ENV 797 - Time Series Analysis for Energy and Environment Applications | Spring 2025

Assignment 4 - Ayoung Kim

Ayoung Kim

Directions

You should open the .rmd file corresponding to this assignment on RStudio. The file is available on our class repository on Github. And to do so you will need to fork our repository and link it to your RStudio.

Once you have the file open on your local machine the first thing you will do is rename the file such that it includes your first and last name (e.g., "LuanaLima_TSA_A04_Sp25.Rmd"). Then change "Student Name" on line 4 with your name.

Then you will start working through the assignment by **creating code and output** that answer each question. Be sure to use this assignment document. Your report should contain the answer to each question and any plots/tables you obtained (when applicable).

When you have completed the assignment, **Knit** the text and code into a single PDF file. Submit this pdf using Sakai.

R packages needed for this assignment: "xlsx" or "readxl", "ggplot2", "forecast","tseries", and "Kendall". Install these packages, if you haven't done yet. Do not forget to load them before running your script, since they are NOT default packages.\

```
#Load/install required package here

library(readxl)
library(ggplot2)
library(forecast)
library(tseries)
library(Kendall)
library(openxlsx)
library(dplyr)
```

Questions

Consider the same data you used for A3 from the spreadsheet "Table_10.1_Renewable_Energy_Production_and_Consumpti
The data comes from the US Energy Information and Administration and corresponds to the January 2021 Monthly Energy Review. **For this assignment you will work only with the column "Total Renewable Energy Production"**.

```
getwd()
```

```
## [1] "/Users/ayoungkim/TSA_Sp25/Assignments"
```

```r
setwd("/Users/ayoungkim/TSA_Sp25/Data")

#Importing data set
library(readxl)
library(openxlsx)

renewable_data2 <- read.xlsx(xlsxFile="./Table_10.1_Renewable_Energy_Production_and_Consumption_by_Sourc

read_col_names2  <- read.xlsx(xlsxFile="./Table_10.1_Renewable_Energy_Production_and_Consumption_by_Sour

#Assign the column names to the data set
colnames(renewable_data2) <- read_col_names2

#Using "select" function, selected only Total Renewable Energy Production
renewable_data2_filtered <- select(renewable_data2,`Total Renewable Energy Production`)

df_renewable_data2_filtered<-as.data.frame(renewable_data2_filtered)

head(df_renewable_data2_filtered)
```

```
##   Total Renewable Energy Production
## 1                           219.839
## 2                           197.330
## 3                           218.686
## 4                           209.330
## 5                           215.982
## 6                           208.249
```

```r
ts1_renewable_data2_filtered<-ts(df_renewable_data2_filtered$`Total Renewable Energy Production`, start

ts1_renewable_data2_filtered
```

```
##         Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep
## 1   219.839 197.330 218.686 209.330 215.982 208.249 207.800 203.432 185.300
## 2   231.010 210.188 226.384 223.218 227.793 218.976 221.909 214.197 200.900
## 3   214.319 198.008 224.384 215.679 223.695 217.798 216.202 206.312 194.934
## 4   236.073 221.374 237.807 224.756 234.082 229.595 235.984 228.336 211.665
## 5   228.907 194.523 225.781 216.602 221.823 211.752 215.097 214.871 208.974
## 6   260.677 233.933 258.863 255.285 272.691 254.703 258.056 250.652 241.494
## 7   270.000 239.377 273.485 265.526 283.727 264.118 262.394 257.423 243.468
## 8   298.221 271.194 294.931 293.043 310.682 299.633 295.537 281.831 268.204
## 9   299.483 273.604 293.454 286.764 305.297 305.860 308.821 296.678 276.720
## 10  320.311 297.475 330.131 316.183 323.939 316.816 321.854 310.059 289.054
## 11  348.969 320.213 352.422 343.331 355.330 346.012 345.359 338.025 315.758
## 12  355.607 333.238 358.566 348.756 363.212 344.623 348.366 340.669 317.887
## 13  353.933 323.067 344.083 334.259 349.644 332.457 332.393 328.026 315.367
## 14  326.552 307.952 349.995 338.487 345.587 334.442 335.334 325.501 316.539
## 15  334.890 296.606 327.541 315.231 330.797 311.957 317.495 311.395 302.090
## 16  334.583 307.533 326.015 316.232 331.539 315.603 317.391 315.766 306.500
## 17  348.321 317.572 358.115 346.511 350.304 349.753 351.720 358.320 341.553
## 18  329.327 321.465 353.956 334.136 317.791 289.276 315.872 332.580 311.965
## 19  370.278 292.511 317.683 293.309 320.120 313.437 309.257 340.813 345.122
## 20  366.577 305.537 311.299 292.073 282.361 323.546 333.005 347.510 324.027
```

```
## 21 373.255 322.185 359.855 330.605 313.546 304.450 309.916 346.577 324.882
## 22 388.854 323.751 354.509 332.955 303.865 313.708 366.741 333.540 307.933
## 23 336.872 299.810 346.752 361.046 333.643 342.092 400.977 399.583 349.815
## 24 385.971 343.243 385.026 325.915 356.221 375.816 395.278 398.870 347.920
## 25 397.124 342.279 381.623 374.093 398.347 362.325 382.540 370.673 343.197
## 26 386.269 323.378 360.492 348.763 374.487 309.019 358.537 354.150 332.989
## 27 383.582 328.183 334.062 355.198 401.370 353.158 379.433 360.215 328.356
## 28 319.978 334.369 366.040 364.110 361.267 326.724 351.077 343.214 312.937
## 29 303.197 272.585 301.844 288.028 290.338 298.272 297.654 304.239 279.069
## 30 314.861 279.136 302.856 309.709 331.378 326.674 337.792 311.593 302.858
## 31 318.956 291.767 330.201 327.749 345.099 341.209 342.647 333.101 308.470
## 32 347.154 321.055 342.168 334.068 344.066 346.968 353.034 344.004 328.252
## 33 361.269 333.479 354.763 342.863 367.186 362.264 372.396 356.107 331.447
## 34 388.583 348.049 368.883 367.940 386.890 383.011 381.340 370.019 345.317
## 35 399.004 343.865 390.167 383.102 398.044 382.096 393.450 386.428 360.587
## 36 427.860 388.671 424.851 421.184 449.522 444.695 446.062 431.761 398.411
## 37 431.011 386.812 432.104 431.059 456.231 455.356 455.962 449.335 421.927
## 38 489.844 449.090 499.560 482.552 507.544 517.750 508.593 497.073 476.105
## 39 530.909 490.715 553.169 538.506 554.011 553.885 549.374 534.036 500.175
## 40 539.030 494.125 541.241 519.625 546.973 528.767 520.173 513.269 475.611
## 41 542.692 487.697 541.012 551.448 578.378 563.561 572.289 542.610 514.219
## 42 574.074 507.104 589.448 582.906 589.532 590.551 588.452 559.856 530.545
## 43 580.459 532.998 579.274 569.372 578.595 564.148 583.940 572.235 539.599
## 44 599.152 581.670 626.078 589.659 609.017 593.636 604.272 591.306 562.170
## 45 627.073 580.264 663.855 635.068 661.222 642.277 625.487 612.088 583.803
## 46 652.294 609.263 668.458 656.425 680.571 668.645 647.806 651.821 600.580
## 47 644.675 593.023 656.855 665.815 689.814 661.001 666.840 647.495 612.975
## 48 648.257 632.086 641.509 560.555 618.177 637.050 632.878 618.503 583.472
## 49 636.532 552.157 677.204 650.405 688.670 656.020 650.413 648.043 619.939
## 50 696.686 651.094 732.321 711.645 742.103 724.756 712.392 671.642 631.913
## 51 696.038 659.518 735.318 708.522 740.890 698.192 715.729 713.484 672.812
## 52 684.313 698.914 771.513 750.907 762.088 757.944 746.007 751.485 695.378
##        Oct     Nov     Dec
## 1  193.514 195.326 220.755
## 2  200.312 200.068 211.046
## 3  206.489 208.436 217.911
## 4  218.818 209.968 216.239
## 5  216.727 222.663 235.754
## 6  241.095 237.214 250.285
## 7  253.559 255.317 262.637
## 8  273.058 270.913 288.131
## 9  284.684 280.364 304.193
## 10 296.056 300.864 323.054
## 11 320.524 325.785 357.437
## 12 326.373 323.172 343.652
## 13 327.776 330.222 346.947
## 14 325.125 323.172 341.787
## 15 309.095 297.439 319.908
## 16 310.737 313.792 326.992
## 17 356.682 359.731 367.555
## 18 312.873 301.883 341.584
## 19 324.454 318.757 355.690
## 20 340.565 345.048 360.200
## 21 331.480 338.485 352.074
```

```
## 22 343.569 338.304 348.732
## 23 384.663 366.200 373.129
## 24 400.155 387.043 378.537
## 25 402.188 355.868 355.807
## 26 345.379 309.809 370.867
## 27 308.985 337.650 332.407
## 28 341.025 339.223 333.069
## 29 292.015 283.668 302.843
## 30 315.739 309.716 328.629
## 31 313.818 314.096 347.074
## 32 332.739 332.106 367.856
## 33 339.018 338.541 360.826
## 34 353.690 359.164 376.761
## 35 374.075 373.327 397.970
## 36 412.573 409.976 428.996
## 37 450.940 456.527 481.882
## 38 489.125 500.488 524.855
## 39 517.691 528.710 552.823
## 40 491.520 489.081 527.555
## 41 543.689 548.475 574.712
## 42 557.212 569.440 593.582
## 43 556.624 575.262 607.029
## 44 584.344 586.159 650.886
## 45 614.591 613.732 635.064
## 46 627.834 623.070 647.358
## 47 633.410 620.528 650.319
## 48 611.896 629.909 640.842
## 49 649.287 662.792 705.767
## 50 658.345 684.997 679.561
## 51 693.952 682.056 720.952
## 52
```

Stochastic Trend and Stationarity Tests

For this part you will work only with the column Total Renewable Energy Production.

Q1

Difference the "Total Renewable Energy Production" series using function diff(). Function diff() is from package base and take three main arguments: * *x* vector containing values to be differenced; * *lag* integer indicating with lag to use; * *differences* integer indicating how many times series should be differenced.

Try differencing at lag 1 only once, i.e., make `lag=1` and `differences=1`. Plot the differenced series. Do the series still seem to have trend?

Answer: Although there is difference between each time period, it seems like it still has some seasonality. No strong trend was found.

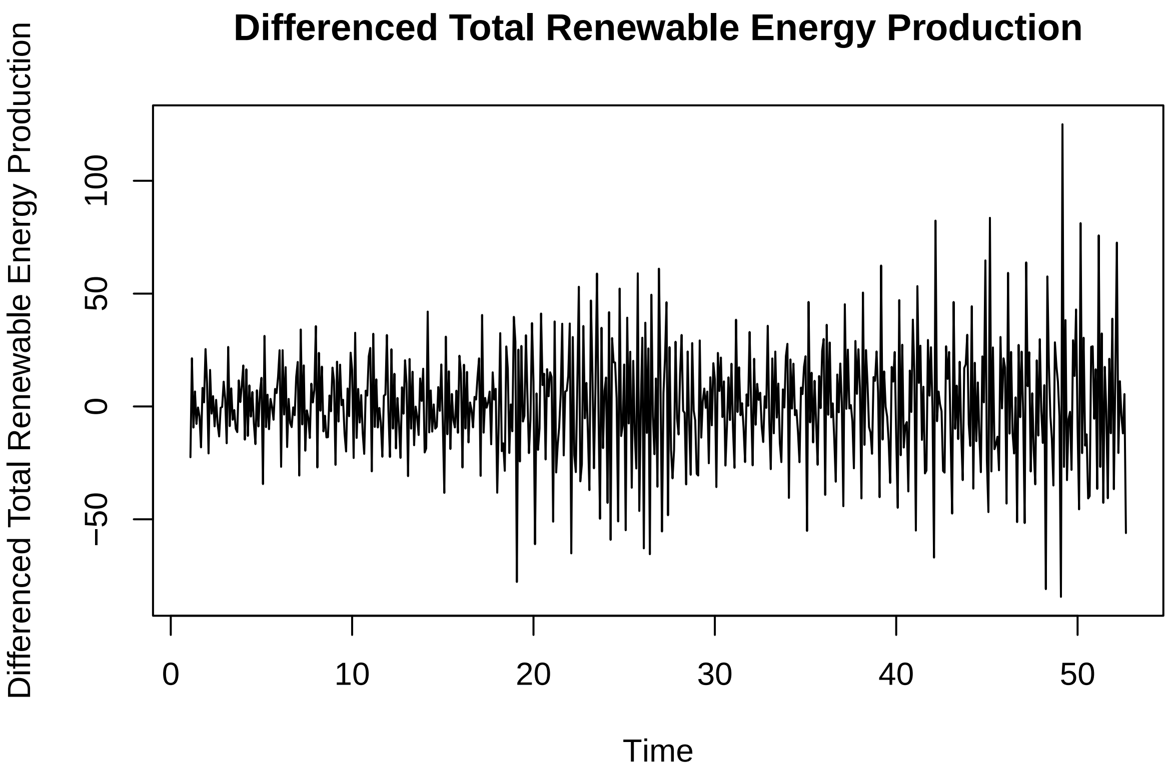
```r
renewable_data2_filtered_diff <-diff(ts1_renewable_data2_filtered,lag = 1,differences = 1)
renewable_data2_filtered_diff
```

```
##        Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep
## 1          -22.509  21.356  -9.356   6.652  -7.733  -0.449  -4.368 -18.132
```

```
## 2   10.255 -20.822  16.196  -3.166   4.575  -8.817   2.933  -7.712 -13.297
## 3    3.273 -16.311  26.376  -8.705   8.016  -5.897  -1.596  -9.890 -11.378
## 4   18.162 -14.699  16.433 -13.051   9.326  -4.487   6.389  -7.648 -16.671
## 5   12.668 -34.384  31.258  -9.179   5.221 -10.071   3.345  -0.226  -5.897
## 6   24.923 -26.744  24.930  -3.578  17.406 -17.988   3.353  -7.404  -9.158
## 7   19.715 -30.623  34.108  -7.959  18.201 -19.609  -1.724  -4.971 -13.955
## 8   35.584 -27.027  23.737  -1.888  17.639 -11.049  -4.096 -13.706 -13.627
## 9   11.352 -25.879  19.850  -6.690  18.533   0.563   2.961 -12.143 -19.958
## 10  16.118 -22.836  32.656 -13.948   7.756  -7.123   5.038 -11.795 -21.005
## 11  25.915 -28.756  32.209  -9.091  11.999  -9.318  -0.653  -7.334 -22.267
## 12  -1.830 -22.369  25.328  -9.810  14.456 -18.589   3.743  -7.697 -22.782
## 13  10.281 -30.866  21.016  -9.824  15.385 -17.187  -0.064  -4.367 -12.659
## 14 -20.395 -18.600  42.043 -11.508   7.100 -11.145   0.892  -9.833  -8.962
## 15  -6.897 -38.284  30.935 -12.310  15.566 -18.840   5.538  -6.100  -9.305
## 16  14.675 -27.050  18.482  -9.783  15.307 -15.936   1.788  -1.625  -9.266
## 17  21.329 -30.749  40.543 -11.604   3.793  -0.551   1.967   6.600 -16.767
## 18 -38.228  -7.862  32.491 -19.820 -16.345 -28.515  26.596  16.708 -20.615
## 19  28.694 -77.767  25.172 -24.374  26.811  -6.683  -4.180  31.556   4.309
## 20  10.887 -61.040   5.762 -19.226  -9.712  41.185   9.459  14.505 -23.483
## 21  13.055 -51.070  37.670 -29.250 -17.059  -9.096   5.466  36.661 -21.695
## 22  36.780 -65.103  30.758 -21.554 -29.090   9.843  53.033 -33.201 -25.607
## 23 -11.860 -37.062  46.942  14.294 -27.403   8.449  58.885  -1.394 -49.768
## 24  12.842 -42.728  41.783 -59.111  30.306  19.595  19.462   3.592 -50.950
## 25  18.587 -54.845  39.344  -7.530  24.254 -36.022  20.215 -11.867 -27.476
## 26  30.462 -62.891  37.114 -11.729  25.724 -65.468  49.518  -4.387 -21.161
## 27  12.715 -55.399   5.879  21.136  46.172 -48.212  26.275 -19.218 -31.859
## 28 -12.429  14.391  31.671  -1.930  -2.843 -34.543  24.353  -7.863 -30.277
## 29 -29.872 -30.612  29.259 -13.816   2.310   7.934  -0.618   6.585 -25.170
## 30  12.018 -35.725  23.720   6.853  21.669  -4.704  11.118 -26.199  -8.735
## 31  -9.673 -27.189  38.434  -2.452  17.350  -3.890   1.438  -9.546 -24.631
## 32   0.080 -26.099  21.113  -8.100   9.998   2.902   6.066  -9.030 -15.752
## 33  -6.587 -27.790  21.284 -11.900  24.323  -4.922  10.132 -16.289 -24.660
## 34  27.757 -40.534  20.834  -0.943  18.950  -3.879  -1.671 -11.321 -24.702
## 35  22.243 -55.139  46.302  -7.065  14.942 -15.948  11.354  -7.022 -25.841
## 36  29.890 -39.189  36.180  -3.667  28.338  -4.827   1.367 -14.301 -33.350
## 37   2.015 -44.199  45.292  -1.045  25.172  -0.875   0.606  -6.627 -27.408
## 38   7.962 -40.754  50.470 -17.008  24.992  10.206  -9.157 -11.520 -20.968
## 39   6.054 -40.194  62.454 -14.663  15.505  -0.126  -4.511 -15.338 -33.861
## 40 -13.793 -44.905  47.116 -21.616  27.348 -18.206  -8.594  -6.904 -37.658
## 41  15.137 -54.995  53.315  10.436  26.930 -14.817   8.728 -29.679 -28.391
## 42  -0.638 -66.970  82.344  -6.542   6.626   1.019  -2.099 -28.596 -29.311
## 43 -13.123 -47.461  46.276  -9.902   9.223 -14.447  19.792 -11.705 -32.636
## 44  -7.877 -17.482  44.408 -36.419  19.358 -15.381  10.636 -12.966 -29.136
## 45 -23.813 -46.809  83.591 -28.787  26.154 -18.945 -16.790 -13.399 -28.285
## 46  17.230 -43.031  59.195 -12.033  24.146 -11.926 -20.839   4.015 -51.241
## 47  -2.683 -51.652  63.832   8.960  23.999 -28.813   5.839 -19.345 -34.520
## 48  -2.062 -16.171   9.423 -80.954  57.622  18.873  -4.172 -14.375 -35.031
## 49  -4.310 -84.375 125.047 -26.799  38.265 -32.650  -5.607  -2.370 -28.104
## 50  -9.081 -45.592  81.227 -20.676  30.458 -17.347 -12.364 -40.750 -39.729
## 51  16.477 -36.520  75.800 -26.796  32.368 -42.698  17.537  -2.245 -40.672
## 52 -36.639  14.601  72.599 -20.606  11.181  -4.144 -11.937   5.478 -56.107
##        Oct     Nov     Dec
## 1    8.214   1.812  25.429
## 2   -0.588  -0.244  10.978
```

```
## 3   11.555   1.947   9.475
## 4    7.153  -8.850   6.271
## 5    7.753   5.936  13.091
## 6   -0.399  -3.881  13.071
## 7   10.091   1.758   7.320
## 8    4.854  -2.145  17.218
## 9    7.964  -4.320  23.829
## 10   7.002   4.808  22.190
## 11   4.766   5.261  31.652
## 12   8.486  -3.201  20.480
## 13  12.409   2.446  16.725
## 14   8.586  -1.953  18.615
## 15   7.005 -11.656  22.469
## 16   4.237   3.055  13.200
## 17  15.129   3.049   7.824
## 18   0.908 -10.990  39.701
## 19 -20.668  -5.697  36.933
## 20  16.538   4.483  15.152
## 21   6.598   7.005  13.589
## 22  35.636  -5.265  10.428
## 23  34.848 -18.463   6.929
## 24  52.235 -13.112  -8.506
## 25  58.991 -46.320  -0.061
## 26  12.390 -35.570  61.058
## 27 -19.371  28.665  -5.243
## 28  28.088  -1.802  -6.154
## 29  12.946  -8.347  19.175
## 30  12.881  -6.023  18.913
## 31   5.348   0.278  32.978
## 32   4.487  -0.633  35.750
## 33   7.571  -0.477  22.285
## 34   8.373   5.474  17.597
## 35  13.488  -0.748  24.643
## 36  14.162  -2.597  19.020
## 37  29.013   5.587  25.355
## 38  13.020  11.363  24.367
## 39  17.516  11.019  24.113
## 40  15.909  -2.439  38.474
## 41  29.470   4.786  26.237
## 42  26.667  12.228  24.142
## 43  17.025  18.638  31.767
## 44  22.174   1.815  64.727
## 45  30.788  -0.859  21.332
## 46  27.254  -4.764  24.288
## 47  20.435 -12.882  29.791
## 48  28.424  18.013  10.933
## 49  29.348  13.505  42.975
## 50  26.432  26.652  -5.436
## 51  21.140 -11.896  38.896
## 52
```

```r
plot(renewable_data2_filtered_diff, type="l", main = "Differenced Total Renewable Energy Production",
     ylab = "Differenced Total Renewable Energy Production", xlab = "Time")
```



Q2

Copy and paste part of your code for A3 where you run the regression for Total Renewable Energy Production and subtract that from the original series. This should be the code for Q3 and Q4. make sure you use the same name for you time series object that you had in A3, otherwise the code will not work.

```
nobs<-nrow(renewable_data2_filtered)
t<-1:nobs
ts1_lm<-lm(ts1_renewable_data2_filtered~t)

#Print the summary of the regression
print(ts1_lm)
```

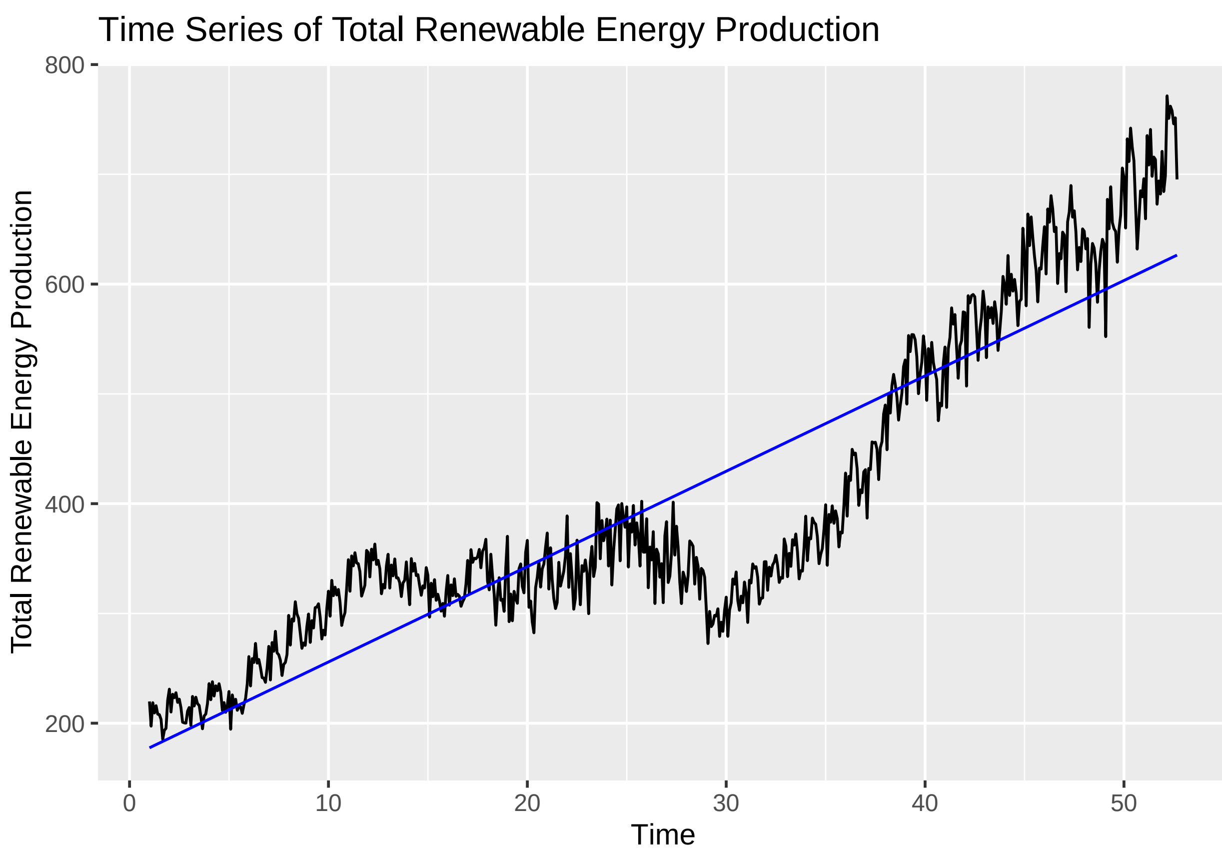
```
##
## Call:
## lm(formula = ts1_renewable_data2_filtered ~ t)
##
## Coefficients:
## (Intercept)            t
##    176.8729       0.7239
```

```
summary(ts1_lm)
```

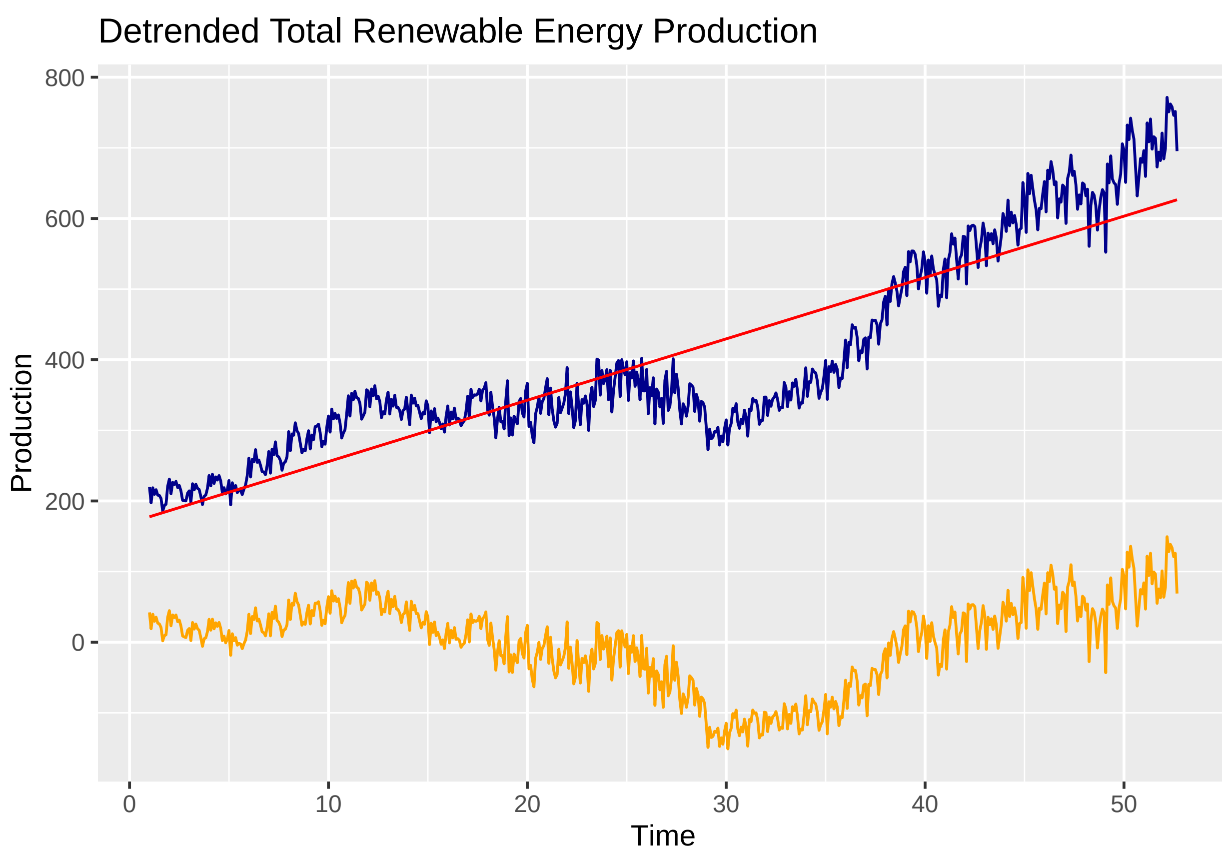
```
##
```

```
## Call:
## lm(formula = ts1_renewable_data2_filtered ~ t)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -151.11  -37.84   13.53   41.76  149.42
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 176.87293    4.96189   35.65   <2e-16 ***
## t             0.72393    0.01382   52.37   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 61.75 on 619 degrees of freedom
## Multiple R-squared:  0.8159, Adjusted R-squared:  0.8156
## F-statistic:  2743 on 1 and 619 DF,  p-value: < 2.2e-16
```

```r
#linear trend in the plot (ts1-Renewable Energy Production)
plot_ts1_lm<-autoplot(ts1_renewable_data2_filtered) +
  geom_line(aes(y = fitted(ts1_lm)), color = "blue") +
  labs(title = "Time Series of Total Renewable Energy Production",
       y = "Total Renewable Energy Production",
       x = "Time")

plot_ts1_lm
```



```r
#Detrend series
beta0_ts1 <- as.numeric(ts1_lm$coefficients[1])
beta1_ts1 <- as.numeric(ts1_lm$coefficients[2])

linear_trend_ts1 <- beta0_ts1 + beta1_ts1 * t

ts1_linear <- ts(linear_trend_ts1,start = c(1,1), frequency=12)

detrend_renewable_ts1 <- ts1_renewable_data2_filtered - linear_trend_ts1

ts1_detrend_renewable <- ts(detrend_renewable_ts1, start = c(1,1), frequency=12)

#Plot 1 - Detrended Total Renewable Energy Production
ts1_detrended_plot<-autoplot(ts1_renewable_data2_filtered, color = "darkblue") +
  autolayer(ts1_detrend_renewable, series = "Detrended", color = "orange") +
  autolayer(ts1_linear, series = "Linear Component", color = "red") +
  labs(title = "Detrended Total Renewable Energy Production", x = "Time", y = "Production")

ts1_detrended_plot
```



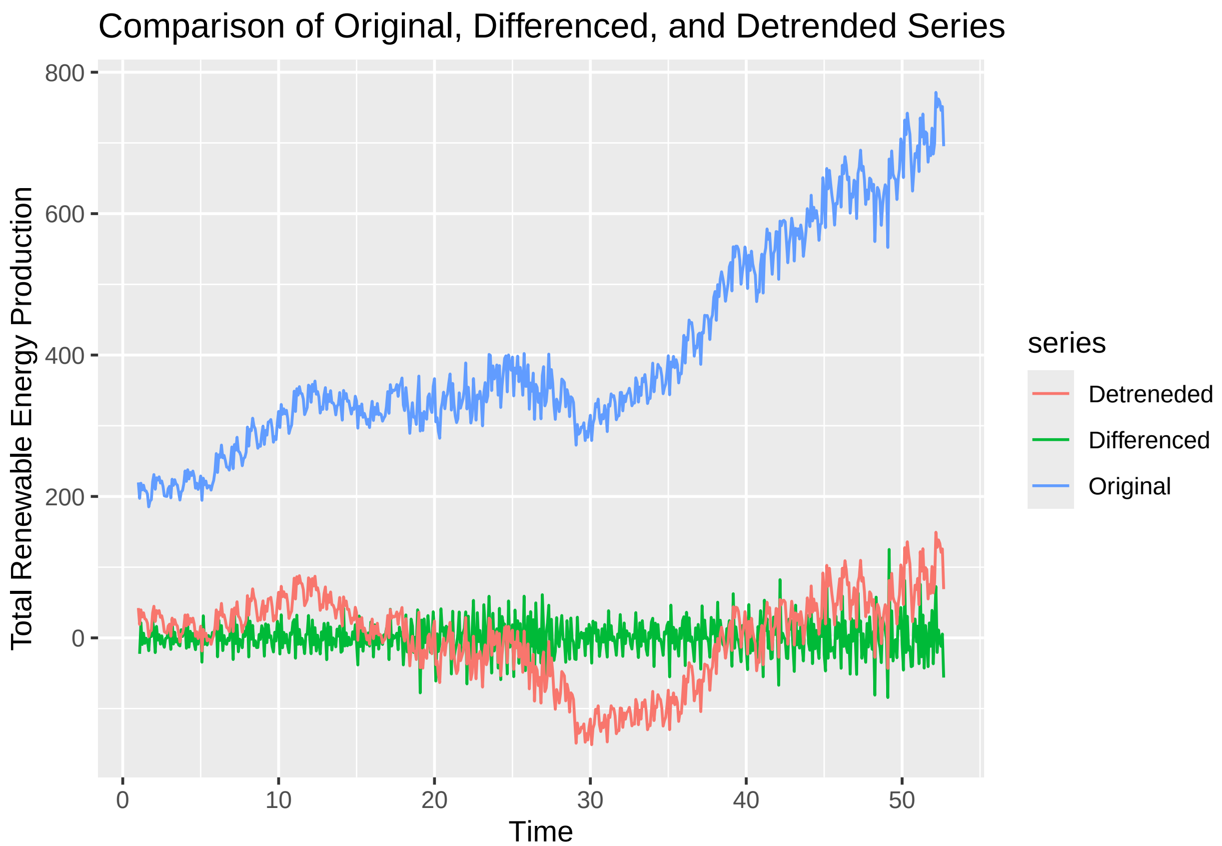
Q3

Now let's compare the differenced series with the detrended series you calculated on A3. In other words, for the "Total Renewable Energy Production" compare the differenced series from Q1 with the series you detrended in Q2 using linear regression.

Using autoplot() + autolayer() create a plot that shows the three series together. Make sure your plot has a legend. The easiest way to do it is by adding the `series=` argument to each autoplot and autolayer function. Look at the key for A03 for an example on how to use autoplot() and autolayer().

What can you tell from this plot? Which method seems to have been more efficient in removing the trend?

```
autoplot(ts1_renewable_data2_filtered,series = "Original")+
  autolayer(renewable_data2_filtered_diff, series = "Differenced")+
  autolayer(ts1_detrend_renewable,series = "Detreneded")+
  labs(y="Total Renewable Energy Production", title="Comparison of Original, Differenced, and Detrended
```

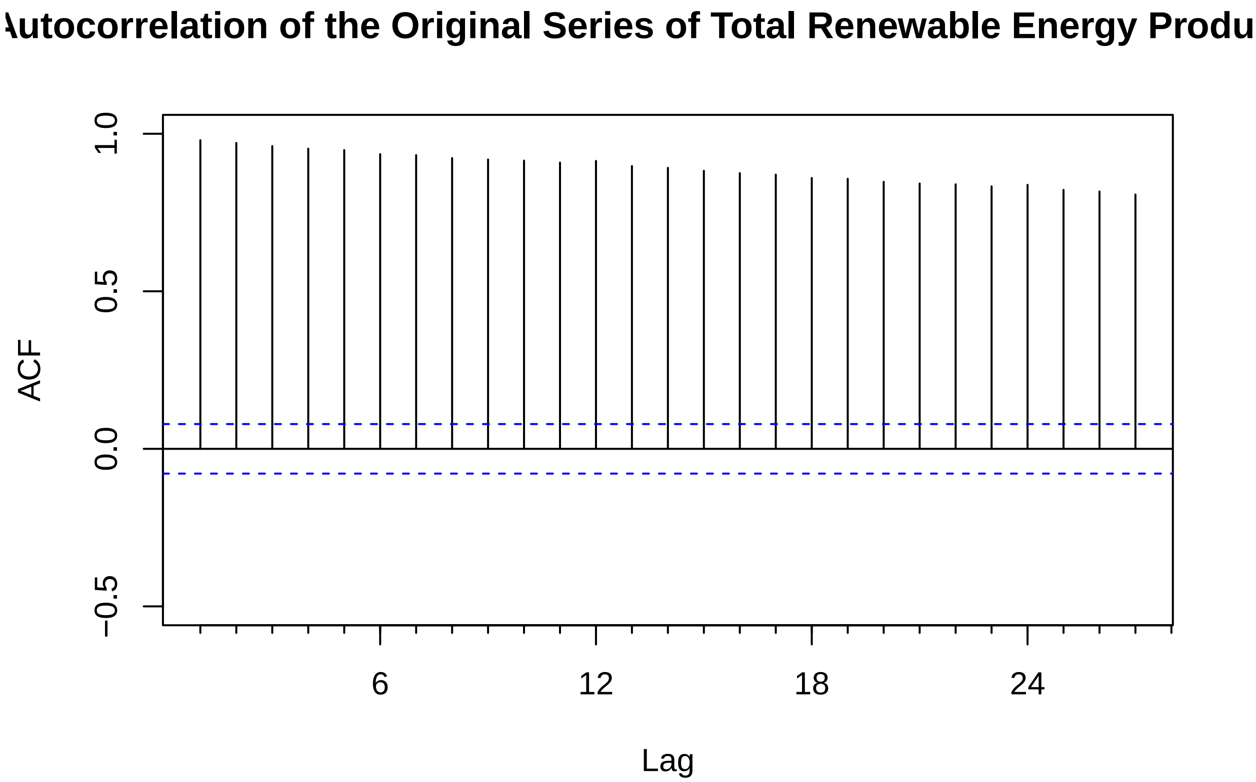


Answer: Differenced series seems to have more efficient in removing the trend. Compared to the original trend (blue), detrended one seems to have less trend, but differenced one (green) seems to have less trend among three. Instead, differenced one shows the seasonality.

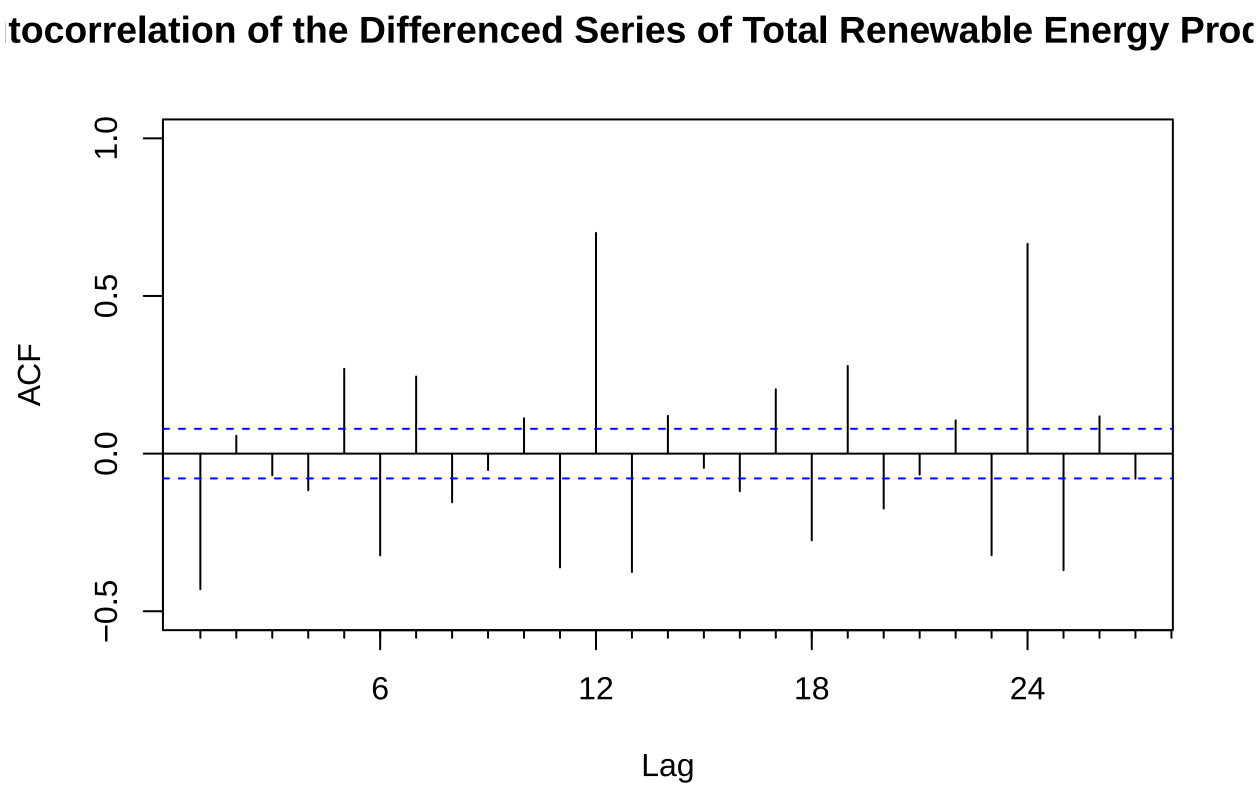
Q4

Plot the ACF for the three series and compare the plots. Add the argument `ylim=c(-0.5,1)` to the autoplot() or Acf() function - whichever you are using to generate the plots - to make sure all three y axis have the same limits. Looking at the ACF which method do you think was more efficient in eliminating the trend? The linear regression or differencing?

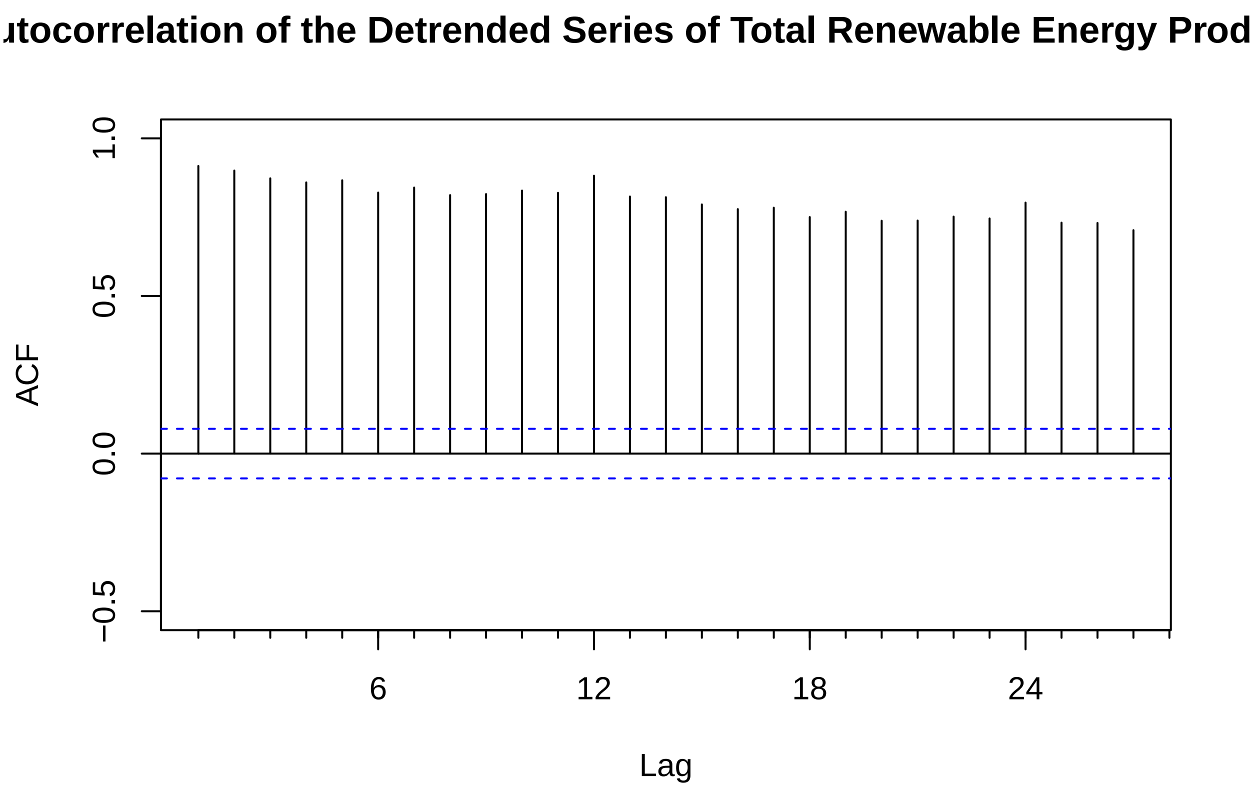
```
#ACF for the original series
acf_ts1_original <- Acf(ts1_renewable_data2_filtered,
                        main="Autocorrelation of the Original Series of Total Renewable Energy Producti
                        type="correlation", ylim=c(-0.5,1))
```




```
acf_ts1_differenced <-Acf(renewable_data2_filtered_diff,
                          main="Autocorrelation of the Differenced Series of Total Renewable Energy Produ
                          type="correlation", ylim=c(-0.5,1))
```




```
acf_ts1_detreneded <-Acf(ts1_detrend_renewable,
                         main="Autocorrelation of the Detrended Series of Total Renewable Energy Producti
                         type="correlation", ylim=c(-0.5,1))
```



Answer: Differenced series seems more efficiecnt in eliminating the trend according to the result of ACF as well. The original linear regression series and the detrended one still show the trend compared to the differenced one.

Q5

Compute the Seasonal Mann-Kendall and ADF Test for the original "Total Renewable Energy Production" series. Ask R to print the results. Interpret the results for both test. What is the conclusion from the Seasonal Mann Kendall test? What's the conclusion for the ADF test? Do they match what you observed in Q3 plot? Recall that having a unit root means the series has a stochastic trend. And when a series has stochastic trend we need to use differencing to remove the trend.

```r
#Seasonal Mann-Kandall
seasonal_Mann_Kendall_1 <-SeasonalMannKendall(ts1_renewable_data2_filtered)
summary(seasonal_Mann_Kendall_1)
```

```
## Score =  12468 , Var(Score) = 190008
## denominator =  15758.5
## tau = 0.791, 2-sided pvalue =< 2.22e-16
```

```r
#ADF test
adf_test_1 <-adf.test(ts1_renewable_data2_filtered)
print(adf_test_1)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts1_renewable_data2_filtered
## Dickey-Fuller = -1.0898, Lag order = 8, p-value = 0.9242
## alternative hypothesis: stationary
```

> Answer: According to the seasonal Mann-Kendall result, the -value is less than or equal to 2,22e-16, which means rejecting the null hypothesis. The result of Augmented Dickey-Fuller Test, Dicket-fuller = -1.0898. The p-value is bigger than 0.05, which means that it has stochastic trend and non-stationary. And the results match with what I've observed in Q3 plot.

Q6

Aggregate the original "Total Renewable Energy Production" series by year. You can use the same procedure we used in class. Store series in a matrix where rows represent months and columns represent years. And then take the columns mean using function colMeans(). Recall the goal is the remove the seasonal variation from the series to check for trend. Convert the accumulates yearly series into a time series object and plot the series using autoplot().

#I checked my code with AI.

```r
#Now let's try the yearly data
renewable_matrix <- matrix(ts1_renewable_data2_filtered, nrow=12, byrow=FALSE)
```

```
## Warning in matrix(ts1_renewable_data2_filtered, nrow = 12, byrow = FALSE): data
## length [621] is not a sub-multiple or multiple of the number of rows [12]
```

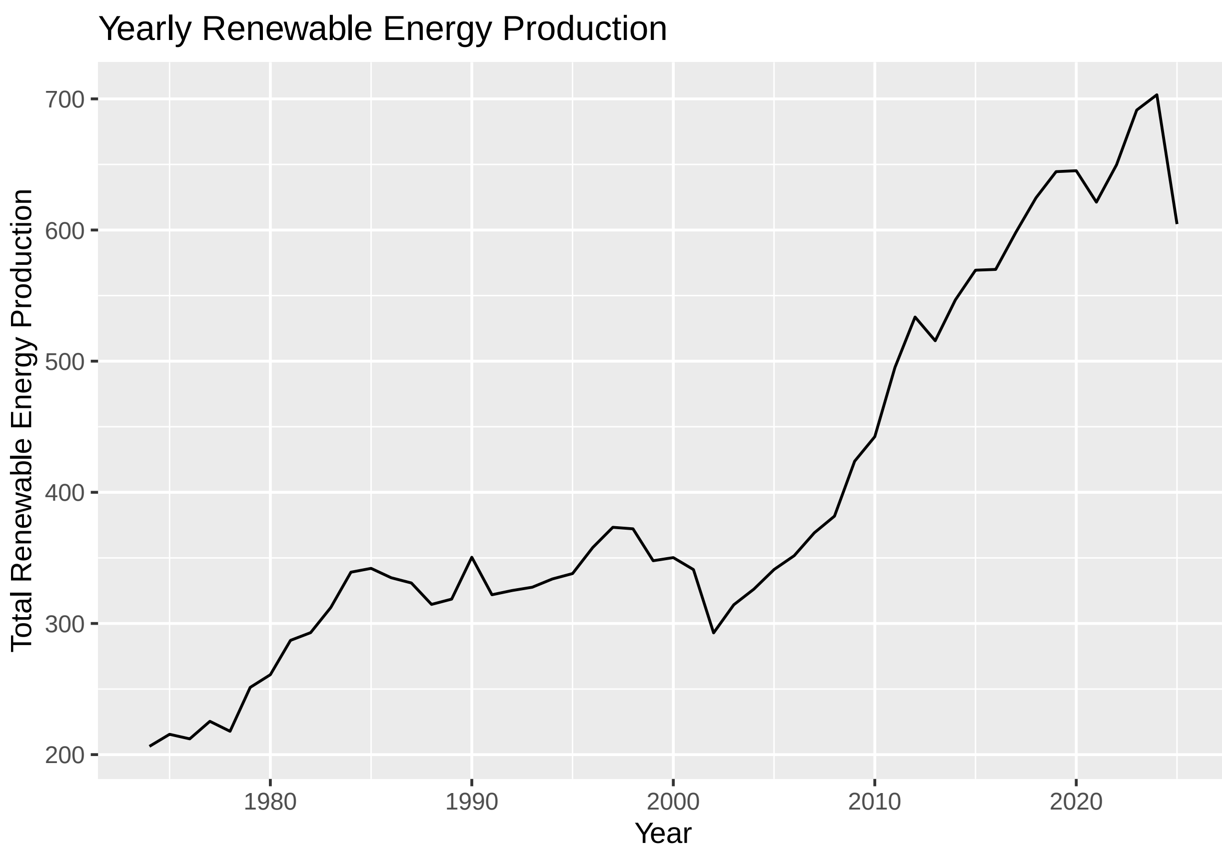
```r
renewable_yearly <- colMeans(renewable_matrix)

my_year <- c(1974:2025)

renewable_yearly <- data.frame(my_year,"renewable_data2_filtered"=renewable_yearly)

print("Results for ADF test on yearly data/n")
```

```
## [1] "Results for ADF test on yearly data/n"
```

```r
print(adf.test(renewable_yearly$renewable_data2_filtered,alternative = "stationary")) #stationary over
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  renewable_yearly$renewable_data2_filtered
## Dickey-Fuller = -1.6634, Lag order = 3, p-value = 0.7098
## alternative hypothesis: stationary
```

```r
#Convert yearly series into a time series object
ts_renewable_yearly <-ts(renewable_yearly$renewable_data2_filtered, start = c(1974), frequency = 1)

#Plot the series
autoplot(ts_renewable_yearly)+
  labs(title="Yearly Renewable Energy Production", x="Year", y="Total Renewable Energy Production")
```



Q7

Apply the Mann Kendall, Spearman correlation rank test and ADF. Are the results from the test in agreement with the test results for the monthly series, i.e., results for Q6?

```
# Seasonal Mann-Kendall Test
seasonal_Mann_Kendall_2 <- SeasonalMannKendall(ts_renewable_yearly)
summary(seasonal_Mann_Kendall_2)
```

```
## Score =  1070 , Var(Score) = 16059.33
## denominator =  1326
## tau = 0.807, 2-sided pvalue =< 2.22e-16
```

```
#Spearman correlation rant test
spearman_1<-cor.test(ts_renewable_yearly,my_year, method="spearman")
print(spearman_1)
```

```
##
##  Spearman's rank correlation rho
##
## data:  ts_renewable_yearly and my_year
## S = 1908, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
```

```
##      rho
## 0.918552
```

```r
#ADF test
adf_test_2 <-adf.test(ts_renewable_yearly,alternative="stationary")
print(adf_test_2)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_renewable_yearly
## Dickey-Fuller = -1.6634, Lag order = 3, p-value = 0.7098
## alternative hypothesis: stationary
```

> Answer: Comparing the result in Q6, the results in A7 have same test results. For example, the p-value of monthly series (ADF test result) is same as the one in Q7 with 0.7098.