Final Project

By: Katie Owens

4/2/2022

```
getwd()
```

```
## [1] "C:/Users/Katherine/Documents/872-Data Analytics/KinserOwensWhite_ENV872_EDA_FinalProject/Code"
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.3

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

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

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.1.3

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v stringr 1.4.0
## v tidyr   1.2.0     v forcats 0.5.1
## v readr   2.1.2

## Warning: package 'ggplot2' was built under R version 4.1.3

## Warning: package 'tibble' was built under R version 4.1.3

## Warning: package 'tidyr' was built under R version 4.1.3

## Warning: package 'readr' was built under R version 4.1.3

## Warning: package 'purrr' was built under R version 4.1.3

## Warning: package 'stringr' was built under R version 4.1.3

## Warning: package 'forcats' was built under R version 4.1.3

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.1.3

##
## Attaching package: 'lubridate'

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

```
library(tidyr)
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 4.1.3

##
## Attaching package: 'zoo'

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

```
library(sf)
```

```
## Warning: package 'sf' was built under R version 4.1.3

## Linking to GEOS 3.9.1, GDAL 3.2.1, PROJ 7.2.1; sf_use_s2() is TRUE
```

```
library(trend)
```

```
## Warning: package 'trend' was built under R version 4.1.3
```

```
library(zoo)
library(Kendall)
```

```
## Warning: package 'Kendall' was built under R version 4.1.3
```

```
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 4.1.3

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

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 4.1.3

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following object is masked from 'package:purrr':
##
##     transpose

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```r
mytheme <- theme_classic(base_size = 14) +
  theme(axis.text = element_text(color = "black"),
        legend.position = "right") +
  theme_bw()
theme_set(mytheme)
```

```r
climate<-read.csv("../Data/Processed/BlueMesaClimate.csv")
power<-read.csv("../Data/Processed/BlueMesaPower.csv")
reservoir<-read.csv("../Data/Processed/BlueMesaReservoir.csv")
```

```r
#getting dates in the proper format
climate$Date <-as.Date(climate$Date, format = "%Y-%m-%d")
# climate <-climate %>%
#   mutate(Year = year(Date)) %>%
#   mutate(Month = rep(1:12, 19)) %>%
#   select(-c(Date))
# climate$Date<-sprintf("%d-%02d", climate$Year, climate$Month)
# climate<-climate %>%
#   select(-c(Year, Month))

# power<-power %>%
#   rename(Date = ï..Date)%>%
#   subset(select = c(Date, MWH))

power$Date <- as.Date(power$Date, format = "%Y-%m-%d")
# power<-power %>%
#   mutate(Month = month(Date)) %>%
#   mutate(Year = year(Date))%>%
#   select(-c(Date))
# power$Date<-sprintf("%d-%02d", power$Year, power$Month)
# power<- power %>%
#   select(-c(Year, Month))

# col_order<-c("Date", "MWH")
# power<-power[, col_order]

reservoir$Date <- as.Date(reservoir$Date, format = "%d-%b-%y")
# reservoir<-reservoir %>%
#   mutate(Month = month(Date)) %>%
#   mutate(Year = year(Date))%>%
#   select(-c(Date))
# reservoir$Date<-sprintf("%d-%02d", reservoir$Year, reservoir$Month)
# reservoir<-reservoir %>%
#   select(-c(Year, Month))

#make a gathered data set for data viz
climate.longer<-pivot_longer(climate, MaxT:MinT, names_to = "Min/Max", values_to = "Temperature")

#make a gathered data set for data viz
reservoir.longer<-reservoir %>%
  pivot_longer(Inflow.cfs:Total.cfs, names_to = "Flow_Type", values_to = "Flow_(cfs)") %>%
  pivot_longer(Storage.af:Evaporation.af, names_to = "Volume_Type", values_to = "Volume_(af)")

#combine power and reservoir data
```

```r
all.data<-left_join(reservoir, climate)
```

```
## Joining, by = "Date"
```

```r
all.data<-left_join(all.data, power)
```

```
## Joining, by = "Date"
```

```r
write.csv(all.data, "AllData.csv")
```

Time Series

Research Question: Has electricity generation (MWh) output of reservoir changed over time?

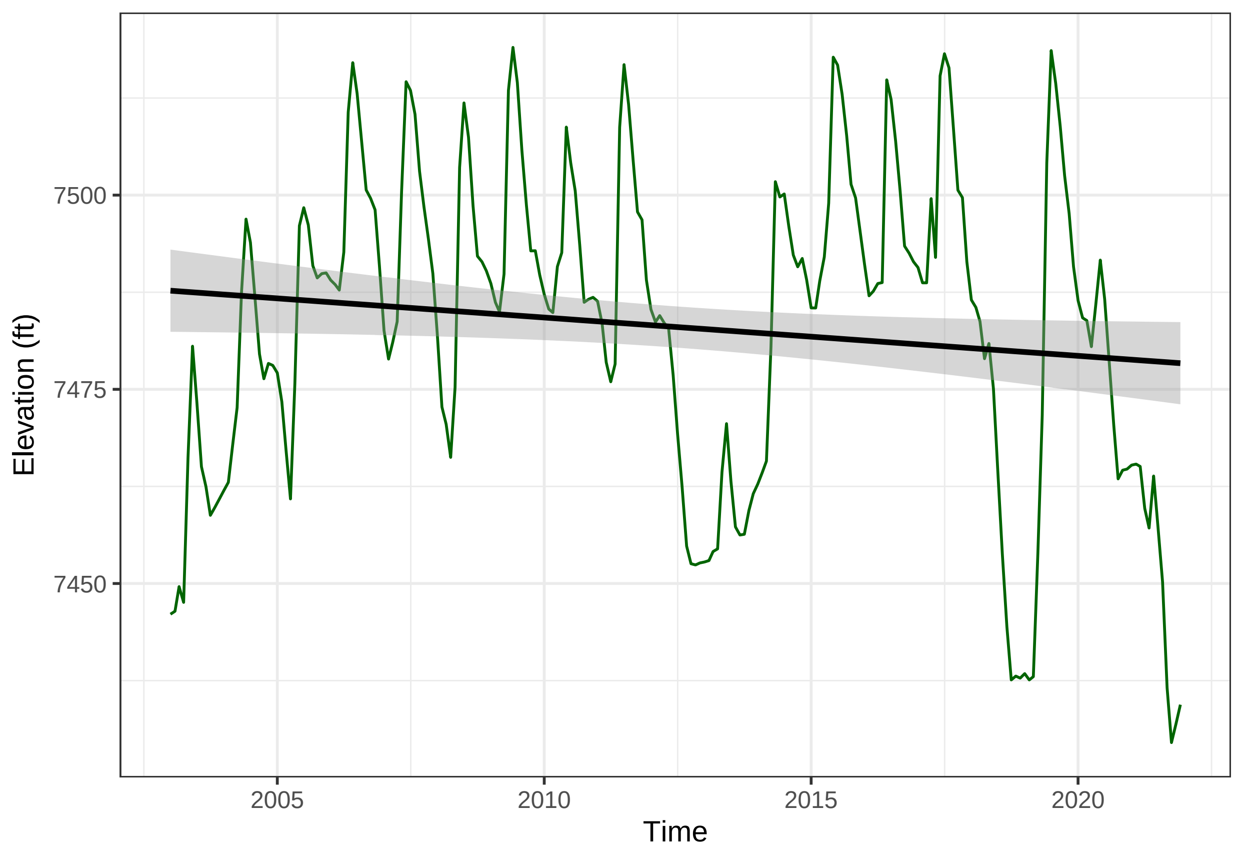
```r
#create new monthly data frame with all months (in case any were missing)
all_months <- as.data.frame(seq(as.Date("2003-01-01"), as.Date("2021-12-01"), "month"))

#rename single column in all_months to Date
colnames(all_months) <- c("Date")

#combine df with ALL days with monthly df
All_months.data <-
  left_join(
  all_months,
  all.data)
```

```
## Joining, by = "Date"
```

```r
#checked data frame and should have 227 months, check!
```

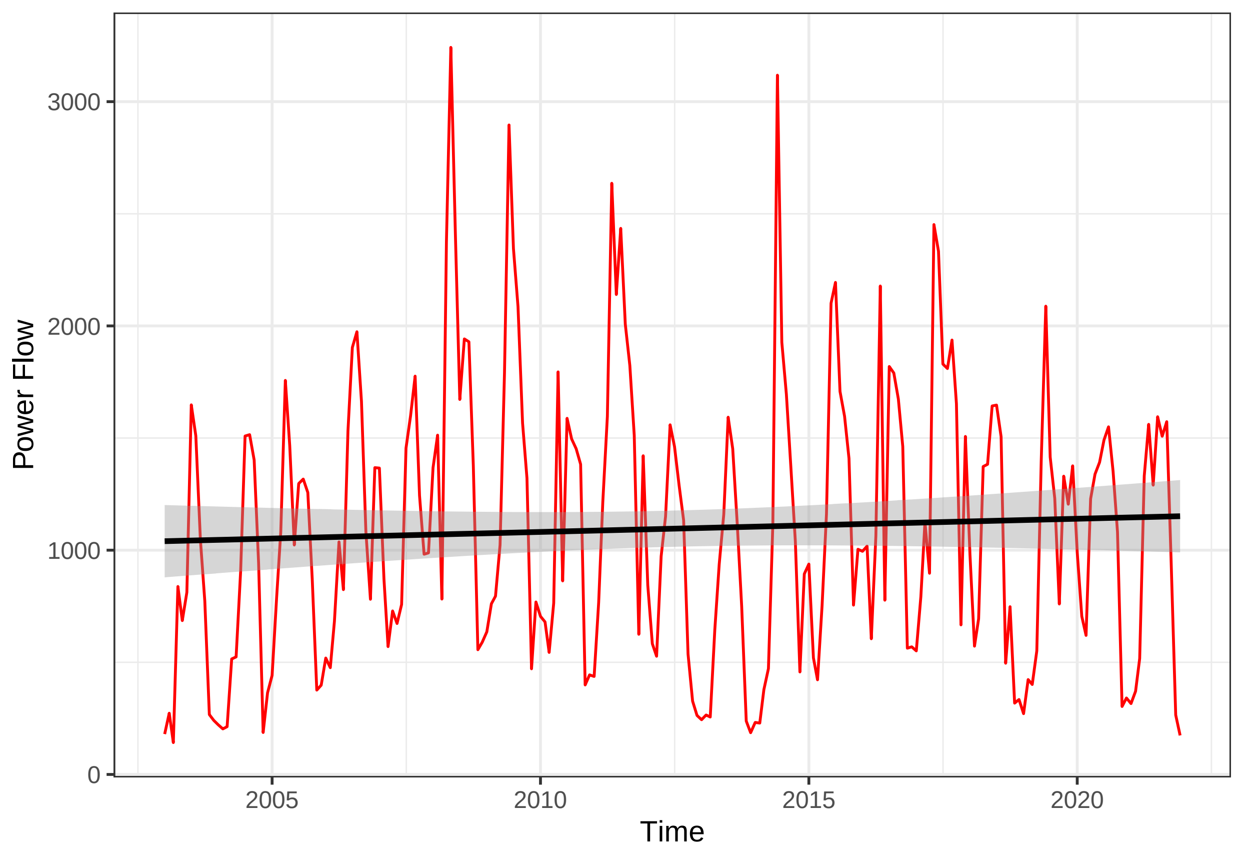
```r
#Explore Elevation, power flow, Min/MaxT, MWH Variables


#Elevation   #analyzing Elevation variable over time
ts.elev.plot <- ggplot(All_months.data, aes(x = Date, y = Elevation.ft)) + #analyzing elevation variabl
  geom_line(color = "darkgreen") +
  labs(x = "Time", y = expression("Elevation (ft)")) +
  geom_smooth(method = lm, color = "black") #add a trendline
print(ts.elev.plot)
```

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



```
#Power Flow   #analyzing Power Flow variable over time
ts.power.plot <- ggplot(All_months.data, aes(x = Date, y = Power.cfs)) + #analyzing power variable over
  geom_line(color = "red") +
  labs(x = "Time", y = expression("Power Flow")) +
  geom_smooth(method = lm, color = "black")
print(ts.power.plot)
```

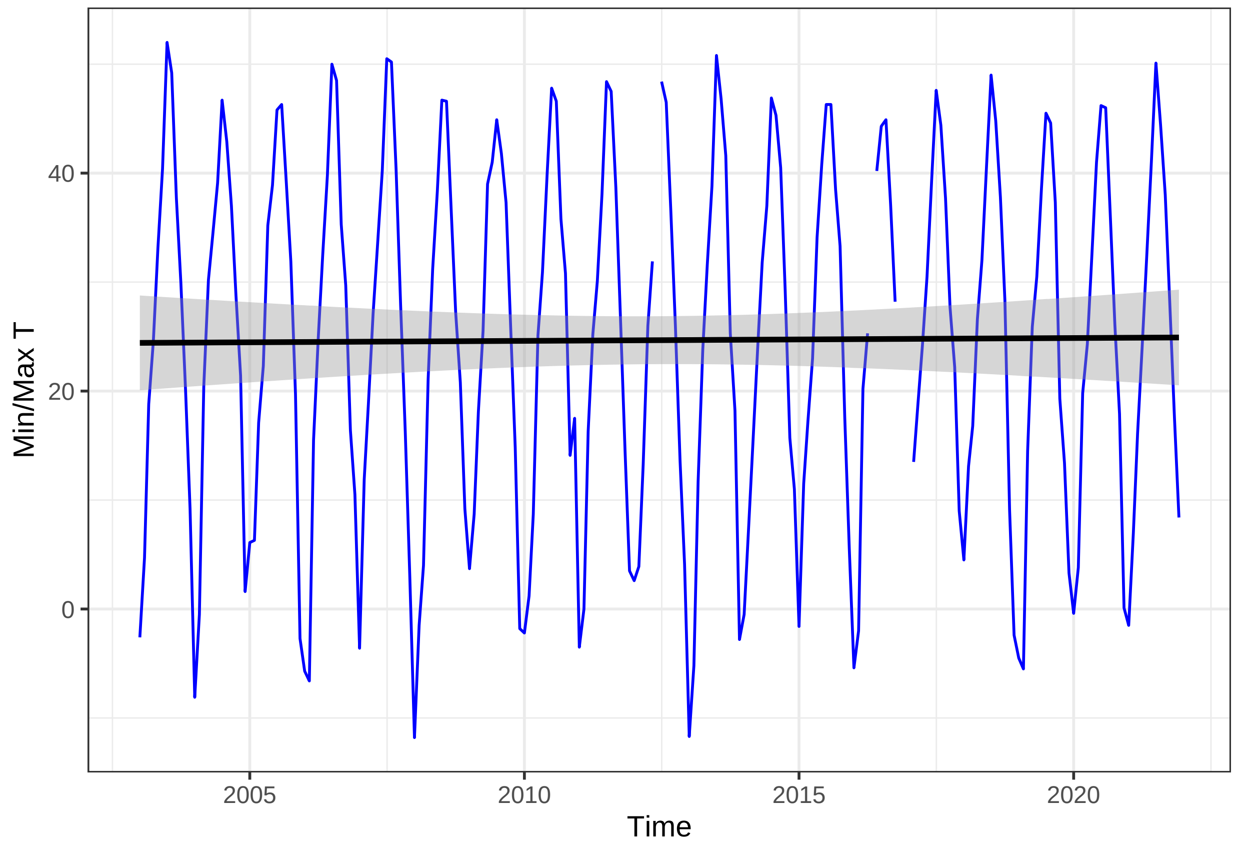
```
## `geom_smooth()` using formula 'y ~ x'
```



```
#Min T   #analyzing Min T variable over time
ts.min_t.plot <- ggplot(All_months.data, aes(x = Date, y = MinT)) +
  geom_line(color = "blue") +
  labs(x = "Time", y = expression("Min/Max T")) +
  geom_smooth(method = lm, color = "black")
print(ts.min_t.plot)
```

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

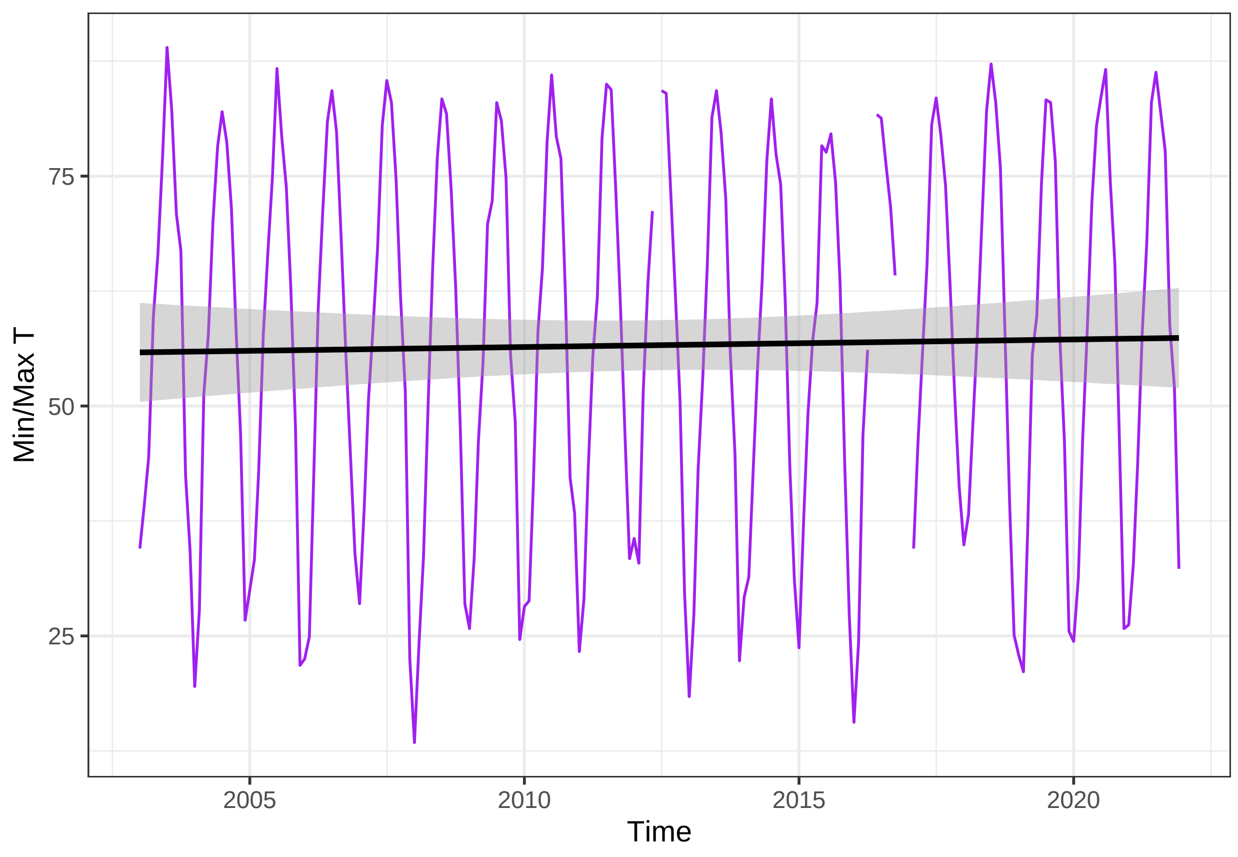
```
## Warning: Removed 4 rows containing non-finite values (stat_smooth).
```



```r
#Max T   #analyzing Max T variable over time
ts.max_t.plot <- ggplot(All_months.data, aes(x = Date, y = MaxT)) +
  geom_line(color = "purple") +
  labs(x = "Time", y = expression("Min/Max T")) +
  geom_smooth(method = lm, color = "black")
print(ts.max_t.plot)
```

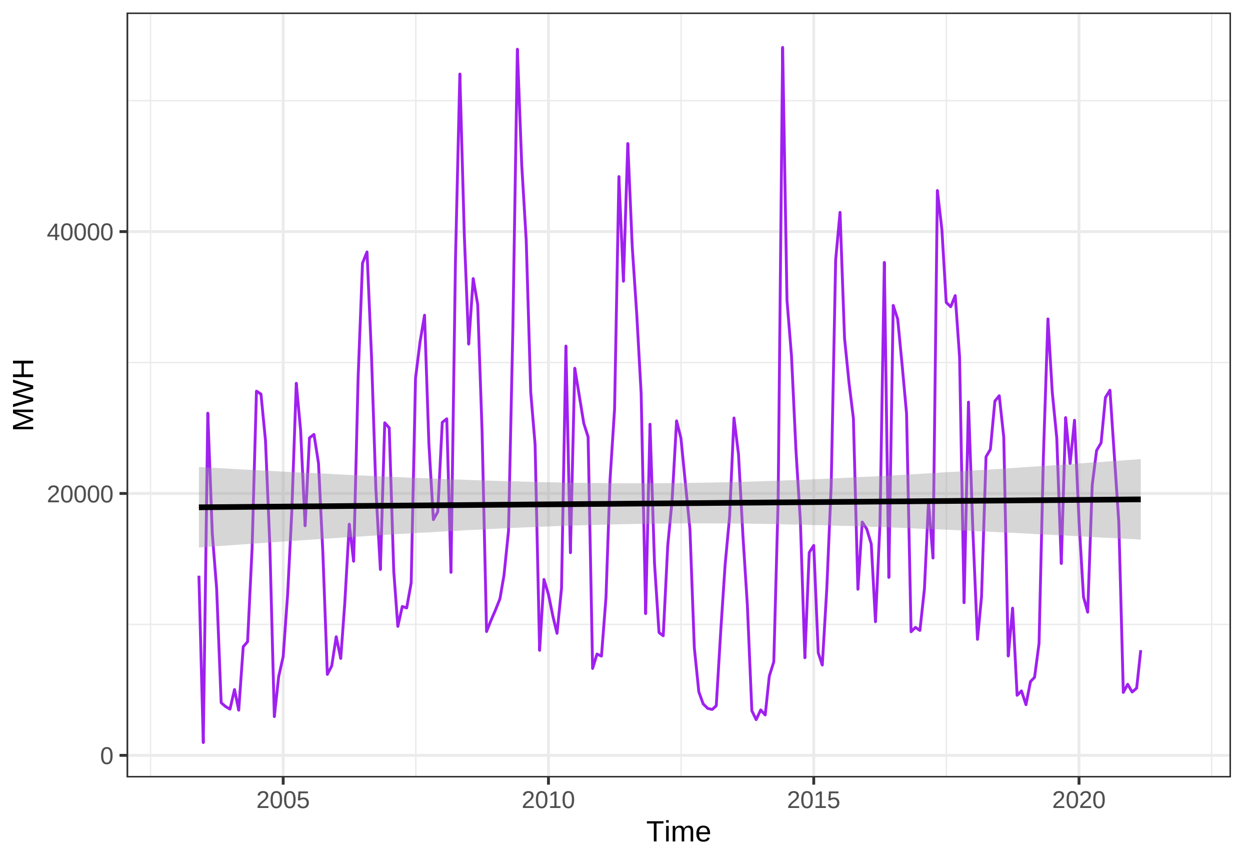
```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 4 rows containing non-finite values (stat_smooth).
```



```
#MWH  #using this one in ts analysis   #analyzing MWH variable over time
ts.mhw.plot <- ggplot(All_months.data, aes(x = Date, y = MWH)) +
  geom_line(color = "purple") +
  labs(x = "Time", y = expression("MWH")) +
  geom_smooth(method = lm, color = "black")
print(ts.mhw.plot)
```

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

## Warning: Removed 14 rows containing non-finite values (stat_smooth).

## Warning: Removed 14 row(s) containing missing values (geom_path).
```



>Can see seasonality trends for all three of the plotted of the variables

Cleaning Data

```
summary(All_months.data) #63 NAs are present, need to be removed
```

```
##       Date              Elevation.ft    Storage.af      Evaporation.af
##  Min.   :2003-01-01   Min.   :7430   Min.   :208892   Min.   : 106.0
##  1st Qu.:2007-09-23   1st Qu.:7466   1st Qu.:410750   1st Qu.: 226.5
##  Median :2012-06-16   Median :7487   Median :555214   Median : 556.5
##  Mean   :2012-06-16   Mean   :7483   Mean   :538830   Mean   : 658.5
##  3rd Qu.:2017-03-08   3rd Qu.:7498   3rd Qu.:642413   3rd Qu.:1039.2
##  Max.   :2021-12-01   Max.   :7519   Max.   :826302   Max.   :1544.0
##
##    Inflow.cfs      UnregInflow.cfs    Power.cfs        Bypass.cfs
##  Min.   : 258.0   Min.   : 195.0   Min.   : 142.0   Min.   :   0.00
##  1st Qu.: 432.5   1st Qu.: 420.8   1st Qu.: 571.5   1st Qu.:   0.00
##  Median : 631.0   Median : 586.5   Median :1023.0   Median :   0.00
##  Mean   :1168.1   Mean   :1169.7   Mean   :1095.7   Mean   :  59.24
##  3rd Qu.:1290.2   3rd Qu.:1213.0   3rd Qu.:1509.2   3rd Qu.:   0.00
##  Max.   :7456.0   Max.   :7915.0   Max.   :3242.0   Max.   :2379.00
##                                                     NA's   :12
##   Spillway.cfs          Total.cfs           MaxT            MinT
##  Min.   :-2.147e+09   Min.   : 142.0   Min.   :13.40   Min.   :-11.80
##  1st Qu.: 0.000e+00   1st Qu.: 580.2   1st Qu.:39.35   1st Qu.: 11.85
##  Median : 0.000e+00   Median :1038.5   Median :58.35   Median : 26.05
##  Mean   :-1.003e+07   Mean   :1159.3   Mean   :56.61   Mean   : 24.67
##  3rd Qu.: 0.000e+00   3rd Qu.:1535.8   3rd Qu.:76.00   3rd Qu.: 38.83
##  Max.   : 1.257e+03   Max.   :5939.0   Max.   :89.00   Max.   : 52.00
##  NA's   :14                            NA's   :4       NA's   :4
##      Precip             Snow             MWH
##  Min.   :0.0000   Min.   : 0.000   Min.   :  974
##  1st Qu.:0.2900   1st Qu.: 0.000   1st Qu.: 9596
##  Median :0.6000   Median : 0.900   Median :17596
##  Mean   :0.7258   Mean   : 3.685   Mean   :19247
##  3rd Qu.:0.9550   3rd Qu.: 5.400   3rd Qu.:26843
##  Max.   :4.3900   Max.   :32.900   Max.   :54068
##  NA's   :9        NA's   :21       NA's   :14
```

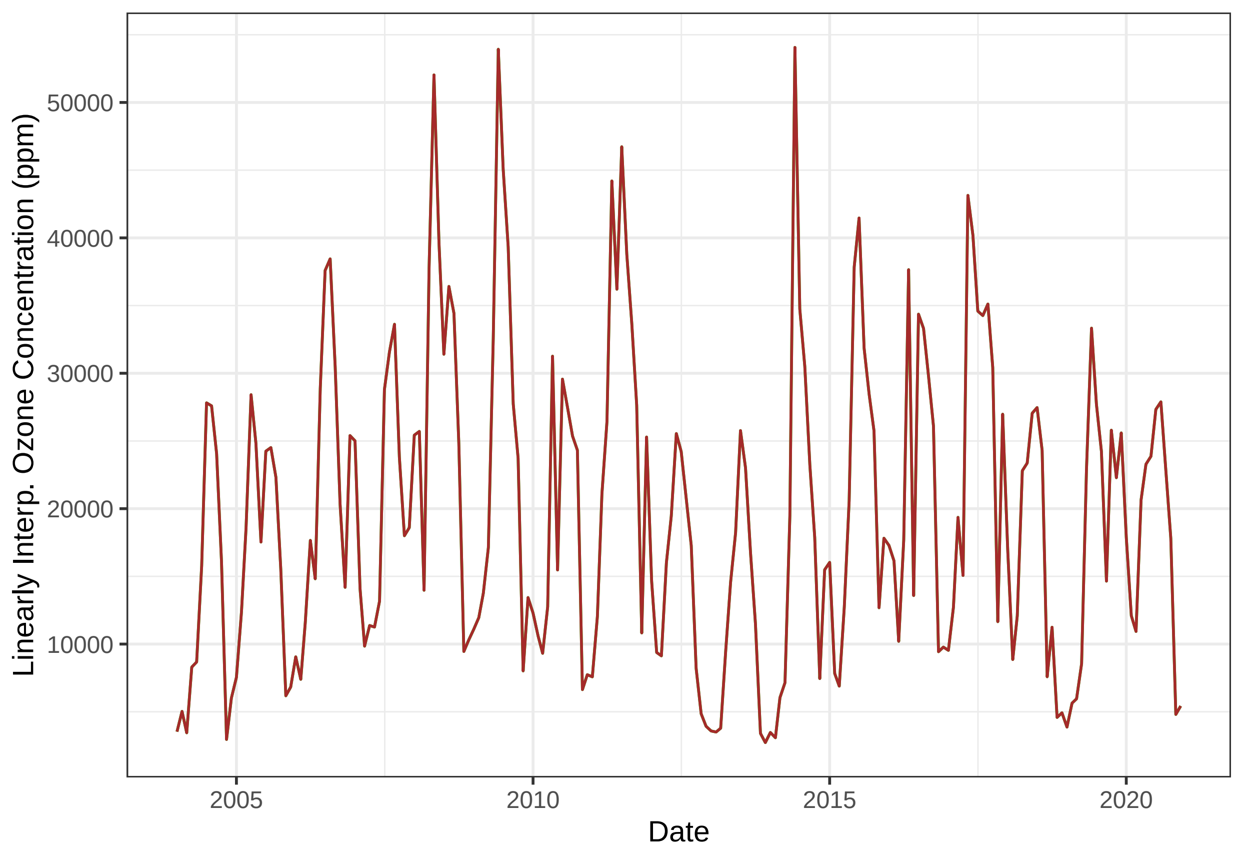
```
# To show I tried the conventional NA removal way we learned in class

#All_months.data_no_NA <- #removing NAs
#  All_months.data$MHW %>%  #referencing data frame to use
#  mutate( MWH_clean = #making new clean column for MWH
#          zoo::na.approx(MWH)) #cut NAs
#received an error with the mutate function
#going to try another way of removing 6 NAs in 2003 and 9 NAs in 2021


# Method that worked
class(All_months.data$Date) #checking class of data column
```

```
## [1] "Date"
```

```
#Remove 3 and 9 rows to get data from 2004-2020

new.04_20.data <- All_months.data[All_months.data$Date >= "2004-01-01" &
                         All_months.data$Date <= "2020-12-01", ]
                        #no NA rows !! Finally !!!
summary(new.04_20.data$MWH) #confirm no NAs
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2724   10275   17846   19719   27364   54068
```

```
#line plot of interpolated data to fill in any gaps, if any
ggplot(new.04_20.data) +
  geom_line(aes(x = Date, y = MWH), color = "green") +
  geom_line(aes(x = Date, y = MWH), color = "brown") +
  ylab("Linearly Interp. Ozone Concentration (ppm)")
```



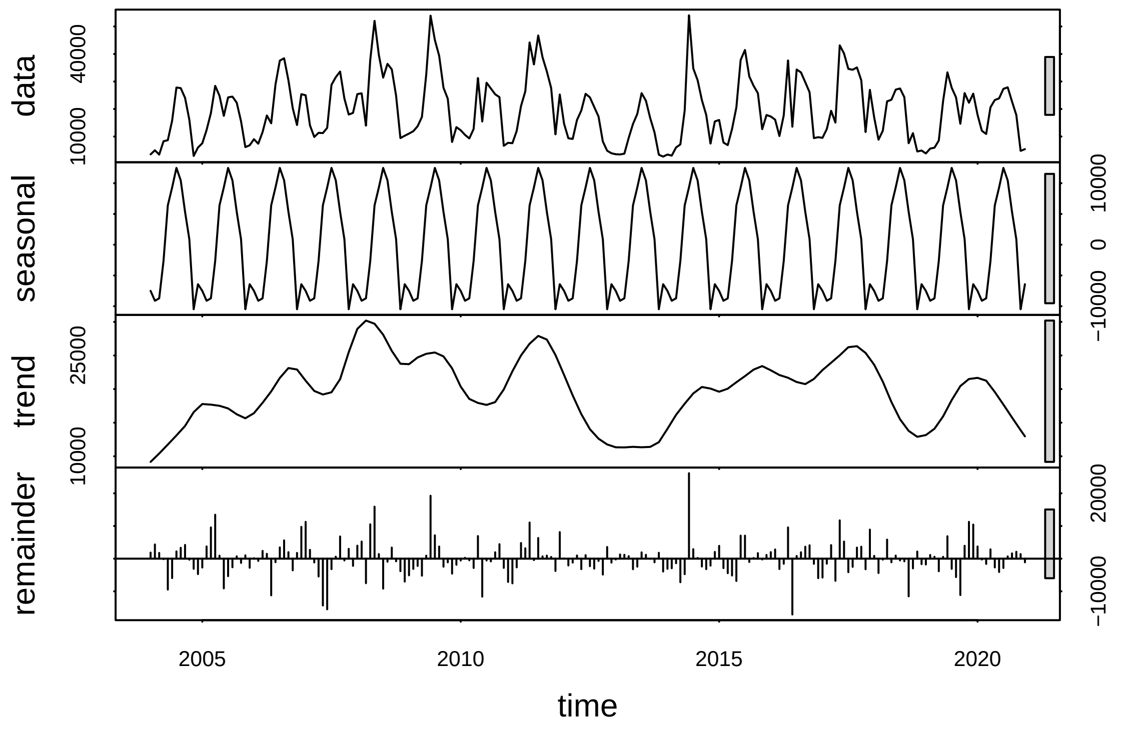
```
#nice try, no need to interpolate anything because data included were complete years between 2004-2020
```

Analysis

```
#make ts object to decompose later
f.month <- month(first(new.04_20.data$Date))
f.year <- year(first(new.04_20.data$Date))

monthly.ts <- ts(new.04_20.data$MWH, start = c(f.year, f.month), frequency=12)
print(monthly.ts)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2004  3528.00  5029.00  3447.00  8300.00  8675.00 15785.00 27816.00 27593.00
## 2005  7548.00 12367.00 18540.00 28418.00 24823.00 17534.00 24244.00 24513.00
## 2006  9060.00  7397.00 11619.00 17657.00 14821.00 28892.00 37592.00 38449.00
## 2007 25009.65 14081.17  9846.50 11371.82 11253.79 13172.69 28803.14 31572.30
## 2008 25428.43 25704.33 13974.08 37838.46 52036.40 39623.30 31403.03 36414.27
## 2009 11094.07 11954.59 13761.77 17134.41 32564.15 53930.05 45083.76 39331.79
## 2010 12291.97 10600.67  9316.80 12756.27 31261.97 15472.18 29568.71 27482.96
## 2011  7579.36 12046.15 21085.62 26438.65 44204.88 36206.31 46728.36 38809.96
## 2012 14654.21  9377.84  9129.81 16007.22 19461.11 25547.83 24217.11 20734.46
## 2013  3583.88  3503.77  3786.27  9504.44 14563.85 18232.63 25767.99 23011.43
## 2014  3469.89  3082.23  6043.70  7148.43 19469.59 54068.24 34752.07 30502.26
## 2015 16030.67  7828.28  6893.16 12747.38 20578.21 37837.30 41470.13 31853.25
## 2016 17286.93 16141.06 10201.73 17694.64 37649.79 13583.85 34366.58 33304.64
## 2017  9536.05 12723.50 19359.56 15065.10 43138.97 40176.89 34589.54 34259.15
## 2018 16970.44  8856.97 12095.95 22801.95 23367.01 27050.60 27467.62 24300.86
## 2019  3866.34  5636.76  5957.77  8547.85 22898.02 33333.26 27727.34 24235.30
## 2020 17898.71 12093.39 10932.17 20640.14 23280.85 23867.49 27330.00 27887.73
##           Sep      Oct      Nov      Dec
## 2004 24060.00 16100.00  2955.00  6012.00
## 2005 22315.00 15534.00  6180.00  6826.00
## 2006 30454.44 20335.14 14188.35 25399.23
## 2007 33619.48 23813.67 18001.93 18598.30
## 2008 34425.70 24805.86  9451.29 10299.39
## 2009 27742.05 23730.22  8017.16 13436.97
## 2010 25362.01 24316.74  6635.17  7738.21
## 2011 33649.91 27634.40 10821.95 25292.00
## 2012 17268.16  8208.16  4848.48  3929.45
## 2013 16701.77 11501.93  3406.28  2723.56
## 2014 23236.67 17874.38  7450.90 15486.70
## 2015 28443.65 25743.43 12681.00 17818.58
## 2016 29739.09 26100.47  9436.04  9776.47
## 2017 35114.75 30422.32 11654.99 26977.17
## 2018  7585.47 11244.60  4582.19  4925.78
## 2019 14648.04 25797.82 22285.26 25594.14
## 2020 22743.30 17812.57  4802.41  5427.88
```

```
#visualize the decomposed series
monthly.decomposed <- stl(monthly.ts, s.window = "periodic")
plot(monthly.decomposed) #results in 4 plots
```



```
#Run a monotonic trend analysis for the monthly series with the seasonal Mann-Kendall b/c that is the o
#run test
MWH.monthly_SMK1 <- Kendall::SeasonalMannKendall(monthly.ts)

#inspect results
MWH.monthly_SMK1
```

```
## tau = 0.0159, 2-sided pvalue =0.75719
```

```
   ## tau = 0.0159, 2-sided pvalue =0.75719
summary(MWH.monthly_SMK1)
```

```
## Score =  26 , Var(Score) = 7072
## denominator =  1632
## tau = 0.0159, 2-sided pvalue =0.75719
```

```
   ##  Score =  26 , Var(Score) = 7072, denominator =  1632
    #  tau = 0.0159, 2-sided pvalue =0.75719

#run 2nd test
MWH.monthly_SMK2 <- trend::smk.test(monthly.ts)

#inspect results
MWH.monthly_SMK2
```

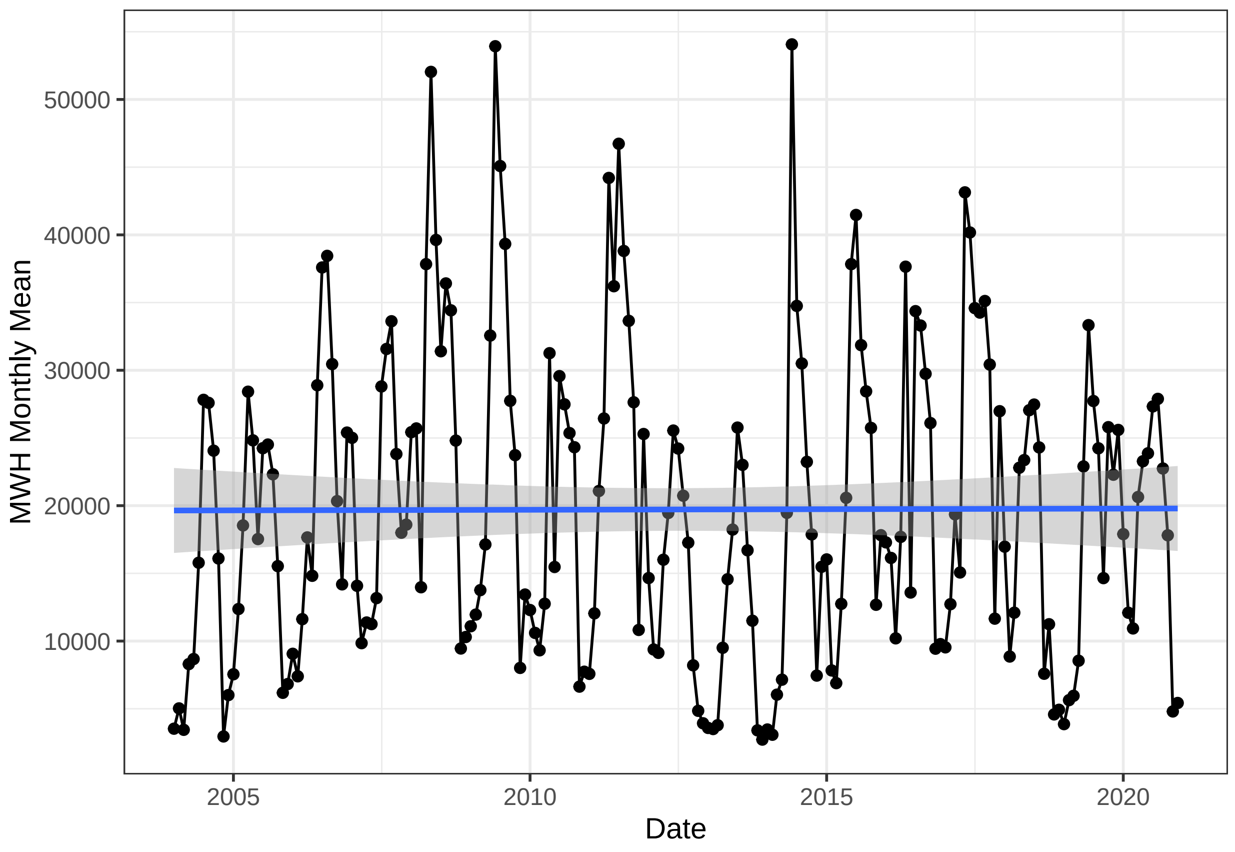
```
##
##  Seasonal Mann-Kendall trend test (Hirsch-Slack test)
##
## data:  monthly.ts
## z = 0.29728, p-value = 0.7663
## alternative hypothesis: true S is not equal to 0
## sample estimates:
##    S varS
##   26 7072
```

```
   ##Seasonal Mann-Kendall trend test (Hirsch-Slack test)
    # data:  monthly.ts
    # z = 0.29728, p-value = 0.7663
    # alternative hypothesis: true S is not equal to 0
    # sample estimates:
    #   S varS
    #  26 7072

summary(MWH.monthly_SMK2)
```

```
##
##  Seasonal Mann-Kendall trend test (Hirsch-Slack test)
##
## data: monthly.ts
## alternative hypothesis: two.sided
##
## Statistics for individual seasons
##
## H0
##                       S  varS    tau      z Pr(>|z|)
## Season 1:   S = 0   22 589.3  0.162  0.865  0.38701
## Season 2:   S = 0  -10 589.3 -0.074 -0.371  0.71084
## Season 3:   S = 0  -10 589.3 -0.074 -0.371  0.71084
## Season 4:   S = 0   -8 589.3 -0.059 -0.288  0.77308
## Season 5:   S = 0   24 589.3  0.176  0.947  0.34342
## Season 6:   S = 0   20 589.3  0.147  0.783  0.43383
## Season 7:   S = 0   -6 589.3 -0.044 -0.206  0.83682
## Season 8:   S = 0  -16 589.3 -0.118 -0.618  0.53665
## Season 9:   S = 0  -24 589.3 -0.176 -0.947  0.34342
## Season 10:   S = 0  26 589.3  0.191  1.030  0.30310
## Season 11:   S = 0   4 589.3  0.029  0.124  0.90165
## Season 12:   S = 0   4 589.3  0.029  0.124  0.90165
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
   ##Seasonal Mann-Kendall trend test (Hirsch-Slack test)
    # data: monthly.ts
    # alternative hypothesis: two.sided
    # Statistics for individual seasons
    # H0

#Plot with seasonality
MWH.monthly_SMK2_plot <- #reference cleaned data b4 making the f_month dataset
ggplot(new.04_20.data, aes(x = Date, y = MWH)) +
  geom_point() +
  geom_line() +
  ylab("MWH Monthly Mean") +
  geom_smooth( method = lm )
print(MWH.monthly_SMK2_plot)
```

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



>Interpretation of test results with seasonal component: Decomposing the data confirmed the presence of a seasonal trend in the data. Then the Seasonal Mann Kendall test was used to test stationarity for monotonic trends. For each season of the year represented by tau, we had a value of 0.0159, and a p-vlaue larger than 0.05 meaning we have evidence to support the null hypothesis. There is not a significant relationship bewteen Average Monthly MWH and Date because the p-value is greater than 0.05. There is not a significant trend decreasing or increasing over time as seen with the horz blue trend line in the graph.

> The SMK.Test was also used, and there are not statistical levels of pronounced results of tau for each season of the year represented by S in the first column.
>
> To answer the research question, no the electricity generation (MWH) has not changed over time 2004-2020 at the Blue Mesa Dam in Colorado.

Subtract Seasonal Component

Subtract the seasonal component from the `monthly.ts`.

Run the Mann Kendall test on the non-seasonal MWH monthly series. Compare the results with the ones obtained with the Seasonal Mann Kendall on the complete series.

```
#Separate the components and turn them into data frame
MWH.monthly_Components <- as.data.frame(monthly.decomposed$time.series[,1:3])
#Exclude seasonal column and make new data frame
MWH.monthly_NOseasonal <- MWH.monthly_Components %>%
#  select(trend, remainder) %>% #made df without seasonal column #ma
  mutate(meanMWH = trend + remainder) #new MWH variable to use later
MWH.monthly_NOseasonal$Date <- new.04_20.data$Date #added date column
```

```
#make new time series with no seasonality component, with no NA monthly data
#create second monthly set with old monthly df that has the date info
f.month2 <- month(first(MWH.monthly_NOseasonal$Date))
f.year2 <- year(first(MWH.monthly_NOseasonal$Date))
#make 2nd ts (no seasonality) for tests
MWH.monthly.ts2 <- ts(MWH.monthly_NOseasonal$meanMWH, start = c(f.year2, f.month2), frequency=12)
print(MWH.monthly.ts2)
```

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2004 11033.055 14140.290 12165.698 10929.911  2281.120  6471.717 15327.568
## 2005 15053.055 21478.290 27258.698 31047.911 18429.120  8220.717 11755.568
## 2006 16565.055 16508.290 20337.698 20286.911  8427.120 19578.717 25103.568
## 2007 32514.705 23192.460 18565.198 14001.731  4859.910  3859.407 16314.708
## 2008 32933.485 34815.620 22692.778 40468.371 45642.520 30310.017 18914.598
## 2009 18599.125 21065.880 22480.468 19764.321 26170.270 44616.767 32595.328
## 2010 19797.025 19711.960 18035.498 15386.181 24868.090  6158.897 17080.278
## 2011 15084.415 21157.440 29804.318 29068.561 37811.000 26893.027 34239.928
## 2012 22159.265 18489.130 17848.508 18637.131 13067.230 16234.547 11728.678
## 2013 11088.935 12615.060 12504.968 12134.351  8169.970  8919.347 13279.558
## 2014 10974.945 12193.520 14762.398  9778.341 13075.710 44754.957 22263.638
## 2015 23535.725 16939.570 15611.858 15377.291 14184.330 28524.017 28981.698
## 2016 24791.985 25252.350 18920.428 20324.551 31255.910  4270.567 21878.148
## 2017 17041.105 21834.790 28078.258 17695.011 36745.090 30863.607 22101.108
## 2018 24475.495 17968.260 20814.648 25431.861 16973.130 17737.317 14979.188
## 2019 11371.395 14748.050 14676.468 11177.761 16504.140 24019.977 15238.908
## 2020 25403.765 21204.680 19650.868 23270.051 16886.970 14554.207 14841.568
##            Aug       Sep       Oct       Nov       Dec
## 2004 17134.241 18717.609 15219.464 13444.763 12434.563
## 2005 14054.241 16972.609 14653.464 16669.763 13248.563
## 2006 27990.241 25112.049 19454.604 24678.113 31821.793
## 2007 21113.541 28277.089 22933.134 28491.693 25020.863
## 2008 25955.511 29083.309 23925.324 19941.053 16721.953
## 2009 28873.031 22399.659 22849.684 18506.923 19859.533
## 2010 17024.201 20019.619 23436.204 17124.933 14160.773
## 2011 28351.201 28307.519 26753.864 21311.713 31714.563
## 2012 10275.701 11925.769  7327.624 15338.243 10352.013
## 2013 12552.671 11359.379 10621.394 13896.043  9146.123
## 2014 20043.501 17894.279 16993.844 17940.663 21909.263
## 2015 21394.491 23101.259 24862.894 23170.763 24241.143
## 2016 22845.881 24396.699 25219.934 19925.803 16199.033
## 2017 23800.391 29772.359 29541.784 22144.753 33399.733
## 2018 13842.101  2243.079 10364.064 15071.953 11348.343
## 2019 13776.541  9305.649 24917.284 32775.023 32016.703
## 2020 17428.971 17400.909 16932.034 15292.173 11850.443
```

```
is.ts(MWH.monthly.ts2)#check it is a ts and no longer a df
```

```
## [1] TRUE
```

```
                                # WOOHOO!!! I figured it out! <3
```

```
#Run the tests again

#run test, no need to run smk.test b/c no trend present
MWH_monthly_SMK1.2 <- Kendall::MannKendall(MWH.monthly.ts2)
```

```
#inspect results
MWH_monthly_SMK1.2
```

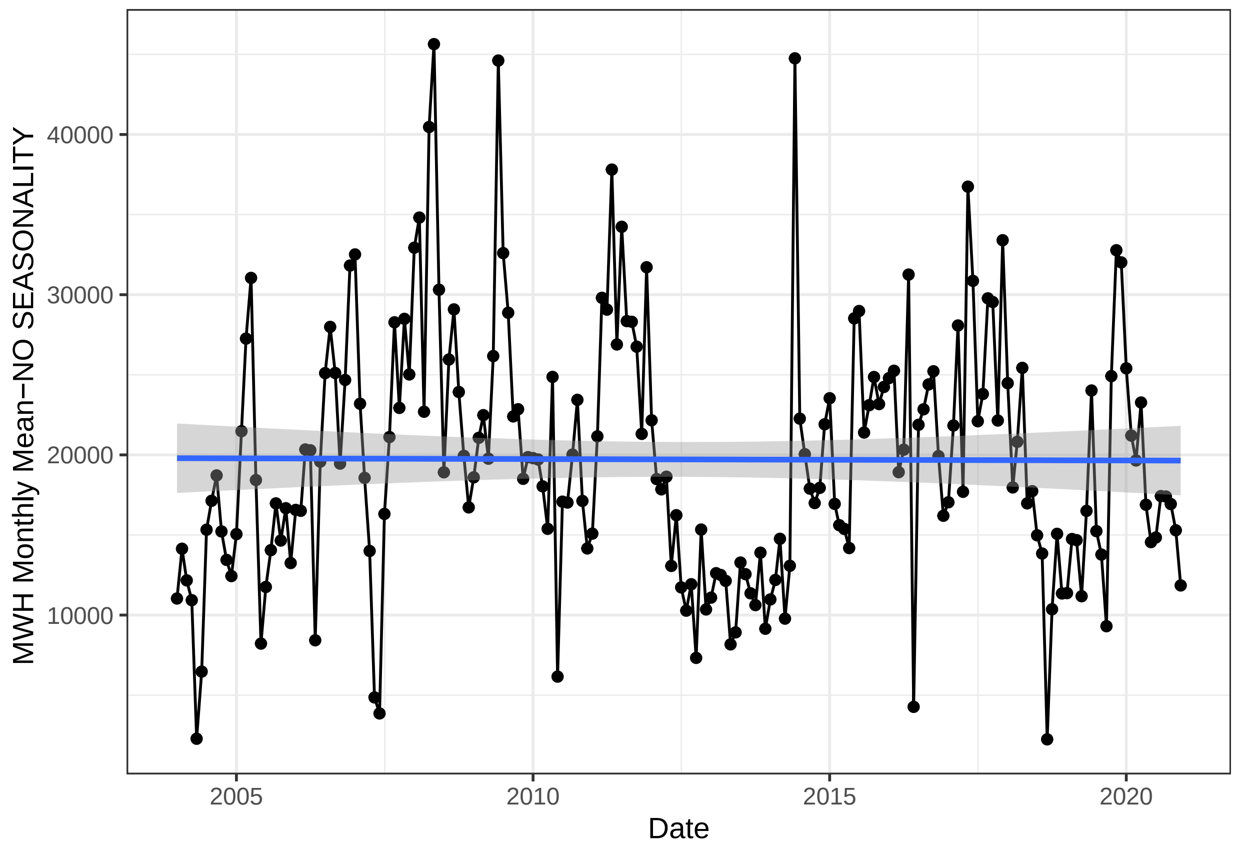
```
## tau = 0.00212, 2-sided pvalue =0.96481
```

```
   ## tau = 0.00212, 2-sided pvalue =0.96481

summary(MWH_monthly_SMK1.2 )
```

```
## Score =  44 , Var(Score) = 950175.3
## denominator =  20706
## tau = 0.00212, 2-sided pvalue =0.96481
```

```
   ## Score =  44 , Var(Score) = 950175.3, denominator =  20706
    #tau = 0.00212, 2-sided pvalue =0.96481

#plot 2 with no seasonal component
MWH.monthly_SMK2.2_plot <- #reference cleaned data before making the f_month dataset
ggplot(MWH.monthly_NOseasonal, aes(x = Date, y = meanMWH)) +
  geom_point() +
  geom_line() +
  ylab("MWH Monthly Mean-NO SEASONALITY ") +
  geom_smooth( method = lm )
print(MWH.monthly_SMK2.2_plot)
```

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



> Answer: After removing the seasonal component, and running the Mann Kendall test, tau had a value of

0.00212, and the p-value again was more than 0.05 giving support for the null hypothesis, and providing support for the alternative hypothesis. This means even without the seasonal component there still is not a significant trend decreasing or increasing over time in electricity generation (MHW).

Spatial Representation of Blue Mesa Reservoir

```
#install.packages('leaflet')
library(leaflet)
```

```
## Warning: package 'leaflet' was built under R version 4.1.3
```

```
#install.packages('mapview')
library(mapview)
```

```
## Warning: package 'mapview' was built under R version 4.1.3
```

```
#Disable on-the-fly projections
sf::sf_use_s2(FALSE)
```

```
## Spherical geometry (s2) switched off
```

Creating a spatial dataframe from known coordinates