Exercise 3 - Veronica Bayani

Name: Veronica Bayani Student Number: 2009-00574

I. Use the sale_app data (Total Sales of Appliance Units in the Philippines, Jan 2000 – Dec 2009) in PhilMonthlyData.csv.

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.2

## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2

## Warning: package 'ggplot2' was built under R version 4.2.2

## Warning: package 'readr' was built under R version 4.2.2

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

```
library(fpp2)
```

```
## Warning: package 'fpp2' was built under R version 4.2.2

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
## -- Attaching packages ---------------------------------------------- fpp2 2.4 --
## v forecast  8.18     v expsmooth 2.3
## v fma       2.4

## Warning: package 'forecast' was built under R version 4.2.2

##
```

```
library(tinytex)
```

```
## Warning: package 'tinytex' was built under R version 4.2.2
```

```
library(forecast)

philmon <- read.csv("PhilMonthlyData.csv", stringsAsFactors = FALSE, na.strings = c("NA"))
```

Getting the Sales of Appliances in Units Sold from 2000 to 2009

```
saleapp <- ts(na.omit(philmon$sale_app),start=c(2000,1),frequency=12)
saleapp
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2000 474500.0 491300.0 561800.0 528900.0 663200.0 665700.0 524800.0 555000.0
## 2001 468800.0 435100.0 527800.0 491700.0 558200.0 559400.0 487100.0 465800.0
## 2002 424400.0 413300.0 482700.0 613800.0 690600.0 637600.0 564300.0 536900.0
## 2003 518495.7 423300.0 560250.0 601000.0 638400.0 596400.0 580100.0 491400.0
## 2004 509400.0 508500.0 668500.0 664300.0 660100.0 627100.0 565300.0 543800.0
## 2005 452500.0 441300.0 484600.0 537300.0 547400.0 519400.0 483200.0 446600.0
## 2006 369300.0 335300.0 405200.0 422300.0 462300.0 428100.0 386700.0 372400.0
## 2007 257400.0 265300.0 310000.0 283100.0 342200.0 326350.0 258700.0 254200.0
## 2008 299503.0 287403.0 346505.0 306229.0 322493.0 322071.0 290698.0 278103.0
## 2009 238044.7 217870.3 303801.5 321495.3 373995.3 348701.5 284970.3 261957.8
##           Sep      Oct      Nov      Dec
## 2000 493000.0 513000.0 581300.0 676900.0
## 2001 438200.0 536700.0 538900.0 568100.0
## 2002 527400.0 592100.0 654800.0 714600.0
## 2003 515200.0 642800.0 618300.0 730500.0
## 2004 557200.0 625300.0 708600.0 710100.0
## 2005 477300.0 471000.0 529400.0 592744.1
## 2006 351600.0 338800.0 426300.0 438500.0
## 2007 265950.0 304000.0 348000.0 373800.0
## 2008 314734.0 344043.5 391781.0 441736.5
## 2009 256926.5 306657.8 354395.3 404350.8
```

Split the data into training and test data set:

Training dataset = Jan 2000 – Dec 2007; and Test dataset = Jan 2008 – Dec 2009.

```
#train dataset
saleapp_train <- window(saleapp,start=2000,end=c(2007,12))
saleapp_train
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2000 474500.0 491300.0 561800.0 528900.0 663200.0 665700.0 524800.0 555000.0
## 2001 468800.0 435100.0 527800.0 491700.0 558200.0 559400.0 487100.0 465800.0
## 2002 424400.0 413300.0 482700.0 613800.0 690600.0 637600.0 564300.0 536900.0
## 2003 518495.7 423300.0 560250.0 601000.0 638400.0 596400.0 580100.0 491400.0
## 2004 509400.0 508500.0 668500.0 664300.0 660100.0 627100.0 565300.0 543800.0
## 2005 452500.0 441300.0 484600.0 537300.0 547400.0 519400.0 483200.0 446600.0
## 2006 369300.0 335300.0 405200.0 422300.0 462300.0 428100.0 386700.0 372400.0
## 2007 257400.0 265300.0 310000.0 283100.0 342200.0 326350.0 258700.0 254200.0
##           Sep      Oct      Nov      Dec
## 2000 493000.0 513000.0 581300.0 676900.0
## 2001 438200.0 536700.0 538900.0 568100.0
## 2002 527400.0 592100.0 654800.0 714600.0
## 2003 515200.0 642800.0 618300.0 730500.0
## 2004 557200.0 625300.0 708600.0 710100.0
## 2005 477300.0 471000.0 529400.0 592744.1
## 2006 351600.0 338800.0 426300.0 438500.0
## 2007 265950.0 304000.0 348000.0 373800.0
```

```
#test dataset
saleapp_test <- window(saleapp,start=2008,end=c(2009,12))
saleapp_test
```

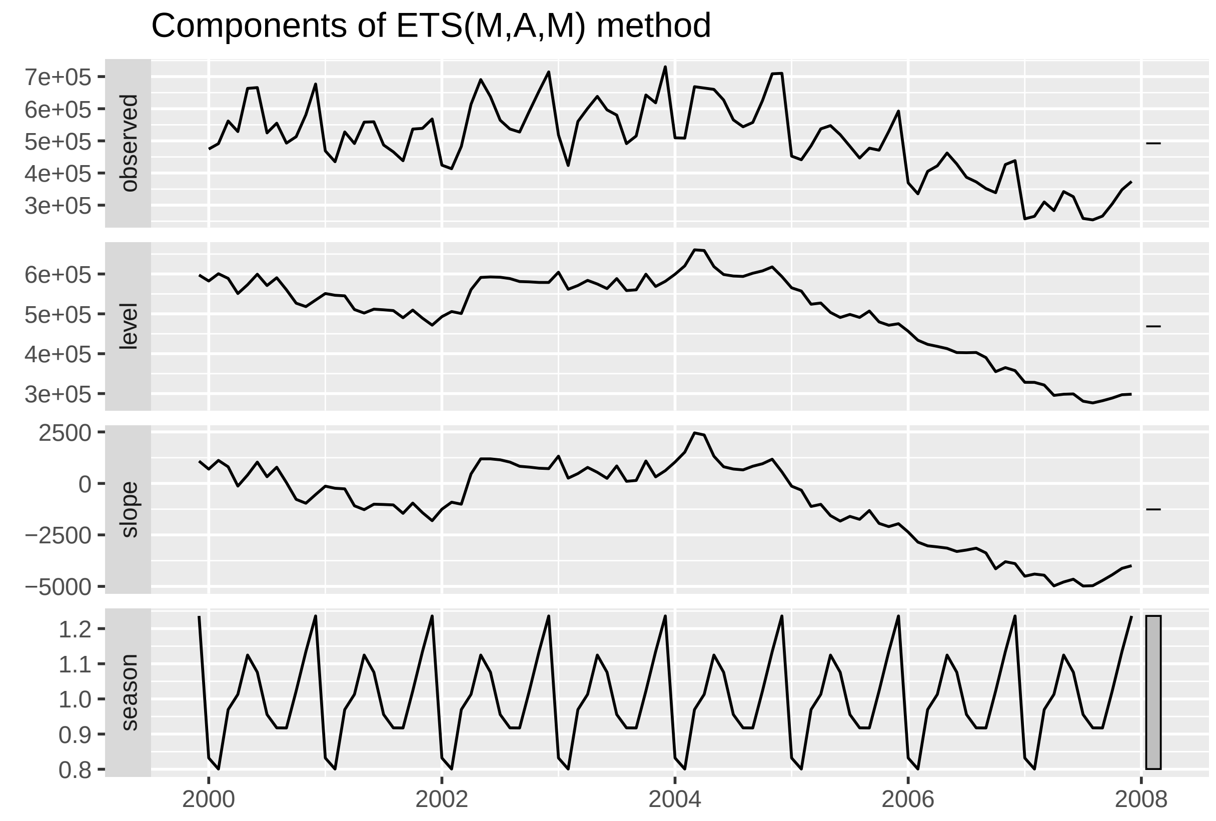
```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2008 299503.0 287403.0 346505.0 306229.0 322493.0 322071.0 290698.0 278103.0
## 2009 238044.7 217870.3 303801.5 321495.3 373995.3 348701.5 284970.3 261957.8
##           Sep      Oct      Nov      Dec
## 2008 314734.0 344043.5 391781.0 441736.5
## 2009 256926.5 306657.8 354395.3 404350.8
```

1) [2pts] Using the ets() function, show the best performing model based on the AICc of the training dataset and show the state space system of equations form of the model with estimated parameter values plugged in to the system of equation (for reference, see "Other ETS Models" in https://otexts.com/fpp2/ets.html)

```
saleapp_ets <- ets(saleapp_train, model="ZZZ")
summary(saleapp_ets)
```

```
## ETS(M,A,M)
##
## Call:
##  ets(y = saleapp_train, model = "ZZZ")
##
##   Smoothing parameters:
##     alpha = 0.5734
##     beta  = 0.0139
##     gamma = 2e-04
##
##   Initial states:
##     l = 597464.3559
##     b = 1084.4734
##     s = 1.2361 1.1346 1.0226 0.9175 0.9177 0.9557
##            1.0759 1.1248 1.013 0.9696 0.8006 0.832
##
##   sigma:  0.0714
##
##      AIC     AICc      BIC
## 2463.934 2471.780 2507.528
##
## Training set error measures:
##                     ME    RMSE   MAE       MPE     MAPE      MASE         ACF1
## Training set -3819.613 33707.2 26105 -1.060642 5.279745 0.3384195 -0.01535319
```

```r
autoplot(saleapp_ets)
```



The best performing model based on the etc function is the ETS(M,A,M) model since it has the lowest AICc.

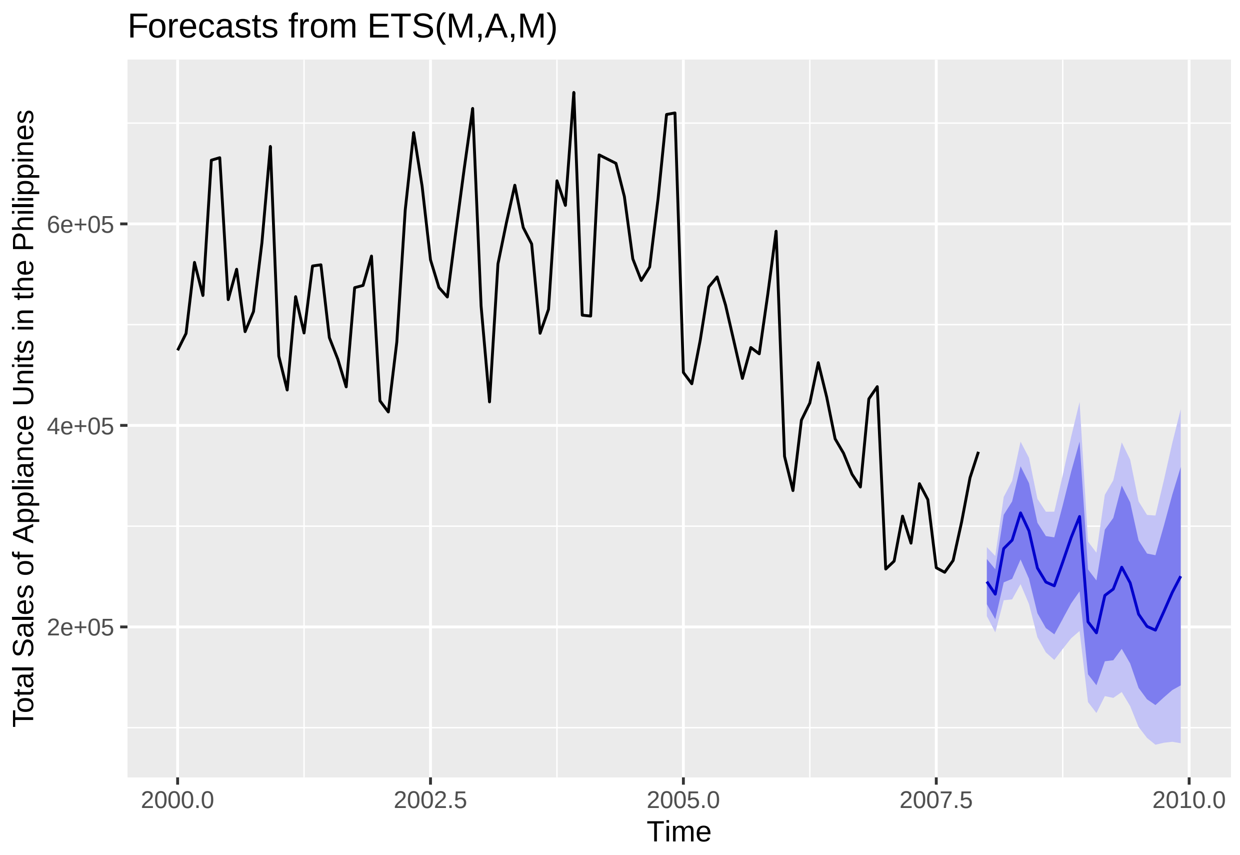
The state space equation for ETS(M,A,M) is given by:

$$y_t = (l_{t-1} + b_{t-1})s_{t-m}(1 + \epsilon_t)$$
$$l_t = (l_{t-1} + b_{t-1})(1 + 0.5734\epsilon_t)$$
$$b_t = b_{t-1} + 0.0139(l_{t-1} + b_{t-1})\epsilon_t$$
$$s_t = s_{t-m}(1 + 2 \times 10^{-4} \times \epsilon_t)$$
$$l_0 = 597464.3559, b_0 = 1084.4734$$
$$s_{initials} = (1.2361, 1.1346, 1.0226, 0.9175, 0.9177, 0.9557, 1.0759, 1.1248, 1.013, 0.9696, 0.8006, 0.832)$$

2) [1pt] Based on the model in (1), show a plot of the forecasted value of sale_app for the test data added into the plot of the full dataset. Analyze the plot in terms of the forecasting performance of the selected model in (1).

Forecasting using the ETS(M,A,M) model

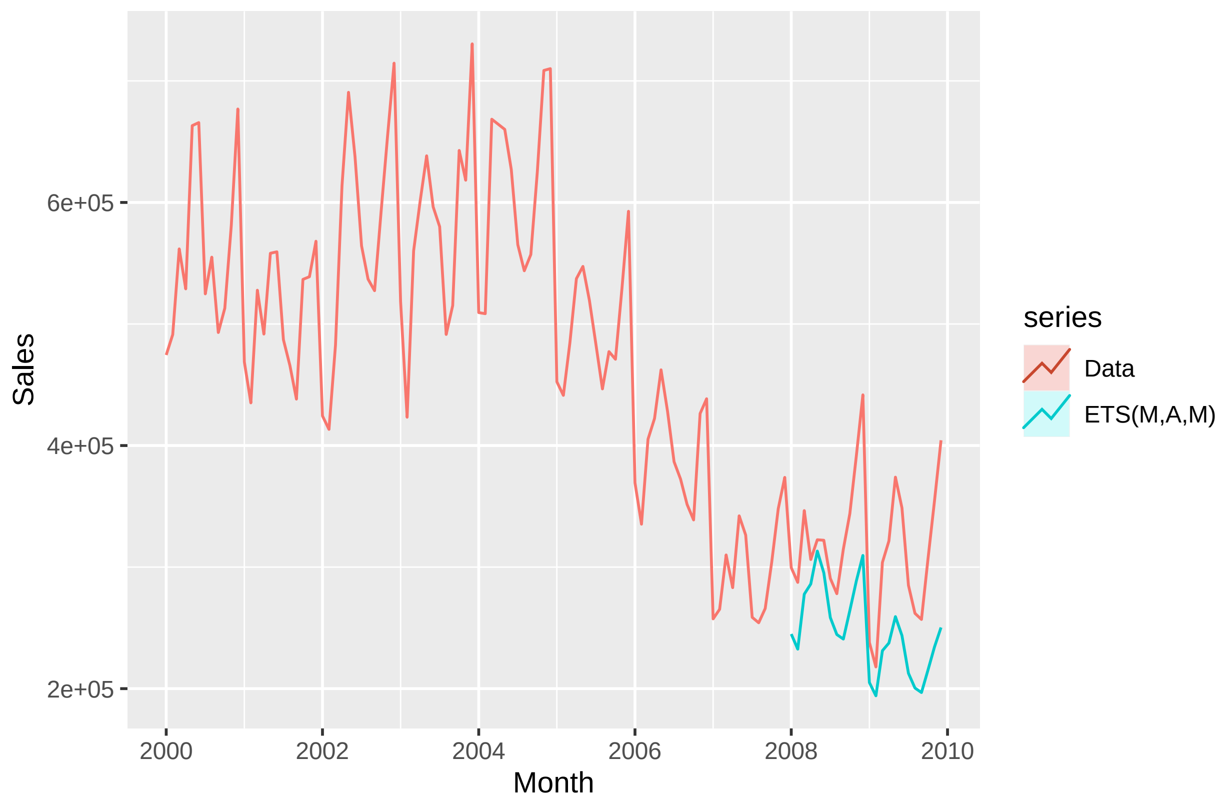
```r
saleapp_ets %>% forecast(h=24) %>%
  autoplot() +
  ylab("Total Sales of Appliance Units in the Philippines")
```



Plotting the forecasts and the actual values together,

```
autoplot(saleapp, series="Data") +
  autolayer(forecast(saleapp_ets, newdata = saleapp_test), series="ETS(M,A,M)", PI=FALSE) +
  xlab("Month") + ylab("Sales") +
  ggtitle("Figure 1. Forecast for Sales of Appliances in Units Sold, Jan 2008-Dec 2009")
```

Figure 1. Forecast for Sales of Appliances in Units Sold, Jan 2008−Dec 2



The ETS (M,A,M) model seems to capture the seasonality reasonably well but forecast values are consistently lower than the actual values from 2008-2009.

3) [1pt] Generate the accuracy measures of the selected model in (1) with respect to the testing dataset. Write a short analysis based on the accuracy measures.

Checking the accuracy for the ETS (M,A,M) model

```
accuracy(forecast(saleapp_ets,h=24), saleapp_test)
```

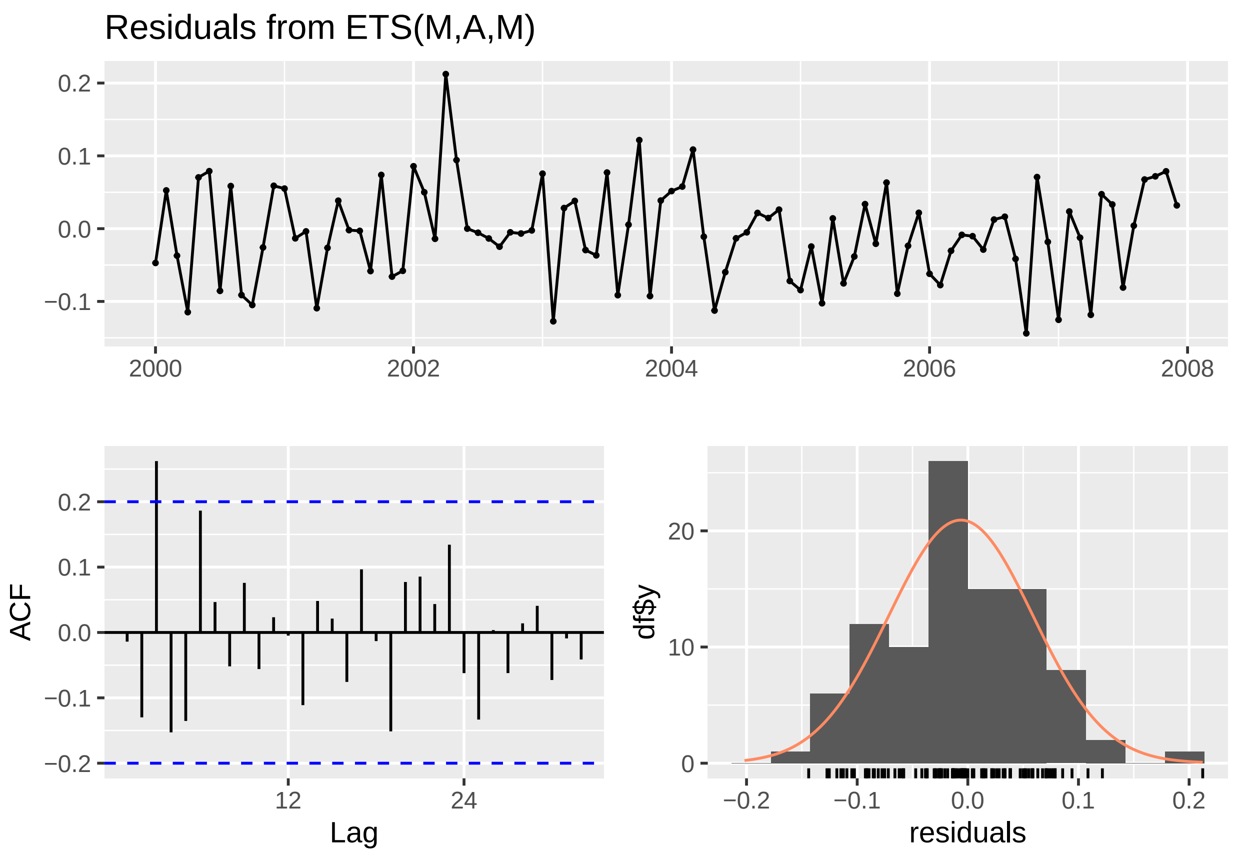
```
##                      ME     RMSE      MAE       MPE      MAPE      MASE
## Training set -3819.613 33707.20 26105.00 -1.060642  5.279745 0.3384195
## Test set     70063.980 79393.65 70063.98 21.273237 21.273237 0.9082941
##                     ACF1 Theil's U
## Training set -0.01535319        NA
## Test set      0.53730178  1.493705
```

The RMSE, MAPE and MAE for the ETS (M,A,M) model are still quite high even if this is the best model with the lowest AICc as recommended by the ets function. This is validated on the plot where the forecast values are consistently lower than the actual values so there is still room for improvement in the ETS(M,A,M) model.

4) [1pt] Check the residuals of the selected model in (1). Has the selected model in (1) comply with the properties that residuals should have for full extraction of the patterns from the time series? Any recommendations?

Testing for the presence of autocorrelation,

```
checkresiduals(saleapp_ets)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,A,M)
## Q* = 24.445, df = 3, p-value = 2.017e-05
##
## Model df: 16.   Total lags used: 19
```

Using a p value of 0.05, there is sufficient evidence to conclude that the residuals are not independently distributed and autocorrelation is present in the data.

Testing for the normality of the residuals,

```
shapiro.test(residuals(saleapp_ets))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(saleapp_ets)
## W = 0.98223, p-value = 0.2199
```

Based on the results of the Shapiro-Wilk normality test, the residuals are normally distributed.

The ETS(M,A,M) does not fully comply with the properties that residuals should have for full extraction of the patterns from the time series. There is sufficient evidence to conclude that autocorrelation is present based on the Ljung-Box test results even if the residuals are normally distributed.

To further improve the performance of the model, the autocorrelation issues must be addressed. This can be done by exploring the addition of other predictor variables in the model or via variable transformation.

II. Use the pce data (Quarterly Personal Consumption Expenditure, in Million Pesos, Q1 1981 – Q4 2008) in PhilQuarterData.csv.

Loading the data:

```
PhilQuarterlyData <- read.csv("PhilQuarterData.csv", stringsAsFactors = FALSE, na.strings = c("NA"))
#View(PhilQuarterlyData)

#Quarterly PCE data

PH_Quarter_pce <- ts(PhilQuarterlyData$pce, start=1981, frequency=4)
PH_Quarter_pce
```

```
##          Qtr1     Qtr2     Qtr3     Qtr4
## 1981  91481.0 100260.0 101908.0 114309.0
## 1982  94727.0 103382.0 104297.0 119661.0
## 1983  96487.0 103859.0 104348.0 119940.0
## 1984  96978.0 104169.0 104453.0 120167.0
## 1985  98973.0 101214.0 101051.0 119594.0
## 1986  99619.0 106344.0 106506.0 122346.0
## 1987 101545.0 111092.0 110414.0 129335.0
## 1988 107828.0 119124.0 119721.0 133889.0
## 1989 113865.0 124365.0 125241.0 141148.0
## 1990 119902.0 130988.0 132046.0 148836.0
## 1991 124334.0 134451.0 134715.0 150288.0
## 1992 129829.0 138703.0 139051.0 153926.0
## 1993 132628.0 143069.0 143802.0 159090.0
## 1994 137450.0 148549.0 148969.0 165138.0
## 1995 143389.0 154434.0 153749.0 171413.0
## 1996 149421.0 161553.0 161074.0 179742.0
## 1997 156862.0 169755.0 169099.0 188600.0
## 1998 163980.0 176450.0 173976.0 193498.0
## 1999 168030.0 181018.0 178325.0 199205.0
## 2000 173407.0 186729.0 184866.0 207064.0
## 2001 179439.0 192885.0 191769.0 214918.0
## 2002 185680.0 200314.0 199932.0 224859.0
## 2003 195033.0 211023.0 210321.0 237221.0
## 2004 206766.3 224318.2 221873.4 250856.5
## 2005 216695.3 235173.0 232159.4 263478.4
## 2006 227928.3 247501.7 244560.8 279737.0
## 2007 241363.4 261401.6 258445.6 296965.8
## 2008 253744.0 272054.0 269923.0 311848.0
```

Split the data into training and test data set:

Training dataset = Q1 1981 – Q4 2005; and Test dataset = Q1 2006 – Q4 2008.

```r
#train dataset
pce_train <- window(PH_Quarter_pce,start=1981,end=c(2005,4))
pce_train
```

```
##          Qtr1     Qtr2     Qtr3     Qtr4
## 1981  91481.0 100260.0 101908.0 114309.0
## 1982  94727.0 103382.0 104297.0 119661.0
## 1983  96487.0 103859.0 104348.0 119940.0
## 1984  96978.0 104169.0 104453.0 120167.0
## 1985  98973.0 101214.0 101051.0 119594.0
## 1986  99619.0 106344.0 106506.0 122346.0
## 1987 101545.0 111092.0 110414.0 129335.0
## 1988 107828.0 119124.0 119721.0 133889.0
## 1989 113865.0 124365.0 125241.0 141148.0
## 1990 119902.0 130988.0 132046.0 148836.0
## 1991 124334.0 134451.0 134715.0 150288.0
## 1992 129829.0 138703.0 139051.0 153926.0
## 1993 132628.0 143069.0 143802.0 159090.0
## 1994 137450.0 148549.0 148969.0 165138.0
## 1995 143389.0 154434.0 153749.0 171413.0
## 1996 149421.0 161553.0 161074.0 179742.0
## 1997 156862.0 169755.0 169099.0 188600.0
## 1998 163980.0 176450.0 173976.0 193498.0
## 1999 168030.0 181018.0 178325.0 199205.0
## 2000 173407.0 186729.0 184866.0 207064.0
## 2001 179439.0 192885.0 191769.0 214918.0
## 2002 185680.0 200314.0 199932.0 224859.0
## 2003 195033.0 211023.0 210321.0 237221.0
## 2004 206766.3 224318.2 221873.4 250856.5
## 2005 216695.3 235173.0 232159.4 263478.4
```

```r
#test dataset
pce_test <- window(PH_Quarter_pce,start=2006,end=c(2008,4))
pce_test
```

```
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2006 227928.3 247501.7 244560.8 279737.0
## 2007 241363.4 261401.6 258445.6 296965.8
## 2008 253744.0 272054.0 269923.0 311848.0
```

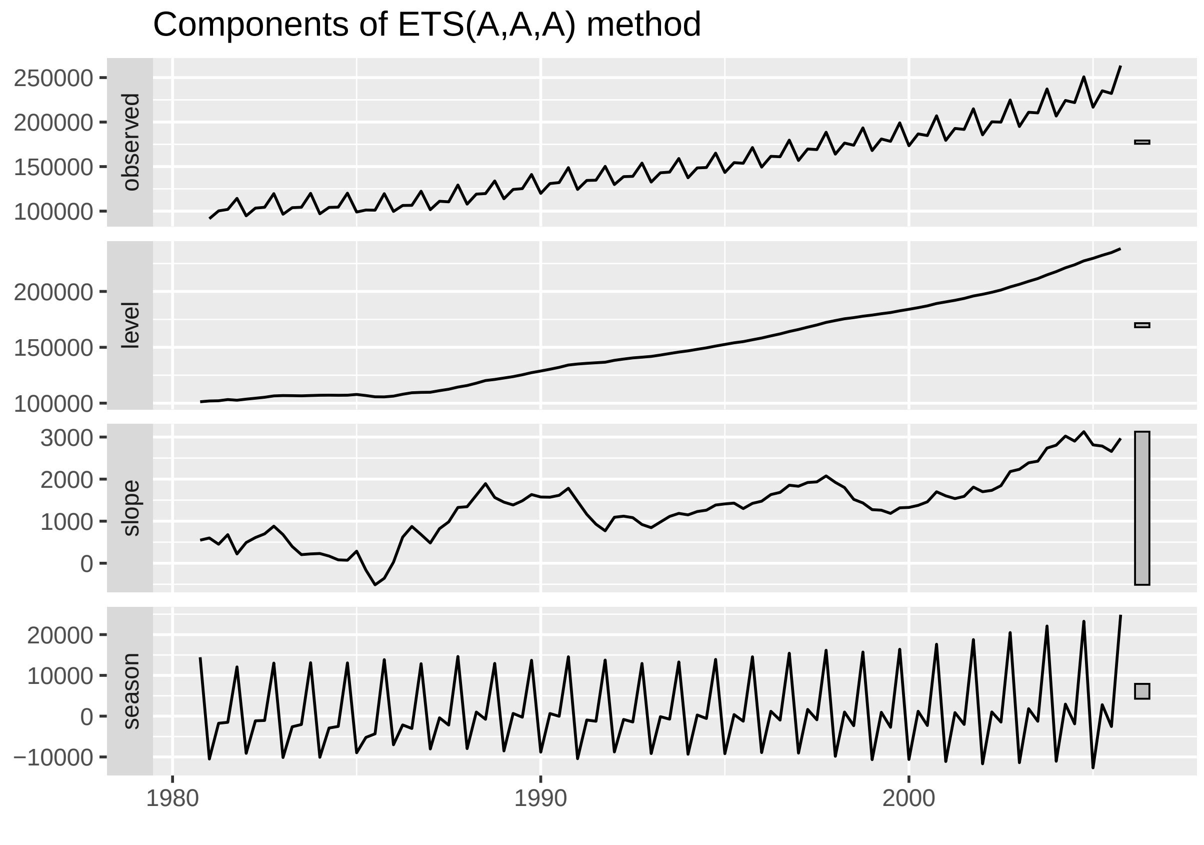
1) [2pts] Using the ets() function, show the best performing model based on the AICc of the training dataset and show the state space system of equations form of the model with estimated parameter values plugged in to the system of equation (for reference, see “Other ETS Models” in https://otexts.com/fpp2/ets.html)

```r
pce_ets <- ets(pce_train, model="ZZZ")
summary(pce_ets)
```

```
## ETS(A,A,A)
##
## Call:
```

```
##  ets(y = pce_train, model = "ZZZ")
##
##   Smoothing parameters:
##     alpha = 0.3154
##     beta  = 0.1138
##     gamma = 0.5858
##
##   Initial states:
##     l = 101255.6841
##     b = 548.7882
##     s = 14442.71 -2683.739 -980.3079 -10778.66
##
##   sigma:  1767.53
##
##      AIC     AICc      BIC
## 1965.647 1967.647 1989.093
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE      MAPE      MASE      ACF1
## Training set 212.7333 1695.355 1343.578 0.1213425 0.9932114 0.2330713 0.1411876
```

```
autoplot(pce_ets)
```



The best performing model based on the ets function is the ETS(A,A,A) model or the Holt-Winters Additive method with additive errors since it has the lowest AICc.

The state space equation for ETS(A,A,A) is given by:

$$y_t = l_{t-1} + b_{t-1} + s_{t-m} + \epsilon_t$$

$$l_t = l_{t-1} + b_{t-1} + 0.3154\epsilon_t$$

$$b_t = b_{t-1} + 0.1138\epsilon_t$$

$$s_t = s_{t-m} + 0.5858\epsilon_t$$

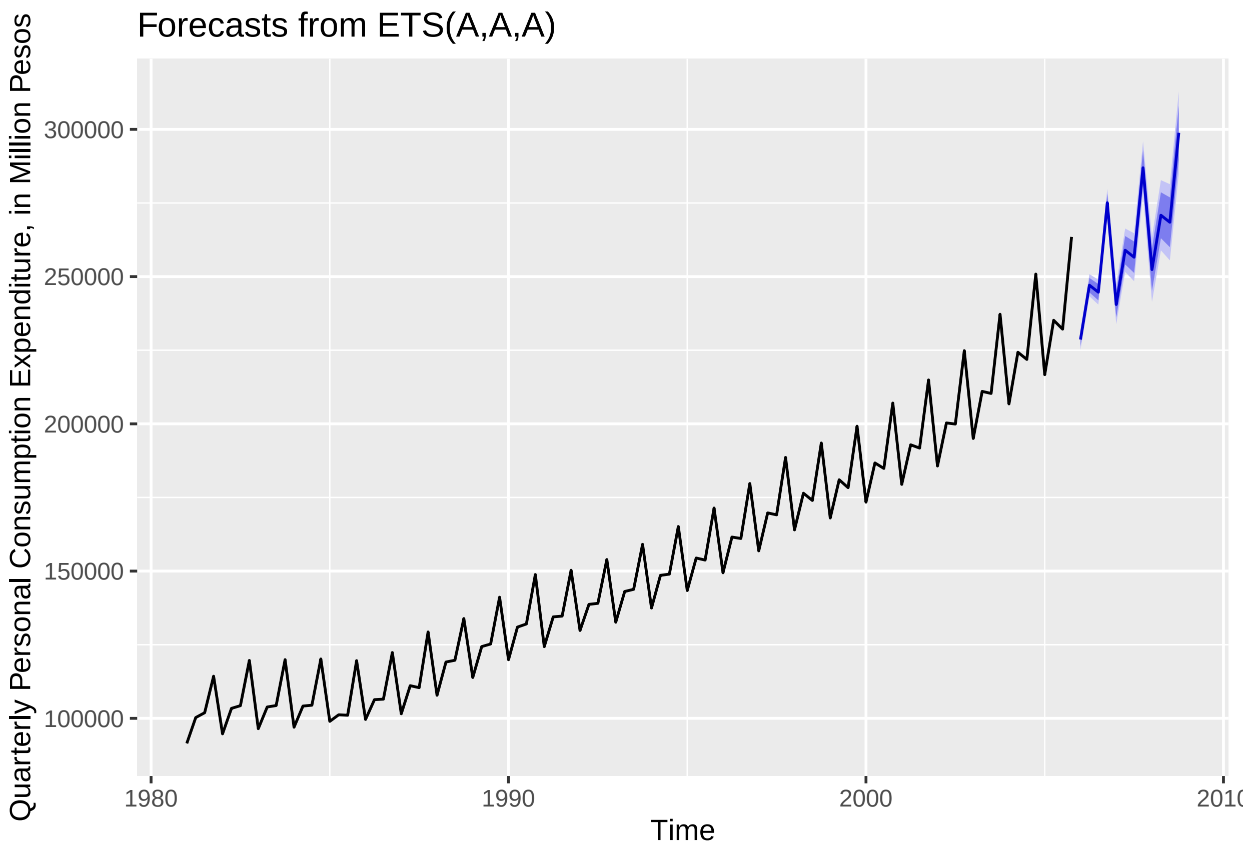
$$l_0 = 101255.6841, b_0 = 548.7882$$

$$s_{initials} = (14442.71, -2683.739, -980.3079, -10778.66)$$

2) [1pt] Based on the model in (1), show a plot of the forecast value of pce for the test data added into the plot of the full dataset. Analyze the plot in terms of the forecasting performance of the selected model in (1)

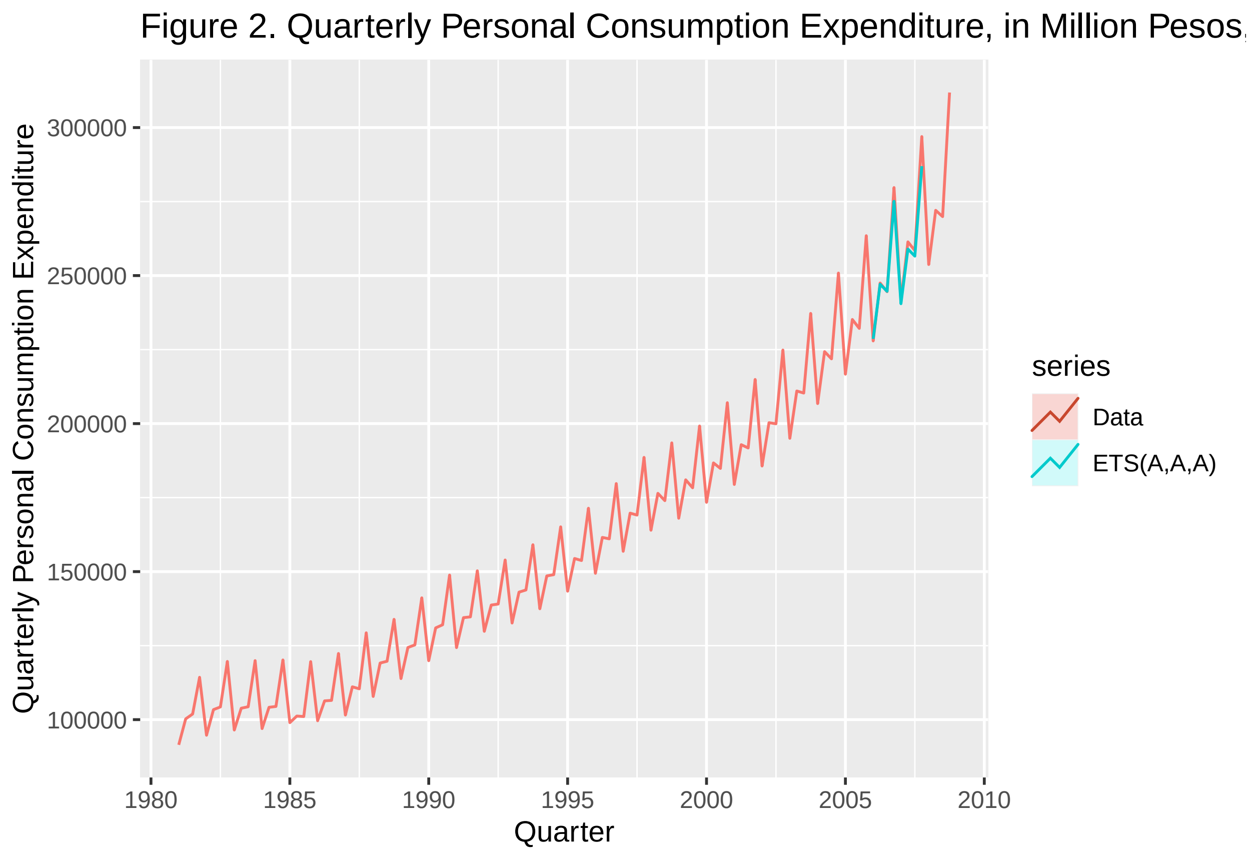
Forecasting using the ETS(A,A,A) model

```r
pce_ets %>% forecast(h=12) %>%
  autoplot() +
  ylab("Quarterly Personal Consumption Expenditure, in Million Pesos")
```



Plotting the forecasts and the actual values together,

```
autoplot(PH_Quarter_pce, series="Data") +
  autolayer(forecast(pce_ets, newdata = pce_test), series="ETS(A,A,A)", PI=FALSE) +
  xlab("Quarter") + ylab("Quarterly Personal Consumption Expenditure") +
  ggtitle("Figure 2. Quarterly Personal Consumption Expenditure, in Million Pesos, Q1 1981 - Q4 2008")
```



Based on the plot of the actual and the forecast values, the ETS(A,A,A) model provides a good forecast for the Quarterly Personal Consumption Expenditure. The seasonality and trend are captured well in the forecast and the actual and forecast values are close to each other.

3) [1pt] Generate the accuracy measures of the selected model in (1) with respect to the testing dataset. Write a short analysis based on the accuracy measures.

Checking the accuracy for the ETS (A,A,A) model

```
accuracy(forecast(pce_ets,h=12), pce_test)
```

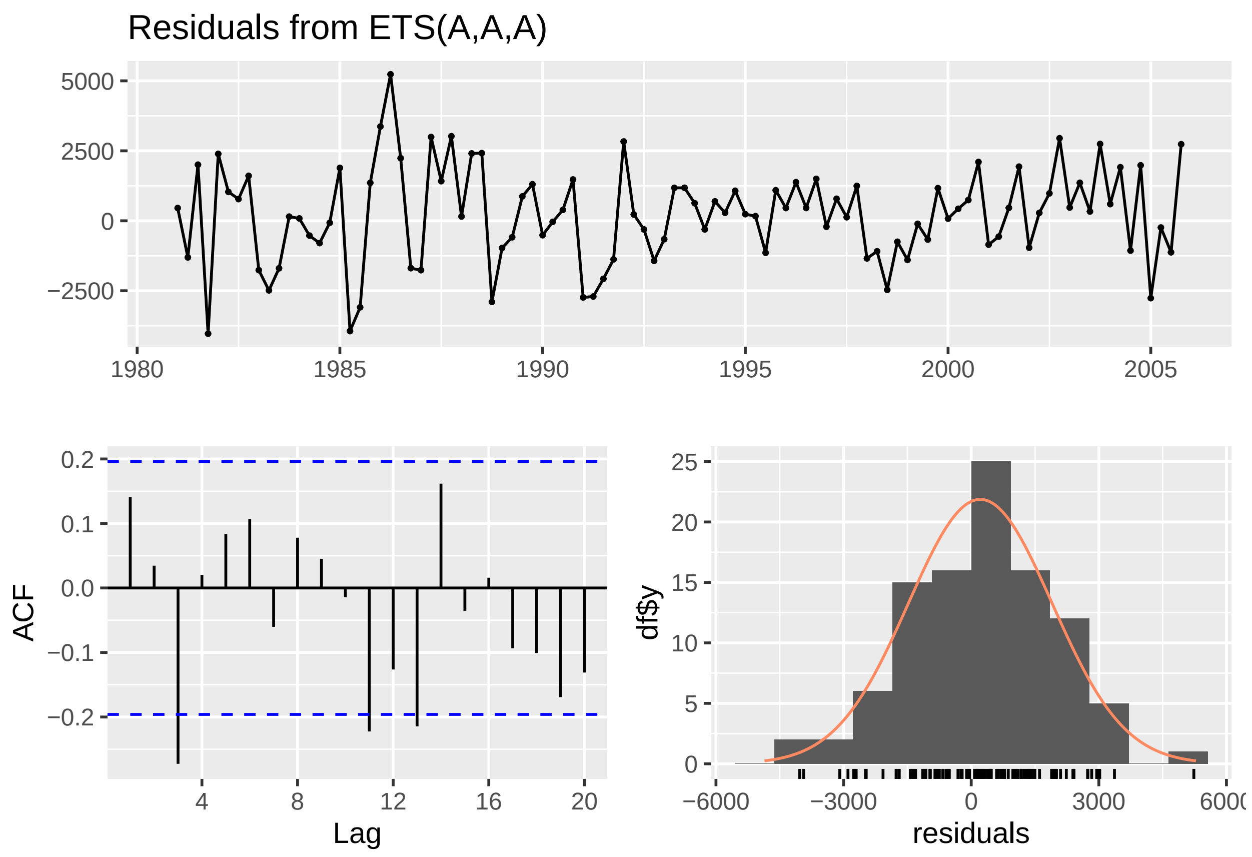
```
##                     ME     RMSE      MAE       MPE      MAPE      MASE
## Training set  212.7333 1695.355 1343.578 0.1213425 0.9932114 0.2330713
## Test set     3039.3750 5061.439 3178.630 1.0461535 1.1065656 0.5513989
##                     ACF1 Theil's U
## Training set  0.14118760        NA
## Test set     -0.09159445 0.1867625
```

The RMSE, MAPE and MAE of the ETS(A,A,A) model is relatively low especially as compared with the ETS(M,A,M) model from item I (sales of appliances in units dataset). This is expected since, as seen on the plot, the forecast values are close to the actual values.

4) [1pt] Check the residuals of the selected model in (1). Has the selected model in (1) comply with the properties that residuals should have for full extraction of the patterns from the time series? Any recommendations?

Testing for the presence of autocorrelation,

```
checkresiduals(pce_ets)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,A,A)
## Q* = 19.018, df = 3, p-value = 0.0002711
##
## Model df: 8.   Total lags used: 11
```

Using a p value of 0.05, there is sufficient evidence to conclude that the residuals are not independently distributed and autocorrelation is present in the data.

Testing for the normality of the residuals,

```
shapiro.test(residuals(pce_ets))
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  residuals(pce_ets)
## W = 0.99229, p-value = 0.842
```

Based on the results of the Shapiro-Wilk normality test, the residuals are normally distributed.

The ETS(A,A,A) does not fully comply with the properties that residuals should have for full extraction of the patterns from the time series even if the forecasted values are close to the actual values as seen in the plot. There is sufficient evidence to conclude that autocorrelation is present based on the Ljung-Box test results even if the residuals are normally distributed.

To further improve the performance of the model, the autocorrelation issues must be addressed. This can be done by exploring the addition of other predictor variables in the model or via variable transformation.