.0681, 2-sided pvalue =2.4352e-12
```

The seasonal Mann-Kendall is the most appropriate test because it is designed to analyze seasonal, non-parametric data, and there is a clear seasonal trend in our data. All other tests we learned about could only be used for non-seasonal data. A positive score (82902071) and tau (0.0681) indicate an increasing trend in the time series data. The p-value smaller than 0.05 allows us to reject the null hypothesis. Therefore, we can conclude that electricity demand has changed over time in PJM.

```
# Extract seasonal component from the time series

monthly_decomposed_components_PJM <- as.data.frame(monthly_decomposed_PJM$time.series[,1:3])

monthly_decomposed_deseasoned_PJM <-
  PJM_load_monthly_ts-monthly_decomposed_components_PJM$seasonal

# Run the Mann Kendall test on the non-seasonal data

monthly_load_trend_deseason_PJM <- Kendall::MannKendall(monthly_decomposed_deseasoned_PJM)
summary(monthly_load_trend_deseason_PJM)
```

```
## Score =  774030 , Var(Score) = 11896706048
## denominator =  11269379
## tau = 0.0687, 2-sided pvalue =< 2.22e-16
```

Mann Kendall test rejects the null hypothesis, which indicates that there is an increasing trend in the data (tau=0.0687, score=11896706048). This means that PJM load has changed over time from 2010 to 2022 without seasonality components. A p-value <0.05 in the seasonal Mann Kendall test also led us to the same conclusion.

```
growth_PJM <- diff(PJM_load_monthly_ts) / stats::lag(PJM_load_monthly_ts)
#growth_ERCOT <- diff(PJM_load_monthly_ts) / stats::lag(PJM_load_monthly_ts)

#plot(Date[-1], growth_PJM, type = "l", col = "blue", ylab = "Growth Rate", xlab = "Time")
#lines(Date[-1], growth_ERCOT, type = "l", col = "red")
```

#Regression Analysis ## PJM Simple Linear Regression

Simple Linear Regression

The null hypothesis: There is no change in electricity load with price in PJM from years...

Alternative hypothesis: Electricity load in PJM has changed in relation to price from years...

Real-time pricing reflects the actual market conditions, while day-ahead pricing provides insight into the expected future market conditions. Both pricing data used to understand regression analysis. Market Performance Clearing Price ($/MWh)used reflecting the cost of the movement of load in response to regulation signals, which is related to the performance of the resource in meeting the demand.

```
# Read the CSV files
rt_2012<- read_csv("Data/Raw/PJM_Source_Data/rt_2012.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 2209 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (7): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
## lgl (2): total_pjm_rmccp_cr, total_pjm_rmpcp_cr
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2013<- read_csv("Data/Raw/PJM_Source_Data/rt_2013.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (7): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
## lgl (2): total_pjm_rmccp_cr, total_pjm_rmpcp_cr
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2014<- read_csv("Data/Raw/PJM_Source_Data/rt_2014.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (7): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
## lgl (2): total_pjm_rmccp_cr, total_pjm_rmpcp_cr
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2015<- read_csv("Data/Raw/PJM_Source_Data/rt_2015.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (7): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
## lgl (2): total_pjm_rmccp_cr, total_pjm_rmpcp_cr
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2016<- read_csv("Data/Raw/PJM_Source_Data/rt_2016.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8783 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (7): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
## lgl (2): total_pjm_rmccp_cr, total_pjm_rmpcp_cr
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2017<- read_csv("Data/Raw/PJM_Source_Data/rt_2017.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (7): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
```

```
## lgl (2): total_pjm_rmccp_cr, total_pjm_rmpcp_cr
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2018<- read_csv("Data/Raw/PJM_Source_Data/rt_2018.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (9): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2019<- read_csv("Data/Raw/PJM_Source_Data/rt_2019.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (9): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2020<- read_csv("Data/Raw/PJM_Source_Data/rt_2020.csv")%>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8783 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (9): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2021<- read_csv("Data/Raw/PJM_Source_Data/rt_2021.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
```

```
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (9): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
rt_2022<- read_csv("Data/Raw/PJM_Source_Data/rt_2022.csv") %>%
  rename(Date = datetime_beginning_ept, Price = rmpcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (9): rmccp, rmpcp, total_pjm_rt_load_mwh, total_pjm_loc_credit, total_pj...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#Merge the data frames based on a common column
rt_load <- bind_rows(rt_2012, rt_2013, rt_2014, rt_2015, rt_2016, rt_2017, rt_2018, rt_2019, rt_2020, r
```

```
# Read the CSV files
dasr_2010<- read_csv("Data/Raw/PJM_Source_Data/dasr_2010.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2011<- read_csv("Data/Raw/PJM_Source_Data/dasr_2011.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2012<- read_csv("Data/Raw/PJM_Source_Data/dasr_2012.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8783 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2013<- read_csv("Data/Raw/PJM_Source_Data/dasr_2013.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2014<- read_csv("Data/Raw/PJM_Source_Data/dasr_2014.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2015<- read_csv("Data/Raw/PJM_Source_Data/dasr_2015.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2016<- read_csv("Data/Raw/PJM_Source_Data/dasr_2016.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8783 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2017<- read_csv("Data/Raw/PJM_Source_Data/dasr_2017.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2018<- read_csv("Data/Raw/PJM_Source_Data/dasr_2018.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2019<- read_csv("Data/Raw/PJM_Source_Data/dasr_2019.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2020<- read_csv("Data/Raw/PJM_Source_Data/dasr_2020.csv")%>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8783 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2021<- read_csv("Data/Raw/PJM_Source_Data/dasr_2021.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 8759 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dasr_2022<- read_csv("Data/Raw/PJM_Source_Data/dasr_2022.csv") %>%
  rename(Date = datetime_beginning_ept, Price = dasrmcp, Load_MW = total_pjm_rt_load_mwh) %>%
  mutate(Date = as.Date(Date, format = "%m/%d/%Y"))
```

```
## Rows: 6550 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): datetime_beginning_utc, datetime_beginning_ept, datetime_ending_utc...
## dbl (5): dasrmcp, total_pjm_rt_load_mwh, total_pjm_cleared_dasr_mwh, total_p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

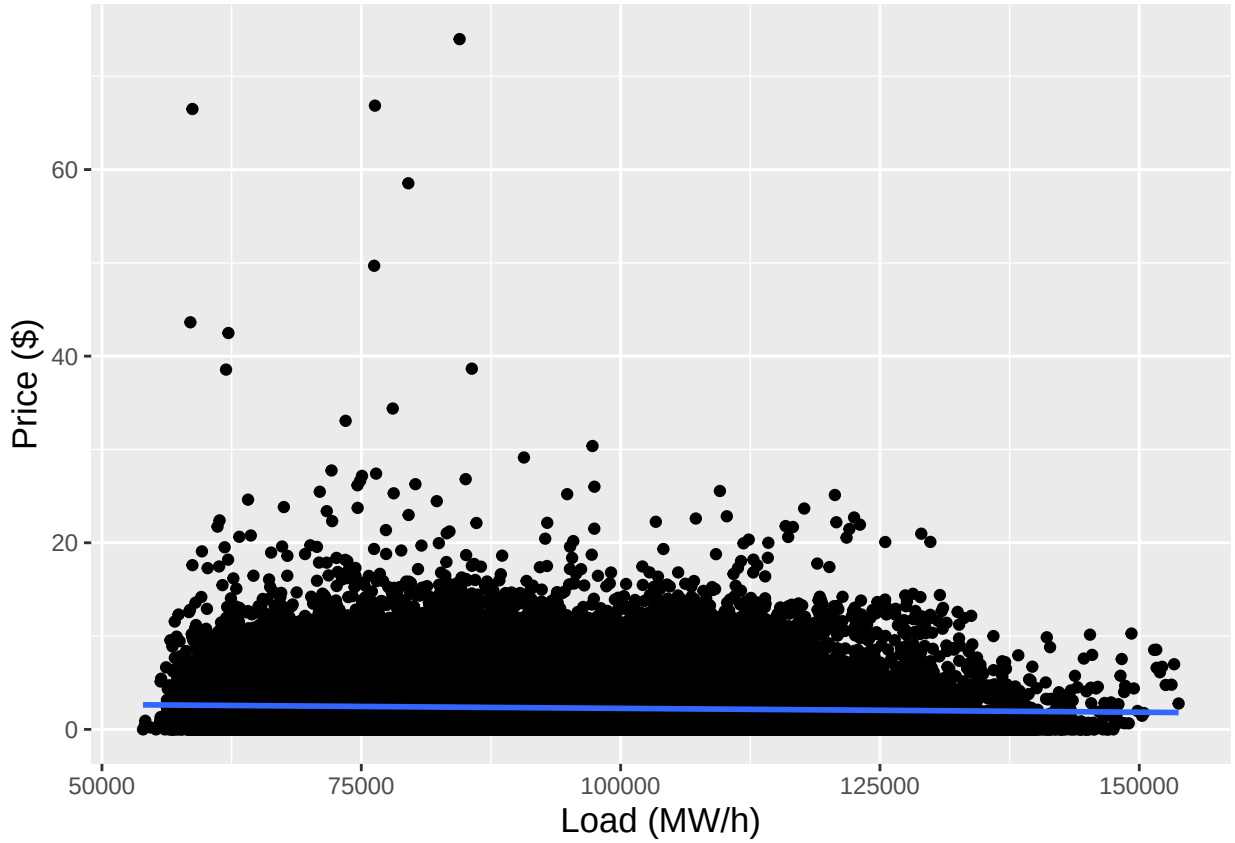
```
#Merge the data frames based on a common column
dasr_load <- bind_rows(dasr_2010, dasr_2011, dasr_2012, dasr_2013, dasr_2014, dasr_2015, dasr_2016, das
```

```
ggplot(rt_load, aes(x= Load_MW, y= Price)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x= "Load (MW/h)", y = "Price ($)")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

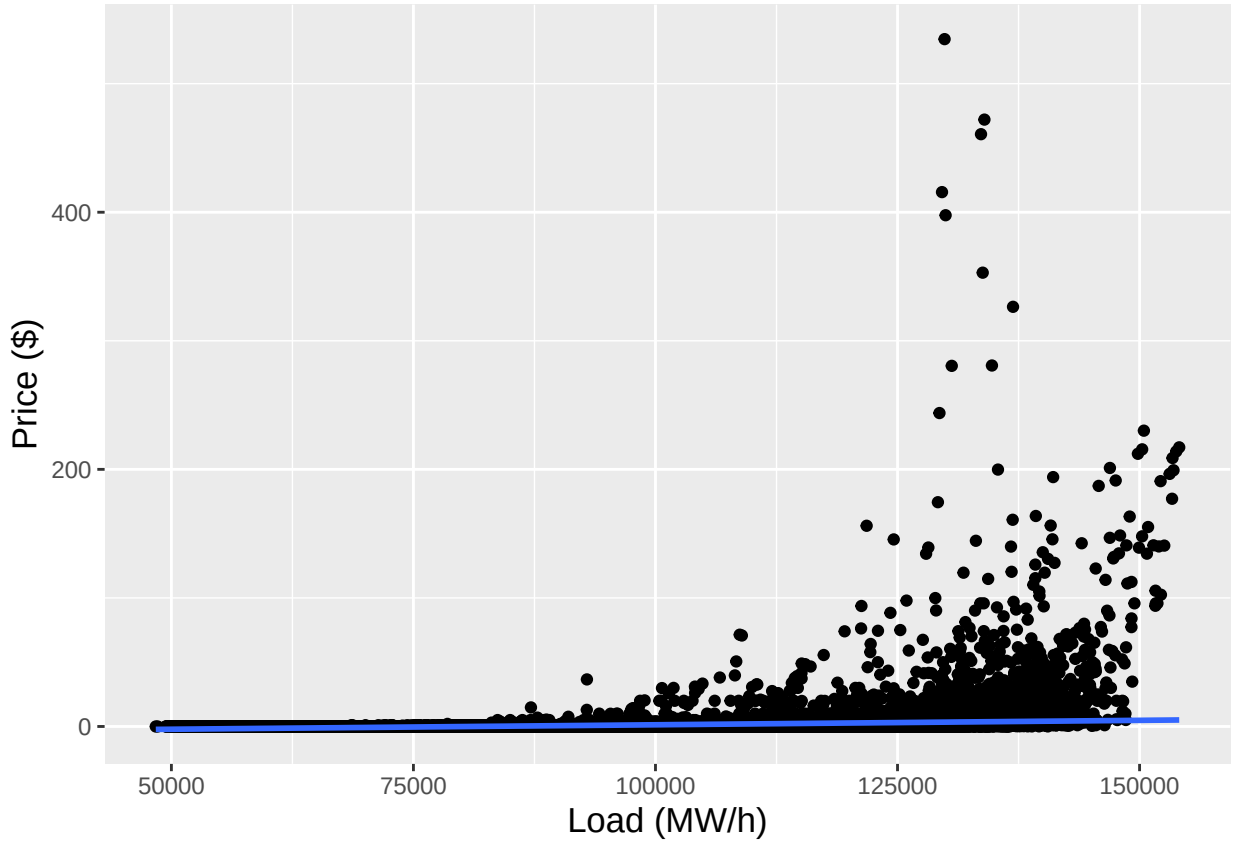
```
## Warning: Removed 8760 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 8760 rows containing missing values (`geom_point()`).
```




```
ggplot(dasr_load, aes(x= Load_MW, y= Price)) +
  geom_point() + geom_smooth(method = "lm", se = FALSE) +
  labs(x= "Load (MW/h)", y = "Price ($)")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
rt.regression <- lm(data = rt_load, Price ~ Load_MW)
summary(rt.regression)
```

```
##
## Call:
## lm(formula = Price ~ Load_MW, data = rt_load)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.632 -1.810 -0.975  1.017 71.599
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.078e+00  5.320e-02   57.85   <2e-16 ***
## Load_MW     -8.254e-06  5.960e-07  -13.85   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.711 on 81086 degrees of freedom
##   (8760 observations deleted due to missingness)
## Multiple R-squared:  0.002359,   Adjusted R-squared:  0.002347
## F-statistic: 191.8 on 1 and 81086 DF,  p-value: < 2.2e-16
```

```
cor.test(rt_load$Price, rt_load$Load_MW)
```

```
##
##  Pearson's product-moment correlation
##
## data:  rt_load$Price and rt_load$Load_MW
## t = -13.848, df = 81086, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.05543753 -0.04170422
## sample estimates:
##         cor
## -0.04857317
```

```
dasr.regression <- lm(data = dasr_load, Price ~ Load_MW)
summary(dasr.regression)
```

```
##
## Call:
## lm(formula = Price ~ Load_MW, data = dasr_load)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -4.07  -0.95  -0.21   0.45 531.28
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.607e+00  9.390e-02  -59.71   <2e-16 ***
## Load_MW      6.921e-05  1.061e-06   65.22   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.777 on 111728 degrees of freedom
## Multiple R-squared:  0.03668,    Adjusted R-squared:  0.03667
## F-statistic:  4254 on 1 and 111728 DF,  p-value: < 2.2e-16
```

```
cor.test(dasr_load$Price, dasr_load$Load_MW)
```

```
##
##  Pearson's product-moment correlation
##
## data:  dasr_load$Price and dasr_load$Load_MW
## t = 65.224, df = 111728, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1858631 0.1971601
## sample estimates:
##      cor
## 0.191518
```

ERCOT Multiple Linear Regression

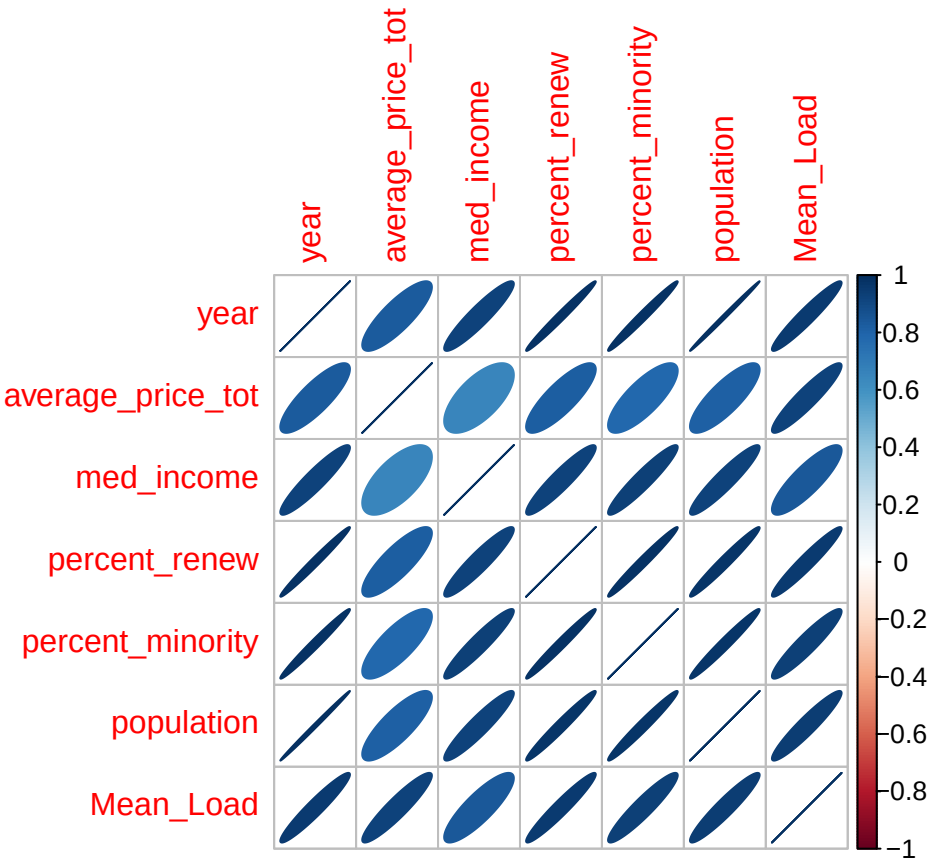
```
ercot_linear_model_data <- read.csv("Data/Raw/ERCOT_Source_Data/ERCOTLinearModelData.csv")
ercot_linear_model_data <- left_join(ercot_linear_model_data, ercot_load_yearly) %>% drop_na()
```

```
## Joining with `by = join_by(year)`
```

```
ercot_linear_model_data$percent_renew <- stringr::str_extract(ercot_linear_model_data$percent_renew, "^
  as.numeric()

ercot_linear_model_data$percent_minority <- ercot_linear_model_data$percent_minority %>%
  as.numeric()

ercot_linear_model_data$population <- as.numeric(gsub(",","",ercot_linear_model_data$population))

ercot_linear_model_data
```

```
##    year average_price_res average_price_com average_price_ind average_price_tot
## 1  2010             11.54             10.19              6.77              9.83
## 2  2011             11.72             10.24              6.82              9.90
## 3  2012             11.88             10.09              6.67              9.84
## 4  2013             12.13             10.26              6.89             10.07
## 5  2014             12.52             10.74              7.10             10.44
## 6  2015             12.65             10.64              6.91             10.41
## 7  2016             12.55             10.43              6.76             10.27
## 8  2017             12.89             10.66              6.88             10.48
## 9  2018             12.87             10.67              6.92             10.53
## 10 2019             13.01             10.68              6.81             10.54
## 11 2020             13.15             10.59              6.67             10.59
## 12 2021             13.66             11.22              7.18             11.10
## 13 2022             15.04             12.41              8.32             12.36
##    med_income percent_renew percent_minority population Mean_Load
## 1       61680          7.39             54.7   25241897  36335.65
## 2       62080          7.63             55.1   25645504  38127.12
## 3       64480          8.74             55.6   26084120  37000.04
## 4       63070          8.75             56.0   26479646  37869.12
## 5       65170         10.10             56.0   26963092  38828.39
## 6       68360         12.50             57.0   27468531  39669.23
## 7       69770         14.50             57.0   27914064  40006.36
## 8       70860         15.80             58.0   28291024  40781.92
## 9       69100         17.40             58.4   28624564  42949.25
## 10      76820         20.10             58.8   28986794  43817.93
## 11      77110         22.10             60.3   29232474  43463.07
## 12      72680         23.90             59.7   29558864  44828.81
## 13      74640         25.00             60.2   30029572  49074.36
```

```
ercot_linear_model_data_totprice <- ercot_linear_model_data %>%
  select(-c(average_price_com, average_price_ind, average_price_res))

ercot.tot.Corr <- cor(ercot_linear_model_data_totprice)
corrplot(ercot.tot.Corr, method = 'ellipse')
```




```
ercot.AIC.total <- lm(data = ercot_linear_model_data_totprice,
                      Mean_Load ~ average_price_tot + population + percent_renew + med_income +
                       percent_minority)

step(ercot.AIC.total)
```

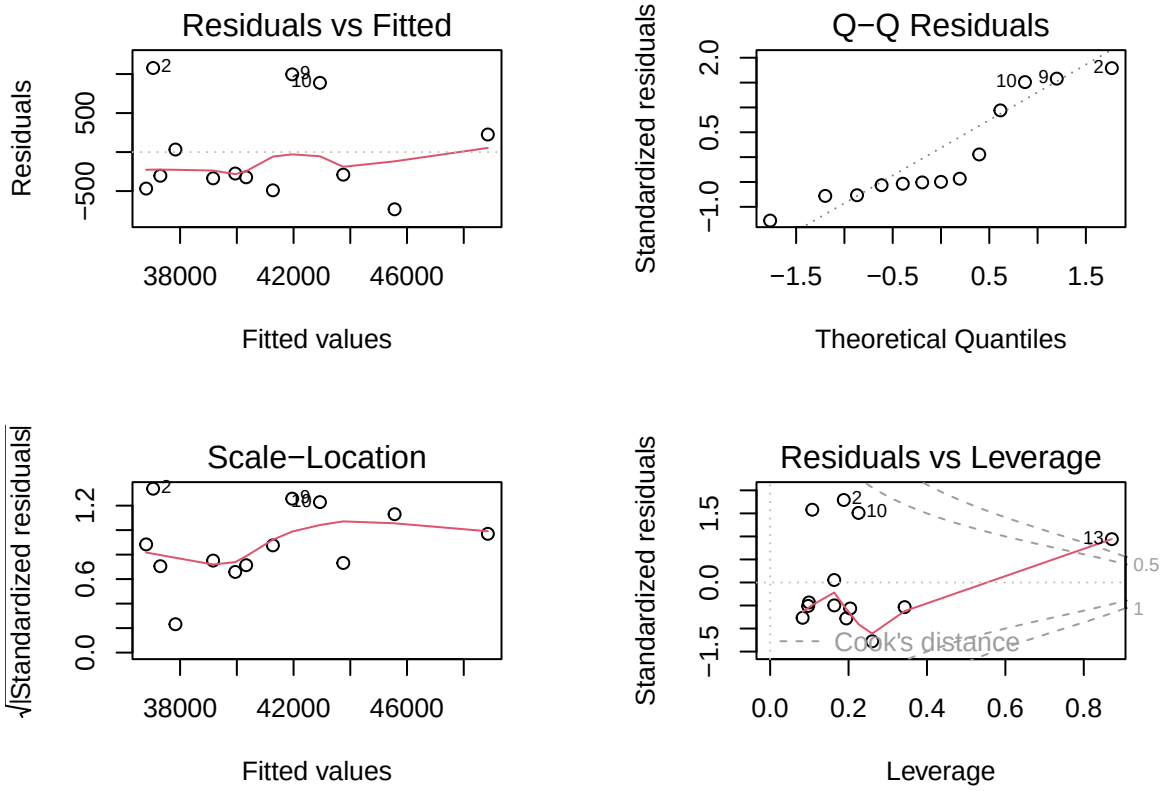
```
## Start:  AIC=177.64
## Mean_Load ~ average_price_tot + population + percent_renew +
##     med_income + percent_minority
##
##                     Df Sum of Sq      RSS    AIC
## - percent_minority   1       961  4442811 175.64
## - population         1      4576  4446425 175.66
## - med_income         1      7472  4449321 175.66
## <none>                             4441849 177.64
## - percent_renew      1   1243905  5685754 178.85
## - average_price_tot  1   6162527 10604376 186.95
##
## Step:  AIC=175.64
## Mean_Load ~ average_price_tot + population + percent_renew +
##     med_income
##
##                     Df Sum of Sq      RSS    AIC
## - population         1      3624  4446435 173.66
## - med_income         1      6798  4449609 173.66
```

```
## <none>                              4442811 175.64
## - percent_renew      1   1911430  6354241 178.30
## - average_price_tot  1   6441468 10884279 185.29
##
## Step:  AIC=173.65
## Mean_Load ~ average_price_tot + percent_renew + med_income
##
##                     Df Sum of Sq      RSS    AIC
## - med_income         1     13525  4459959 171.69
## <none>                             4446435 173.66
## - percent_renew      1   3236589  7683024 178.76
## - average_price_tot  1   7110458 11556893 184.07
##
## Step:  AIC=171.69
## Mean_Load ~ average_price_tot + percent_renew
##
##                     Df Sum of Sq      RSS    AIC
## <none>                             4459959 171.69
## - average_price_tot  1   9019288 13479248 184.07
## - percent_renew      1  19338242 23798201 191.46


##
## Call:
## lm(formula = Mean_Load ~ average_price_tot + percent_renew, data = ercot_linear_model_data_totprice)
##
## Coefficients:
##       (Intercept)  average_price_tot      percent_renew
##           11580.9             2300.1              353.6
```

```
ercot.totalavg.regression <- lm(data = ercot_linear_model_data_totprice,
                      Mean_Load ~ average_price_tot + percent_renew)

summary(ercot.totalavg.regression)
```

```
##
## Call:
## lm(formula = Mean_Load ~ average_price_tot + percent_renew, data = ercot_linear_model_data_totprice)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -733.5 -336.3 -289.7  225.1 1077.8
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        11580.9     4729.9   2.448  0.03435 *
## average_price_tot   2300.1      511.5   4.497  0.00115 **
## percent_renew        353.6       53.7   6.585  6.2e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 667.8 on 10 degrees of freedom
## Multiple R-squared:  0.9723, Adjusted R-squared:  0.9668
## F-statistic: 175.5 on 2 and 10 DF,  p-value: 1.632e-08
```

```
par(mfrow = c(2,2), mar=c(4,4,4,4))
plot(ercot.totalavg.regression)
```




```
par(mfrow = c(1,1))
```

Summary and Conclusions

References

<add references here if relevant, otherwise delete this section>