ENV 790.30 - Time Series Analysis for Energy Data | Spring 2023

Assignment 8 - Due date 03/27/23

Qiuying Liao

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 project open the first thing you will do is change "Student Name" on line 3 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. Rename the pdf file such that it includes your first and last name (e.g., "LuanaLima_TSA_A08_Sp22.Rmd"). Submit this pdf using Sakai.

Set up

Some packages needed for this assignment: `forecast`,`tseries`,`smooth`. Do not forget to load them before running your script, since they are NOT default packages.

```
#Load/install required package here
library(forecast)
```

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

```
library(tseries)
library(smooth)
```

```
## Loading required package: greybox

## Package "greybox", v1.0.7 loaded.

## This is package "smooth", v3.2.0
```

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'

## The following object is masked from 'package:greybox':
##
##     spread
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:greybox':
##
##     hm

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

```
library(dplyr)
```

```
##
## 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(ggplot2)
library(kableExtra)
```

```
## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 2 > 1' in coercion to 'logical(1)'

##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
##     group_rows
```

Importing and processing the data set

Consider the data from the file "inflowtimeseries.txt". The data corresponds to the monthly inflow in m^3/s for some hydro power plants in Brazil. You will only use the last column of the data set which represents one hydro plant in the Amazon river basin. The data span the period from January 1931 to August 2011 and is provided by the Brazilian ISO.

For all parts of the assignment prepare the data set such that the model consider only the data from January 2000 up to December 2009. Leave the year 2010 of data (January 2010 to December 2010) for the out-of-sample analysis. Do **NOT** use data fro 2010 and 2011 for model fitting. You will only use it to compute forecast accuracy of your model.

Part I: Preparing the data sets

Q1

Read the file into a data frame. Prepare your time series data vector such that observations start in January 2000 and end in December 2009. Make you sure you specify the **start=** and **frequency=** arguments. Plot the time series over time, ACF and PACF.

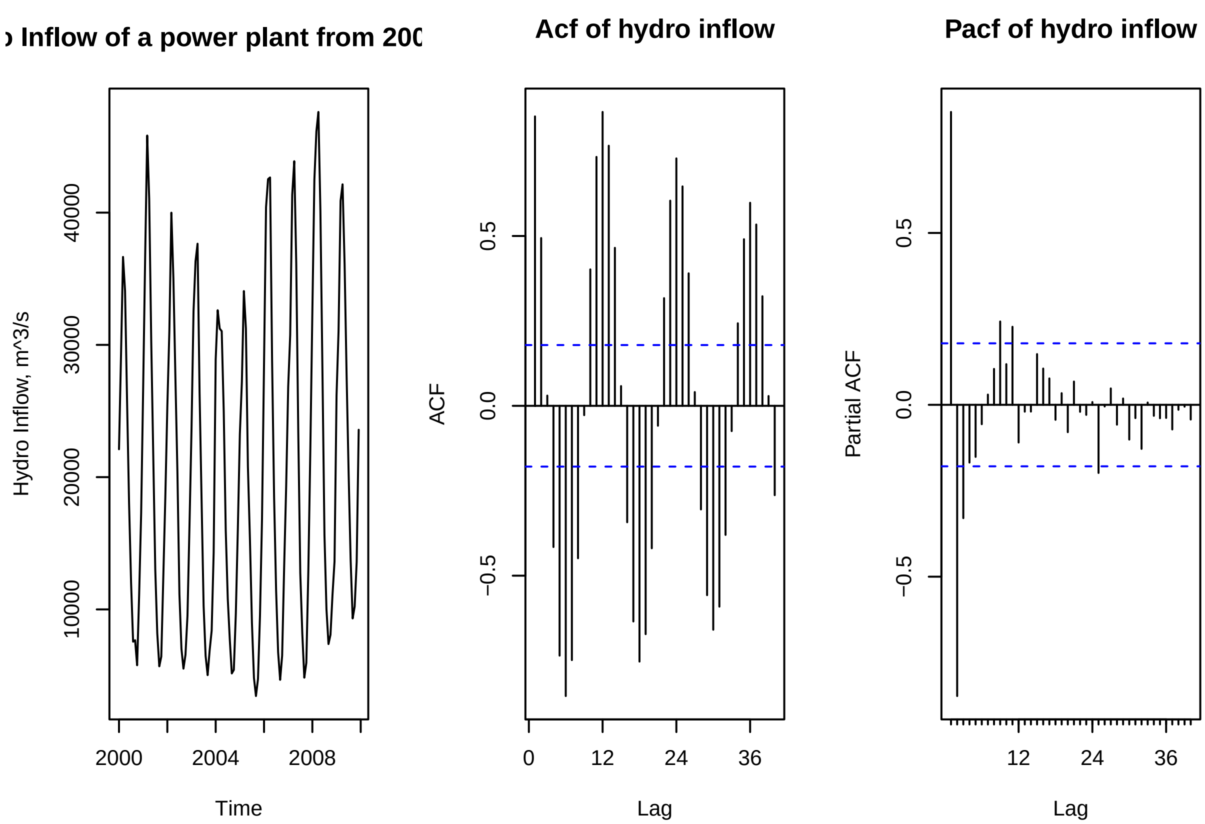
```
inflow_data <- read.table(file="../Data/inflowtimeseries.txt", header=FALSE,skip=0)

#last column of the table and date
inflow_data1 <- inflow_data[,1:2]
inflow_data2 <- inflow_data[,17]
inflow_data_hydro <- cbind(inflow_data1,inflow_data2)
```

```r
#add column names
colnames(inflow_data_hydro)=c("Month", "Year", "HP17")

#Date
inflow_data_hydro$Month <- unite(inflow_data_hydro, col = 'Time', c('Month','Year'), sep=" ")
inflow_data_hydro <- inflow_data_hydro[,1:1]
inflow_data_hydro$Time <- my(inflow_data_hydro$Time)

#January 2000 up to December 2009
inflow_data_hydro_2009 <- inflow_data_hydro[829:948,]
head(inflow_data_hydro_2009)
```

```
##           Time  HP17
## 829 2000-01-01 22107
## 830 2000-02-01 29247
## 831 2000-03-01 36651
## 832 2000-04-01 34089
## 833 2000-05-01 25821
## 834 2000-06-01 18007
```

```r
#ts
ts_inflow_data_hydro_2009 <- ts(inflow_data_hydro_2009[,2], start = c(2000,01), frequency = 12)
```

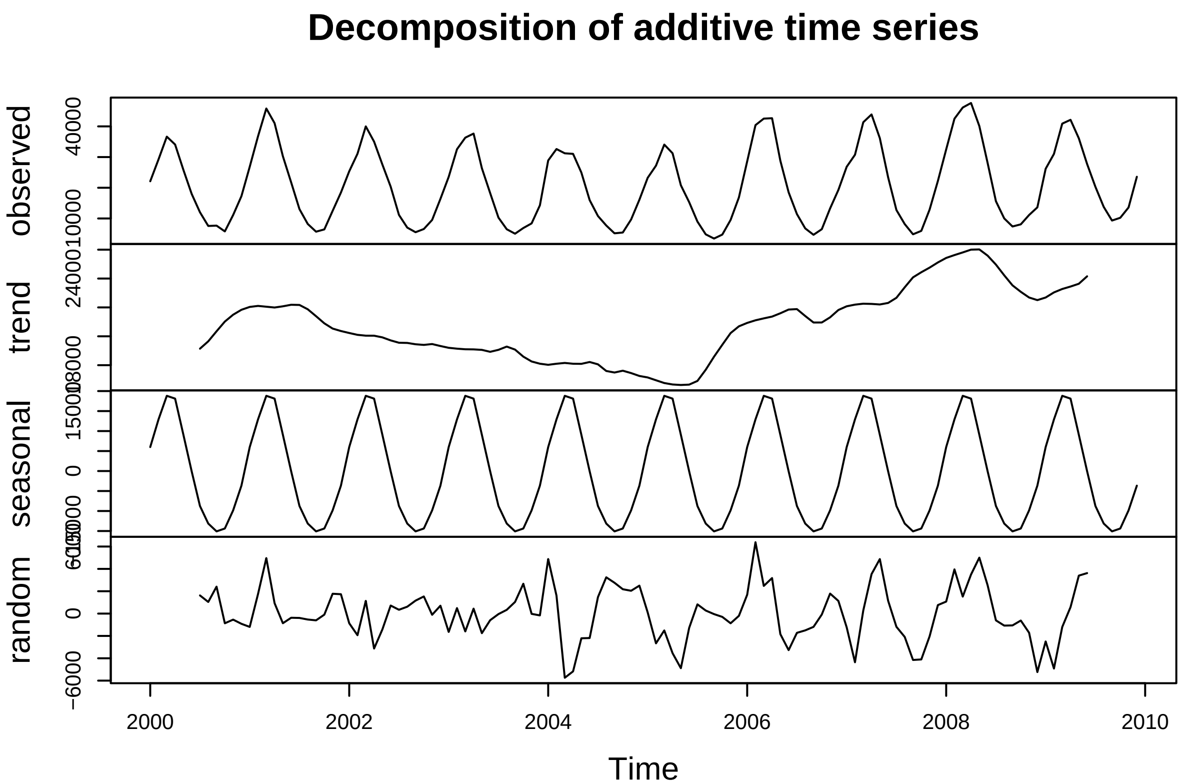
```r
#Acf and Pacf
par(mfrow=c(1,3))
plot(ts_inflow_data_hydro_2009, xlab = "Time", ylab = "Hydro Inflow, m^3/s", main = "Hydro Inflow of a p
Acf(ts_inflow_data_hydro_2009, lag =40, main = "Acf of hydro inflow")
Pacf(ts_inflow_data_hydro_2009, lag =40, main = "Pacf of hydro inflow")
```



Q2

Using the *decompose*() or *stl*() and the *seasadj*() functions create a series without the seasonal component, i.e., a deseasonalized inflow series. Plot the deseasonalized series and original series together using ggplot, make sure your plot includes a legend. Plot ACF and PACF for the deaseasonalized series. Compare with the plots obtained in Q1.

```
#decompose
decompose_hydro <- decompose(ts_inflow_data_hydro_2009, "additive")
plot(decompose_hydro)
```



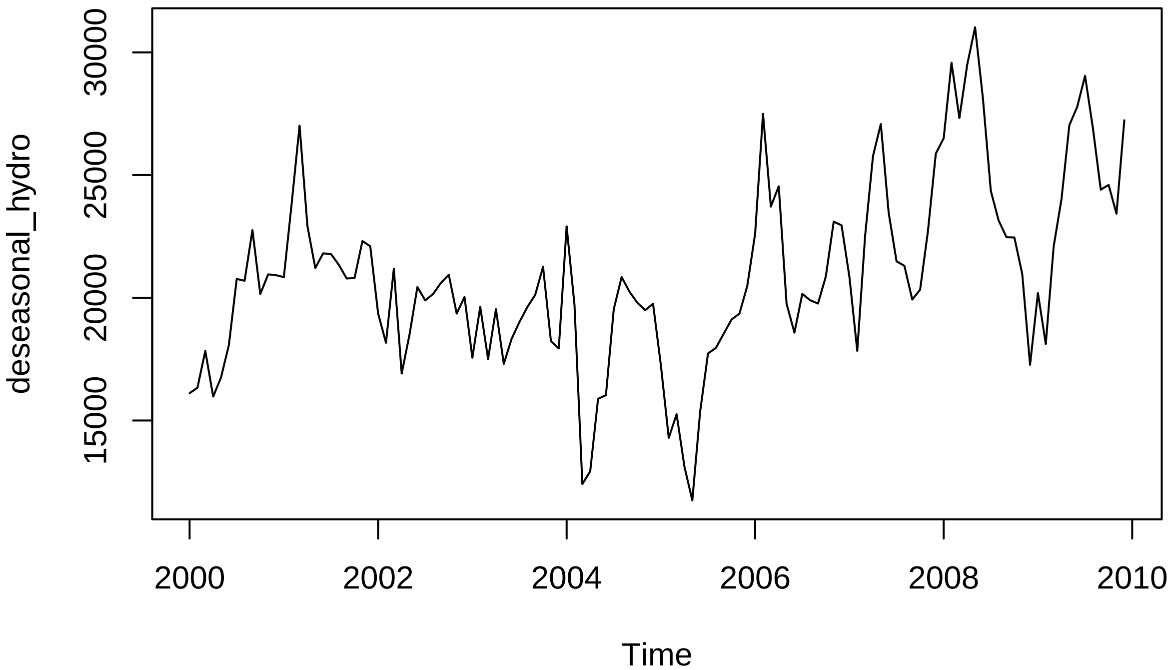

```
print(decompose_hydro)
```

```
## $x
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 2000 22107 29247 36651 34089 25821 18007 12027  7557  7670  5777 11116 17269
## 2001 26835 36764 45828 41038 30282 21745 13040  8204  5697  6429 12473 18447
## 2002 25356 31078 39996 35027 27575 20372 11153  7020  5519  6563  9518 16380
## 2003 23553 32547 36319 37651 26376 18267 10274  6488  5029  6888  8393 14285
## 2004 28901 32620 31225 31042 24951 15966 10793  7711  5159  5424  9656 16103
## 2005 23242 27204 34070 31232 20811 15307  8992  4820  3449  4742  9516 16836
## 2006 28607 40407 42531 42658 28826 18517 11419  6766  4675  6499 13263 19300
## 2007 26840 30748 41345 43887 36150 23392 12742  8168  4838  5956 12880 22223
## 2008 32493 42492 46138 47613 40093 28051 15616 10012  7378  8083 11131 13617
## 2009 26184 31029 40892 42138 36104 27713 20302 13772  9316 10225 13586 23583
##
## $seasonal
##              Jan         Feb         Mar         Apr         May
## 2000  5992.19367 12911.55478 18814.60571 18113.28164  9068.14275
## 2001  5992.19367 12911.55478 18814.60571 18113.28164  9068.14275
## 2002  5992.19367 12911.55478 18814.60571 18113.28164  9068.14275
```

```
## 2003   5992.19367  12911.55478  18814.60571  18113.28164   9068.14275
## 2004   5992.19367  12911.55478  18814.60571  18113.28164   9068.14275
## 2005   5992.19367  12911.55478  18814.60571  18113.28164   9068.14275
## 2006   5992.19367  12911.55478  18814.60571  18113.28164   9068.14275
## 2007   5992.19367  12911.55478  18814.60571  18113.28164   9068.14275
## 2008   5992.19367  12911.55478  18814.60571  18113.28164   9068.14275
## 2009   5992.19367  12911.55478  18814.60571  18113.28164   9068.14275
##              Jun          Jul          Aug          Sep          Oct
## 2000    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2001    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2002    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2003    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2004    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2005    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2006    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2007    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2008    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## 2009    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
##              Nov          Dec
## 2000  -9839.24151  -3652.44985
## 2001  -9839.24151  -3652.44985
## 2002  -9839.24151  -3652.44985
## 2003  -9839.24151  -3652.44985
## 2004  -9839.24151  -3652.44985
## 2005  -9839.24151  -3652.44985
## 2006  -9839.24151  -3652.44985
## 2007  -9839.24151  -3652.44985
## 2008  -9839.24151  -3652.44985
## 2009  -9839.24151  -3652.44985
##
## $trend
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2000       NA       NA       NA       NA       NA       NA 19141.83 19652.04
## 2001 22034.54 22103.71 22048.46 21993.42 22077.12 22182.75 22170.21 21871.67
## 2002 20229.21 20101.25 20044.50 20042.67 19925.12 19715.87 19554.62 19540.71
## 2003 19202.21 19143.42 19100.83 19093.96 19060.62 18926.46 19062.00 19287.88
## 2004 18026.79 18099.37 18155.75 18100.17 18091.79 18220.17 18060.12 17598.67
## 2005 17150.96 16955.46 16763.75 16664.08 16629.83 16654.54 16908.62 17682.29
## 2006 20926.21 21108.42 21240.58 21364.88 21594.21 21853.00 21882.04 21405.96
## 2007 22078.79 22192.33 22257.54 22241.71 22203.12 22308.96 22666.29 23391.17
## 2008 25427.00 25623.58 25806.25 26000.71 26016.46 25585.00 24963.54 24223.04
## 2009 22686.67 23038.58 23276.00 23446.00 23637.54 24155.08       NA       NA
##           Sep      Oct      Nov      Dec
## 2000 20347.62 21019.54 21494.96 21836.58
## 2001 21391.75 20898.29 20535.04 20365.04
## 2002 19448.71 19404.83 19464.21 19326.54
## 2003 19078.67 18591.04 18256.29 18101.04
## 2004 17491.54 17618.00 17453.42 17253.46
## 2005 18584.96 19413.58 20223.62 20691.33
## 2006 20954.08 20955.88 21312.25 21820.54
## 2007 24080.21 24435.17 24754.71 25113.12
## 2008 23526.83 23080.12 22685.79 22505.50
## 2009       NA       NA       NA       NA
##
```

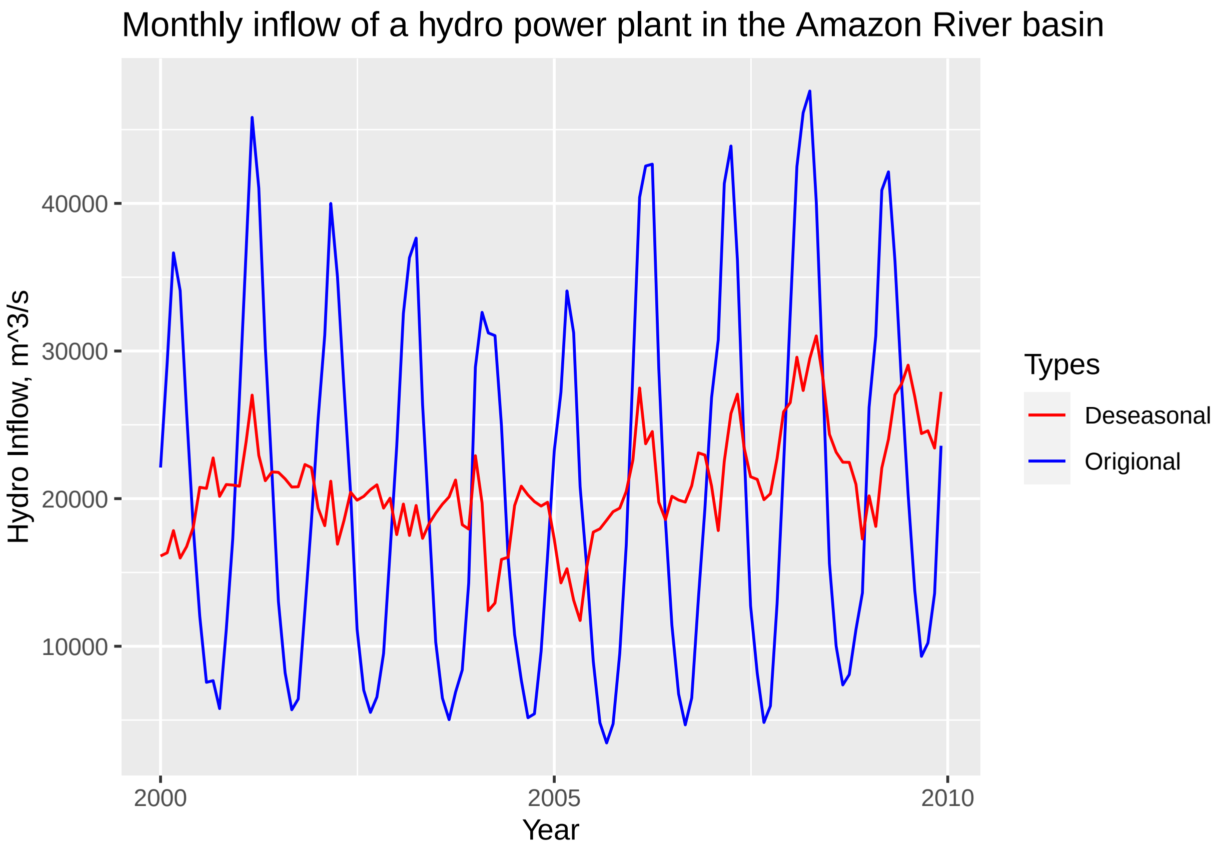
```
## $random
##              Jan         Feb         Mar         Apr         May         Jun
## 2000          NA          NA          NA          NA          NA          NA
## 2001 -1191.73534  1748.73688  4964.93596   931.30170  -863.26775  -372.11497
## 2002  -865.40201 -1934.80478  1136.89429 -3128.94830 -1418.26775   721.76003
## 2003 -1641.40201   492.02855 -1596.43904   443.76003 -1752.76775  -593.82330
## 2004  4882.01466  1609.07022 -5745.35571 -5171.44830 -2208.93441 -2188.53164
## 2005    98.84799 -2663.01312 -1508.35571 -3545.36497 -4886.97608 -1281.90664
## 2006  1688.59799  6387.02855  2475.81096  3179.84336 -1836.35108 -3270.36497
## 2007 -1230.98534 -4355.88812   272.85262  3532.01003  4878.73225  1148.67670
## 2008  1073.80633  3956.86188  1517.14429  3499.01003  5008.39892  2531.63503
## 2009 -2494.86034 -4921.13812 -1198.60571   578.71836  3398.31559  3623.55170
##              Jul         Aug         Sep         Oct         Nov         Dec
## 2000  1626.51929  1041.21373  2412.29244  -867.61497  -539.71682  -915.13349
## 2001  -388.85571  -531.41127  -604.83256   -94.36497  1777.19985  1734.40818
## 2002   339.72762   615.54707  1160.20910  1533.09336  -106.96682   705.90818
## 2003   -46.64738   336.38040  1040.25077  2671.88503   -24.05015  -163.59182
## 2004  1474.22762  3248.58873  2757.37577  2180.92670  2041.82485  2501.99151
## 2005   824.72762   273.96373   -46.04090  -296.65664  -868.38349  -202.88349
## 2006 -1721.68904 -1503.70293 -1189.16590   -81.94830  1789.99151  1131.90818
## 2007 -1182.93904 -2086.91127 -4152.29090 -4104.23997 -2035.46682   762.32485
## 2008  -606.18904 -1074.78627 -1058.91590  -622.19830 -1715.55015 -5236.05015
## 2009          NA          NA          NA          NA          NA          NA
##
## $figure
##  [1]   5992.19367  12911.55478  18814.60571  18113.28164   9068.14275
##  [6]    -65.63503  -8741.35262 -13136.25540 -15089.91744 -14374.92670
## [11]  -9839.24151  -3652.44985
##
## $type
## [1] "additive"
##
## attr(,"class")
## [1] "decomposed.ts"
```

```r
#create non-seasonal natural gas time series
#seasonal adjustment to original data, only works for decompose
deseasonal_hydro <- seasadj(decompose_hydro)
plot(deseasonal_hydro)
```

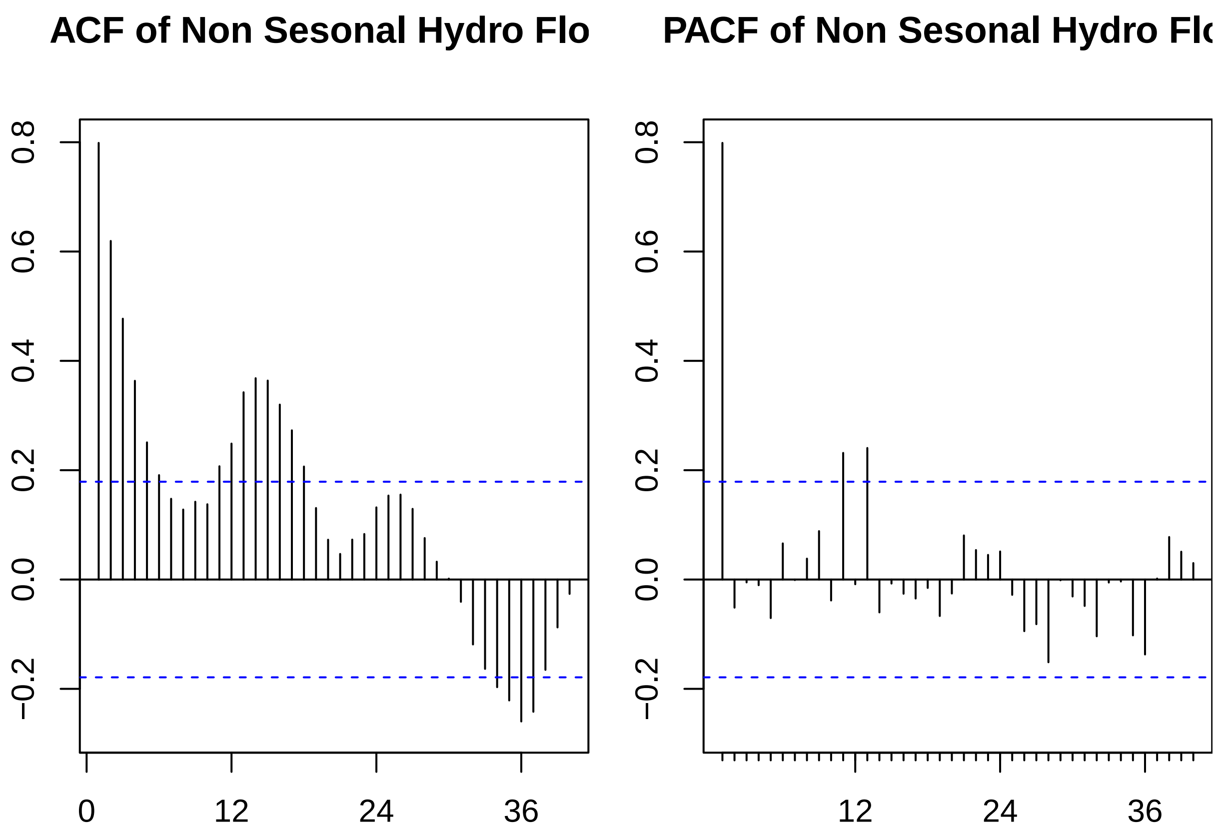



```
#Plot the deseasonalized series and original series together using ggplot with legend

ggplot(data = inflow_data_hydro_2009, aes(x=Time, y=inflow_data_hydro_2009[,2]))+
  geom_line(aes(x=Time, y=ts_inflow_data_hydro_2009, color = "Origional"))+
  geom_line(aes(x=Time, y=deseasonal_hydro, color = "Deseasonal"))+
  ylab("Hydro Inflow, m^3/s")+
  xlab("Year")+
  ggtitle("Monthly inflow of a hydro power plant in the Amazon River basin")+
  scale_color_manual(name = "Types", values = c("Origional" = "blue", "Deseasonal" = "red"), labels=c("
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```




```
#Comparing ACFs
#seasonality is removed
par(mar=c(3,3,3,0));par(mfrow=c(1,2))
Acf(deseasonal_hydro,lag.max=40,main="ACF of Non Sesonal Hydro Flow")
Pacf(deseasonal_hydro,lag.max=40,main="PACF of Non Sesonal Hydro Flow")
```



Answer: Compared the plots in Q1, the acf of nonseasonal hydro flow have slow decay instead of seasonal spike. Therefore, we removed the seasonality. Also, we removed those little spikes for the seasonality in the pacf of nonseasonal hydro flow compared to the pacf of hydro flow.

Part II: Forecasting with ARIMA models and its variations

Q3

Fit a non-seasonal ARIMA(p, d, q) model using the auto.arima() function to the non-seasonal data. Forecast 12 months ahead of time using the $forecast()$ function. Plot your forecasting results and further include on the plot the last year of non-seasonal data to compare with forecasted values (similar to the plot on the lesson file for M10).

```
#auto arima
ARIMA_autofit <- auto.arima(deseasonal_hydro,max.D=0,max.P = 0,max.Q=0)
print(ARIMA_autofit)
```

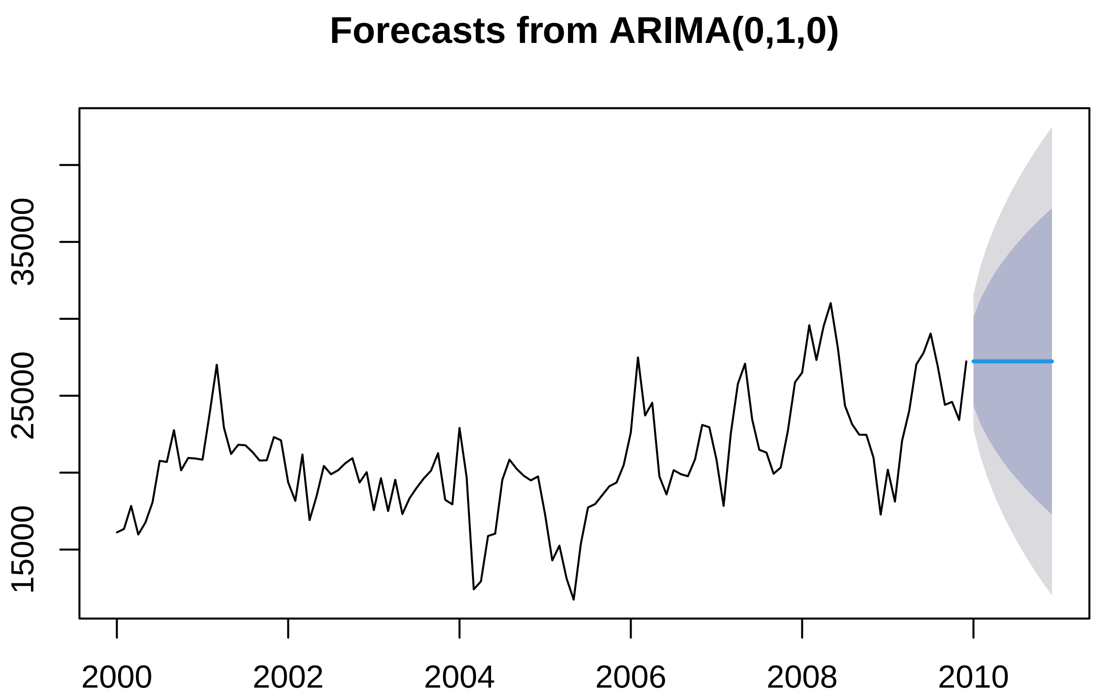
```
## Series: deseasonal_hydro
## ARIMA(0,1,0)
##
## sigma^2 = 5031380:  log likelihood = -1087.01
## AIC=2176.02   AICc=2176.05   BIC=2178.8
```

```
#forecast
ARIMA_forecast <- forecast(object = ARIMA_autofit, h = 12)
print(ARIMA_forecast)
```

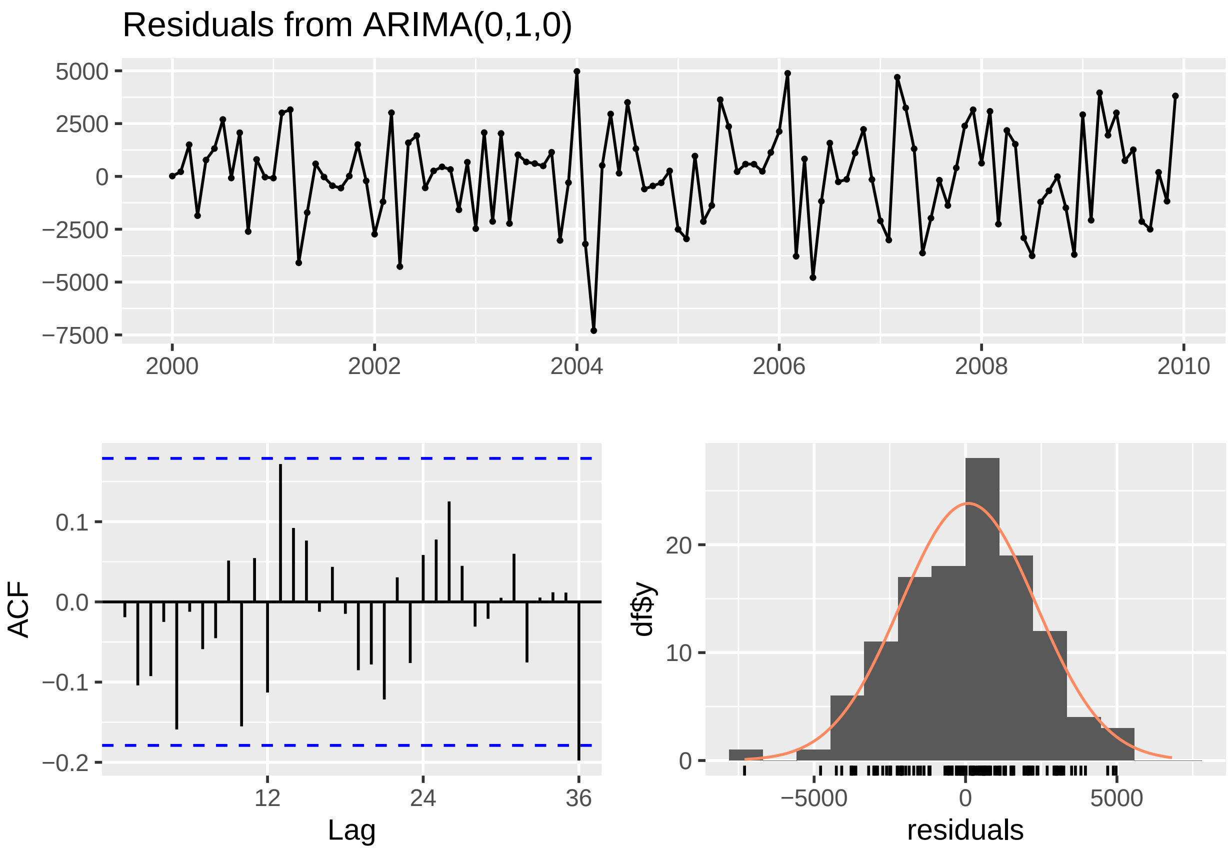
```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 2010       27235.45 24360.84 30110.06 22839.11 31631.79
## Feb 2010       27235.45 23170.13 31300.77 21018.08 33452.82
```

```
## Mar 2010       27235.45 22256.47 32214.43 19620.76 34850.14
## Apr 2010       27235.45 21486.22 32984.68 18442.76 36028.14
## May 2010       27235.45 20807.62 33663.28 17404.93 37065.97
## Jun 2010       27235.45 20194.11 34276.79 16466.65 38004.25
## Jul 2010       27235.45 19629.93 34840.97 15603.82 38867.08
## Aug 2010       27235.45 19104.81 35366.09 14800.71 39670.19
## Sep 2010       27235.45 18611.61 35859.29 14046.42 40424.48
## Oct 2010       27235.45 18145.12 36325.78 13332.99 41137.91
## Nov 2010       27235.45 17701.43 36769.47 12654.43 41816.47
## Dec 2010       27235.45 17277.49 37193.41 12006.07 42464.83
```

```
plot(ARIMA_forecast)
```




```
checkresiduals(ARIMA_forecast)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,0)
## Q* = 24.208, df = 24, p-value = 0.4498
##
## Model df: 0.   Total lags used: 24
```

```
#plot your forecasting results and further include on the plot the last year of non-seasonal data to co

#create time series of 2010 data
#January 2000 up to December 2010
inflow_data_hydro_2010 <- inflow_data_hydro[829:960,]
#ts
ts_inflow_data_hydro_2010 <- ts(inflow_data_hydro_2010[,2], start = c(2000,01), frequency = 12)
#decompose
decompose_hydro_2010 <- decompose(ts_inflow_data_hydro_2010, "additive")
#seasonal adjustment to original data, only works for decompose
deseasonal_hydro_2010 <- seasadj(decompose_hydro_2010)
#only for 2010
deseasonal2_hydro_2010 <- window(deseasonal_hydro_2010, start = c(2010,01), end = c(2010,12))
```

```
autoplot(deseasonal_hydro, series="Deseasonal 2000 to 2009",PI=FALSE) +
    autolayer(deseasonal2_hydro_2010,series="Deseasonal for 2010",PI=FALSE) +
    autolayer(ARIMA_forecast,series="Forecast",PI=FALSE) +
    ylab("Hydro flow, m^3/s") +
    xlab("Year") +
    labs(col="Models")
```

```
## Warning in ggplot2::geom_line(ggplot2::aes_(group = ~series, colour =
## ~series), : Ignoring unknown parameters: `PI`

## Warning in ggplot2::geom_line(ggplot2::aes_(x = ~timeVal, y = ~seriesVal, :
## Ignoring unknown parameters: `PI`
```

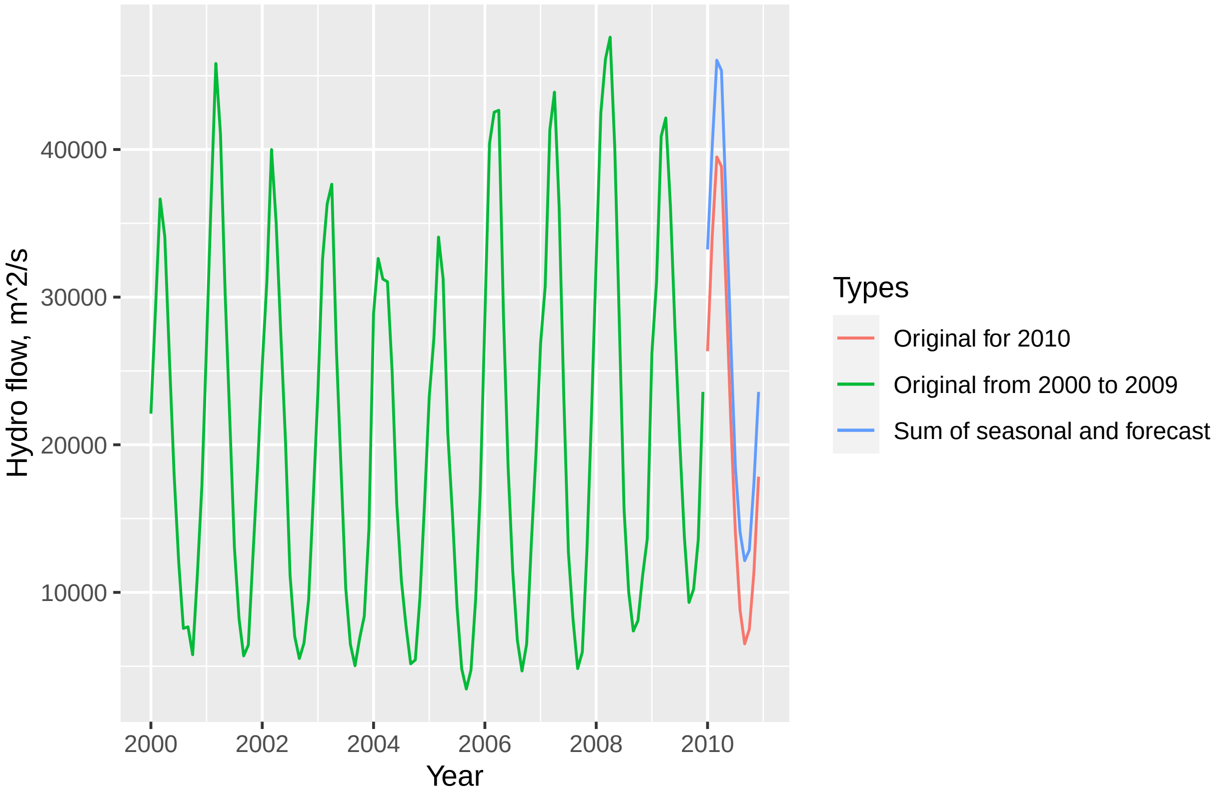
Q4

Put the seasonality back on your forcasted values and compare with the original seasonal data values. *Hint* : One way to do it is by summing the last year of the seasonal component from your decompose object to the forecasted series.

```
#Seasonal component from decompose
Seasonal_decompose_inflow <- as.data.frame(decompose_hydro$seasonal)
Seasonal_decompose_inflow1 <- cbind(inflow_data_hydro_2009[,1], Seasonal_decompose_inflow)
Sea_decom_inflow_2009 <- Seasonal_decompose_inflow[109:120,]

#Add seasonality to the forecasted series
Sum_Sea_forecast <- Sea_decom_inflow_2009 + ARIMA_forecast$mean

#origional seasonal data values (2010)
inflow_data_only_2010 <- inflow_data_hydro[949:960,]
ts_inflow_data_only_2010 <- ts(inflow_data_only_2010[,2], start = c(2010,1), end = c(2010,12), frequenc

#plot
autoplot(ts_inflow_data_hydro_2009, series = "Original from 2000 to 2009") +
    autolayer(Sum_Sea_forecast,series="Sum of seasonal and forecast",PI=FALSE) +
    autolayer(ts_inflow_data_only_2010,series="Original for 2010",PI=FALSE) +
    ylab("Hydro flow, m^2/s") +
    xlab("Year") +
```

```
    labs(col="Types")
```

```
## Warning in ggplot2::geom_line(ggplot2::aes_(x = ~timeVal, y = ~seriesVal, : Ignoring unknown paramete
## Ignoring unknown parameters: `PI`
```



Q5

Repeat Q3 for the original data, but now fit a seasonal ARIMA$(p, d, q)x(P, D, Q)_{12}$ also using the auto.arima().

```
#auto sarima
SARIMA_autofit <- auto.arima(ts_inflow_data_hydro_2009)
print(SARIMA_autofit)
```

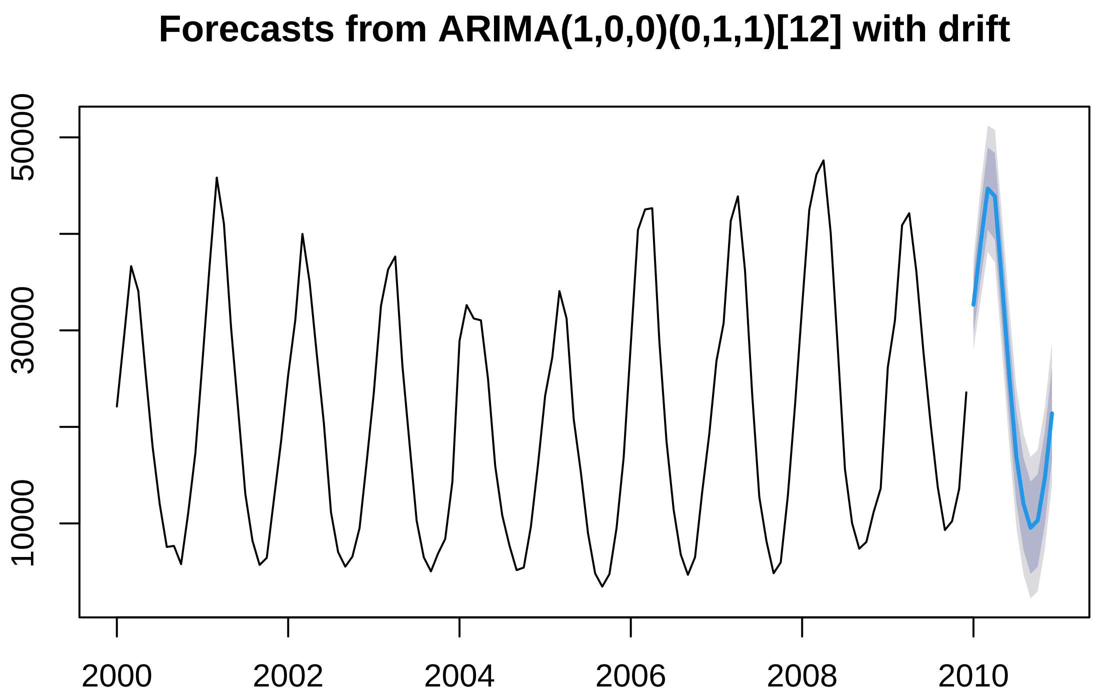
```
## Series: ts_inflow_data_hydro_2009
## ARIMA(1,0,0)(0,1,1)[12] with drift
##
## Coefficients:
##          ar1     sma1    drift
##       0.7739  -0.8724  54.7963
## s.e.  0.0614   0.1767  25.8402
##
## sigma^2 = 5566545:  log likelihood = -999.07
## AIC=2006.14   AICc=2006.52   BIC=2016.86
```

```
#forecast
SARIMA_forecast <- forecast(object = SARIMA_autofit, h = 12)
print(SARIMA_forecast)
```

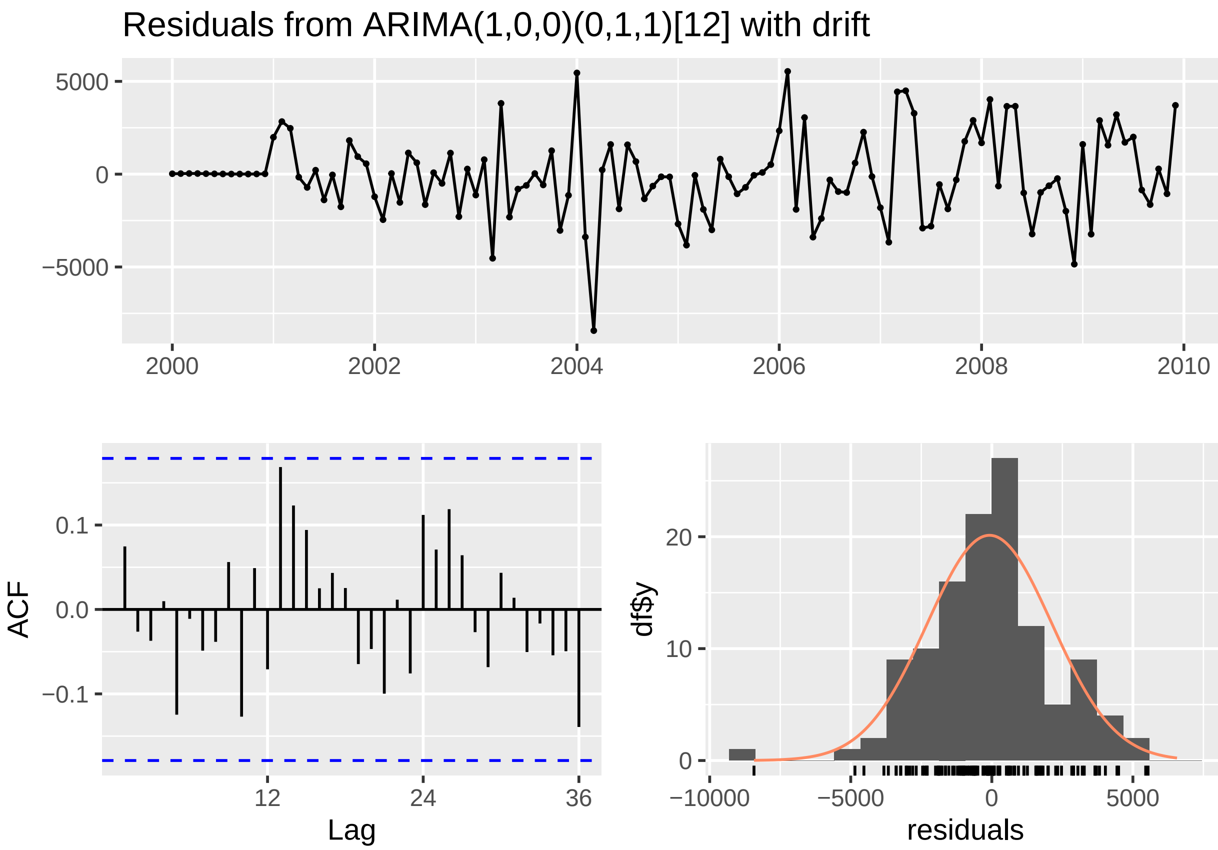
```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2010      32655.388 29601.817 35708.96 27985.355 37325.42
```

```
## Feb 2010      38975.665 35118.619 42832.71 33076.822 44874.51
## Mar 2010      44694.285 40427.930 48960.64 38169.459 51219.11
## Apr 2010      43882.582 39388.955 48376.21 37010.173 50754.99
## May 2010      34835.315 30210.978 39459.65 27763.001 41907.63
## Jun 2010      25402.604 20701.808 30103.40 18213.357 32591.85
## Jul 2010      16934.482 12188.618 21680.35  9676.308 24192.66
## Aug 2010      12051.454  7279.018 16823.89  4752.643 19350.26
## Sep 2010       9546.441  4758.520 14334.36  2223.948 16868.93
## Oct 2010      10305.013  5508.438 15101.59  2969.285 17640.74
## Nov 2010      14706.158  9905.401 19506.92  7364.033 22048.28
## Dec 2010      21396.427 16594.823 26198.03 14053.008 28739.84
```

```
plot(SARIMA_forecast)
```




```
checkresiduals(SARIMA_forecast)
```

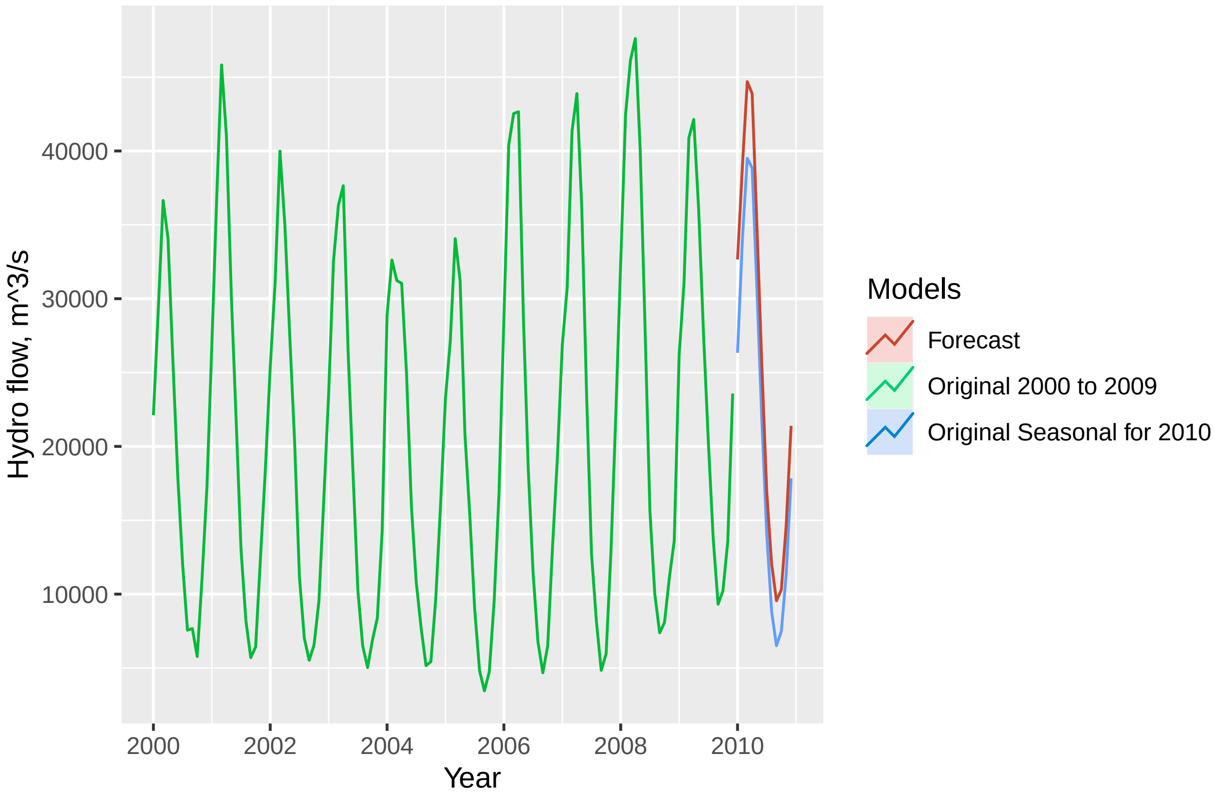



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(0,1,1)[12] with drift
## Q* = 19.881, df = 22, p-value = 0.5905
##
## Model df: 2.   Total lags used: 24
```

```
#plot your forecasting results and further include on the plot the last year of seasonal data to compar

autoplot(ts_inflow_data_hydro_2009, series="Original 2000 to 2009",PI=FALSE) +
    autolayer(ts_inflow_data_only_2010,series="Original Seasonal for 2010",PI=FALSE) +
    autolayer(SARIMA_forecast,series="Forecast",PI=FALSE) +
    ylab("Hydro flow, m^3/s") +
    xlab("Year") +
    labs(col="Models")
```

```
## Warning in ggplot2::geom_line(ggplot2::aes_(group = ~series, colour =
## ~series), : Ignoring unknown parameters: `PI`
```

```
## Warning in ggplot2::geom_line(ggplot2::aes_(x = ~timeVal, y = ~seriesVal, :
## Ignoring unknown parameters: `PI`
```

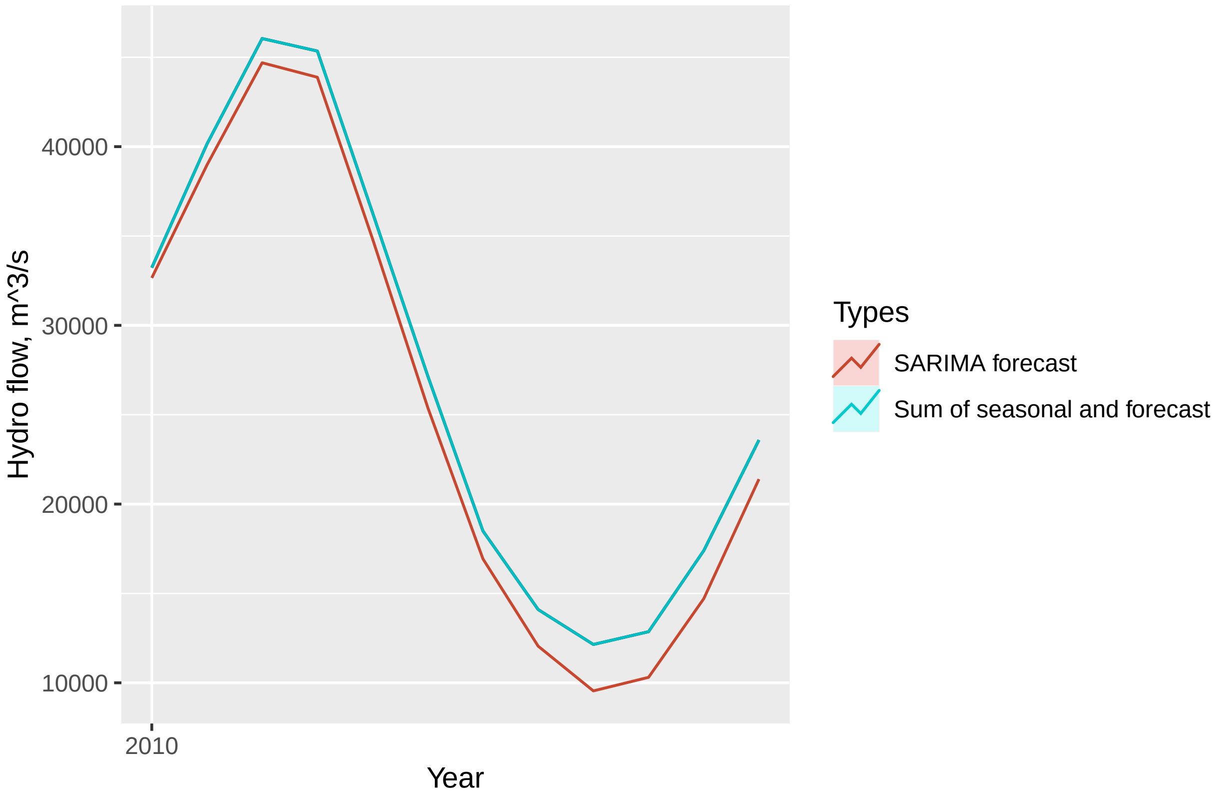


Q6

Compare the plots from Q4 and Q5 using the autoplot() funwction.

```
#plot
autoplot(Sum_Sea_forecast) +
    autolayer(Sum_Sea_forecast,series="Sum of seasonal and forecast",PI=FALSE) +
    autolayer(SARIMA_forecast,series="SARIMA forecast",PI=FALSE) +
    ylab("Hydro flow, m^3/s") +
    xlab("Year") +
    labs(col="Types")
```

```
## Warning in ggplot2::geom_line(ggplot2::aes_(x = ~timeVal, y = ~seriesVal, :
## Ignoring unknown parameters: `PI`
```



Part III: Forecasting with Other Models

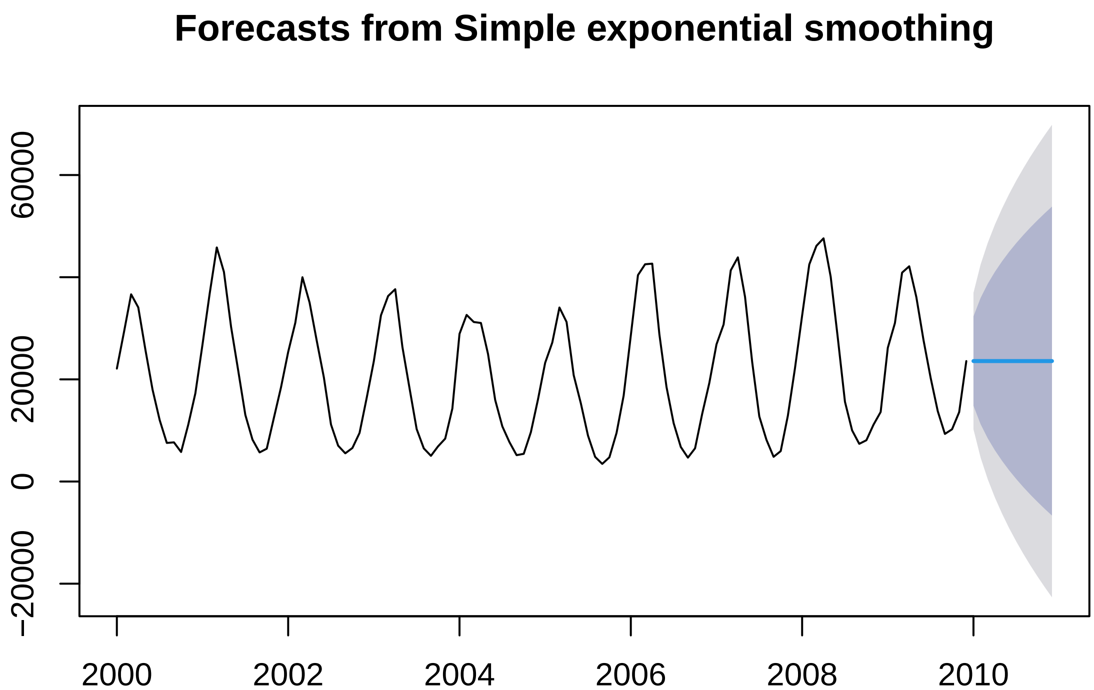
Q7

Fit an exponential smooth model to the original time series using the function *ses*() from package `forecast`. Note that this function automatically do the forecast. Do not forget to set the arguments: silent=FALSE and holdout=FALSE, so that the plot is produced and the forecast is for the year of 2010.

```
# Exponential smooth model to the original ts
SES_origional=ses(y = ts_inflow_data_hydro_2009, h = 12, holdout = FALSE, silent = FALSE)
print(SES_origional)
```

```
##          Point Forecast      Lo 80    Hi 80      Lo 95    Hi 95
## Jan 2010          23582 14849.8153 32314.19  10227.276 36936.72
## Feb 2010          23582 11233.4433 35930.56   4696.512 42467.49
## Mar 2010          23582  8458.4205 38705.58    452.481 46711.52
## Apr 2010          23582  6118.9402 41045.06  -3125.445 50289.45
## May 2010          23582  4057.8032 43106.20  -6277.682 53441.68
## Jun 2010          23582  2194.3852 44969.62  -9127.534 56291.53
## Jul 2010          23582   480.7907 46683.21 -11748.251 58912.25
## Aug 2010          23582 -1114.1874 48278.19 -14187.559 61351.56
## Sep 2010          23582 -2612.2260 49776.23 -16478.612 63642.61
## Oct 2010          23582 -4029.1079 51193.11 -18645.546 65809.55
## Nov 2010          23582 -5376.7480 52540.75 -20706.583 67870.58
## Dec 2010          23582 -6664.4029 53828.40 -22675.882 69839.88
```

```
plot(SES_origional)
```



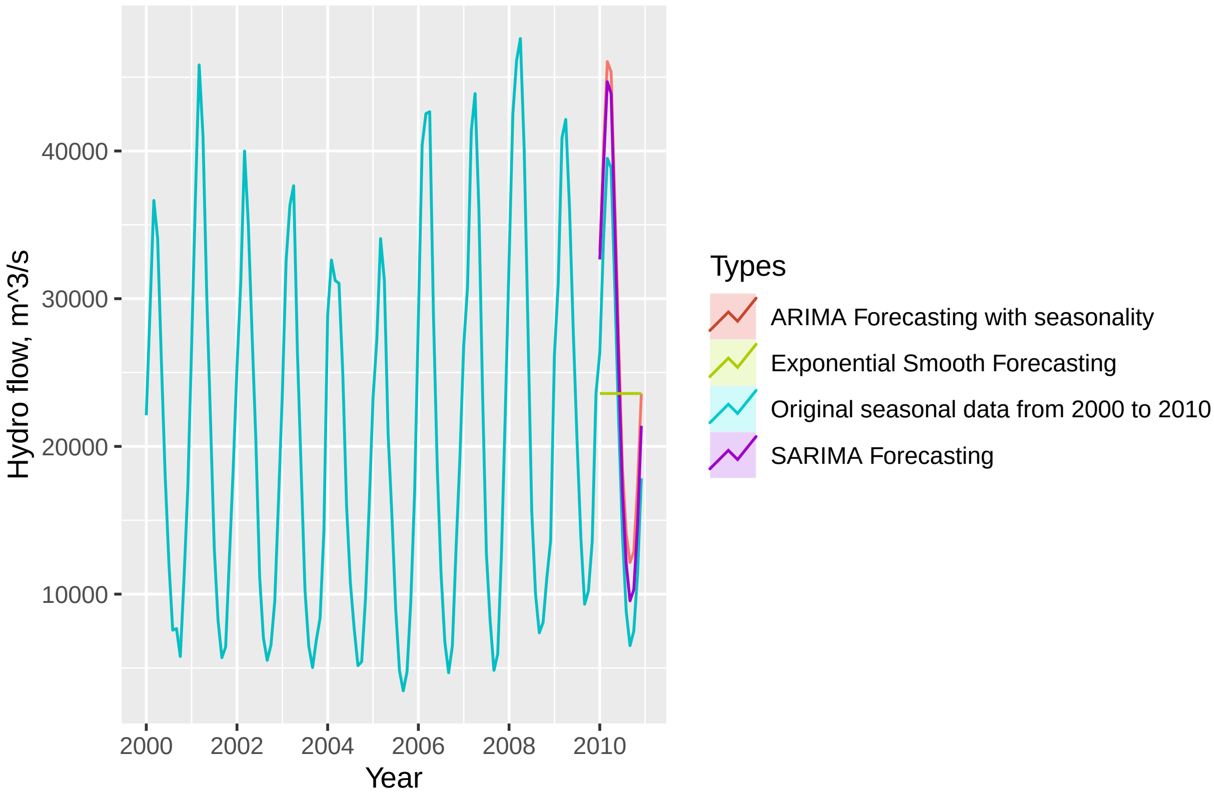
Part IV: Checking Forecast Accuracy

Q8

Make one plot with the complete original seasonal historical data (Jan 2000 to Dec 2010). Now add the forecasts from each of the developed models in parts Q4, Q5, Q7 and Q8. You can do it using the autoplot() combined with autolayer(). If everything is correct in terms of time line, the forecasted lines should appear only in the final year. If you decide to use ggplot() you will need to create a data frame with all the series will need to plot. Remember to use a different color for each model and add a legend in the end to tell which forecast lines corresponds to each model.

```
autoplot(ts_inflow_data_hydro_2010, series = "Original seasonal data from 2000 to 2010", PI=FALSE) +
      autolayer(Sum_Sea_forecast,series="ARIMA Forecasting with seasonality",PI=FALSE)+
      autolayer(SARIMA_forecast,series="SARIMA Forecasting",PI=FALSE) +
      autolayer(SES_origional,series="Exponential Smooth Forecasting",PI=FALSE) +
    ylab("Hydro flow, m^3/s") +
    xlab("Year") +
    labs(col="Types")
```

```
## Warning in ggplot2::geom_line(ggplot2::aes_(group = ~series, colour =
## ~series), : Ignoring unknown parameters: `PI`

## Warning in ggplot2::geom_line(ggplot2::aes_(x = ~timeVal, y = ~seriesVal, :
## Ignoring unknown parameters: `PI`
```



Q9

From the plot in Q9 which model or model(s) are leading to the better forecasts? Explain your answer. Hint: Think about which models are doing a better job forecasting the high and low inflow months for example.

> Answer:SARIMA model is leading to the better forecast because it did a better job forecasting the high and low inflow months. The difference between the SARIMA forecast's values and the actual historical data values is the smallest compared to the other models (the highest point and lowest point in the graph).

Q10

Now compute the following forecast metrics we learned in class: RMSE and MAPE, for all the models you plotted in part Q9. You can do this by hand since your have forecasted and observed values for the year of 2010. Or you can use R function *accuracy*() from package "forecast" to do it. Build and a table with the results and highlight the model with the lowest MAPE. Does the lowest MAPE corresponds match your answer for part Q10?

```
last_obs1 <- as.numeric(inflow_data_only_2010[,2])

#Model 1: ARIMA Forecasting with seasonality
ARIMA_forecast_scores <- accuracy(Sum_Sea_forecast,last_obs1)  #store the performance metrics

#Model 2: SARIMA forecast
SARIMA_forecast_scores <- accuracy(SARIMA_forecast$mean,last_obs1)

# Model 3:  SES
SES_scores <- accuracy(SES_origional$mean,last_obs1)
```

Table 1: Forecast Accuracy for Seasonal Data

	ME	RMSE	MAE	MPE	MAPE
ARIMA	-5818.450	5852.520	5818.450	-38.04169	38.04169
SARIMA	-4031.818	4171.259	4031.818	-23.87315	23.87315
SES	-2165.000	11888.481	10718.167	-59.74209	83.65282

```r
#create data frame
seas_scores <- as.data.frame(rbind(ARIMA_forecast_scores, SARIMA_forecast_scores, SES_scores))
row.names(seas_scores) <- c("ARIMA", "SARIMA","SES")
```

```r
kbl(seas_scores,
      caption = "Forecast Accuracy for Seasonal Data",
      digits = array(5,ncol(seas_scores))) %>%
  kable_styling(full_width = FALSE, position = "center") %>%
  #highlight model with lowest MAPE
  kable_styling(latex_options="striped", stripe_index = which.min(seas_scores[,"MAPE"]))
```

```r
#choose model with lowest MAPE
best_model_index <- which.min(seas_scores[,"MAPE"])
cat("The best model by MAPE is:", row.names(seas_scores[best_model_index,]))
```

```
## The best model by MAPE is: SARIMA
```

Answer: Yes, the lowest MAPE corresponds match my answer for part Q9 that SARIMA model is the best models by MAPE.