Stat 280 Forecasting Analytics Exercise 1 - Veronica Bayani

Stat 280 Forecasting Analytics Exercise 1 Name: Veronica Bayani Student Number: 2009-00574

Exercise 1 Answer the following problems with codes and their results presented. Expound your answers with analysis and interpretation for each question. Please save your work in a PDF file.

Deadline on 11 January 2023, 11:59pm.

1. [5 pts] Monthly Data (PhilMonthlyData.csv, available at UVLe): Please use from January 2000 to December 2009. Using the ofw_deployed (Number of OFW Deployed, in Persons) series, answer the following questions:

```
#this is to load the data

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 ggplot2   3.4.0     v fma       2.4
## v forecast  8.18      v expsmooth 2.3

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

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

##
```

```
library(tidyverse)
```

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

## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## 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
## v purrr   0.3.4
```

```
## 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(tinytex)
```

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

```
PhilMonthlyData <- read_csv("PhilMonthlyData.csv")
```

```
## Rows: 264 Columns: 37
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (36): cpi, cpifbt, deporate_savings, dubaicrude, expenditures, exports,...
## date  (1): _date_
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
PhilMonthlyData
```

```
## # A tibble: 264 x 37
##    `_date_`     cpi cpifbt depor~1 dubai~2 expen~3 exports fx_rate   gir hotel~4
##    <date>     <dbl>  <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>
##  1 1989-01-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
##  2 1989-02-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
##  3 1989-03-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
##  4 1989-04-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
##  5 1989-05-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
##  6 1989-06-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
##  7 1989-07-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
##  8 1989-08-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
##  9 1989-09-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
## 10 1989-10-01    NA     NA      NA      NA      NA      NA      NA    NA      NA
## # ... with 254 more rows, 27 more variables: imports <dbl>, libor_3m <dbl>,
## #   libor_6m <dbl>, m1_imf <dbl>, m2_imf <dbl>, man_avecaputili <dbl>,
## #   man_valnetsales <dbl>, man_ppi <dbl>, man_vopi <dbl>, mrt3 <dbl>,
## #   ofw_deployed <dbl>, peso_euro <dbl>, peso_sgd <dbl>, peso_yen <dbl>,
## #   psei <dbl>, remit <dbl>, revenues <dbl>, ricep <dbl>, sale_app <dbl>,
## #   sale_automotive <dbl>, sibor_3m <dbl>, sibor_6m <dbl>, tbill182_1 <dbl>,
## #   tbill364_1 <dbl>, tbill91_1 <dbl>, v_arrival <dbl>, wpi <dbl>, and ...
```

Getting the ofw deployed data from 2000 to 2009

```
ofwdeployed <- ts(na.omit(PhilMonthlyData$ofw_deployed),start=c(2000,1),frequency=12)
ofwdeployed
```

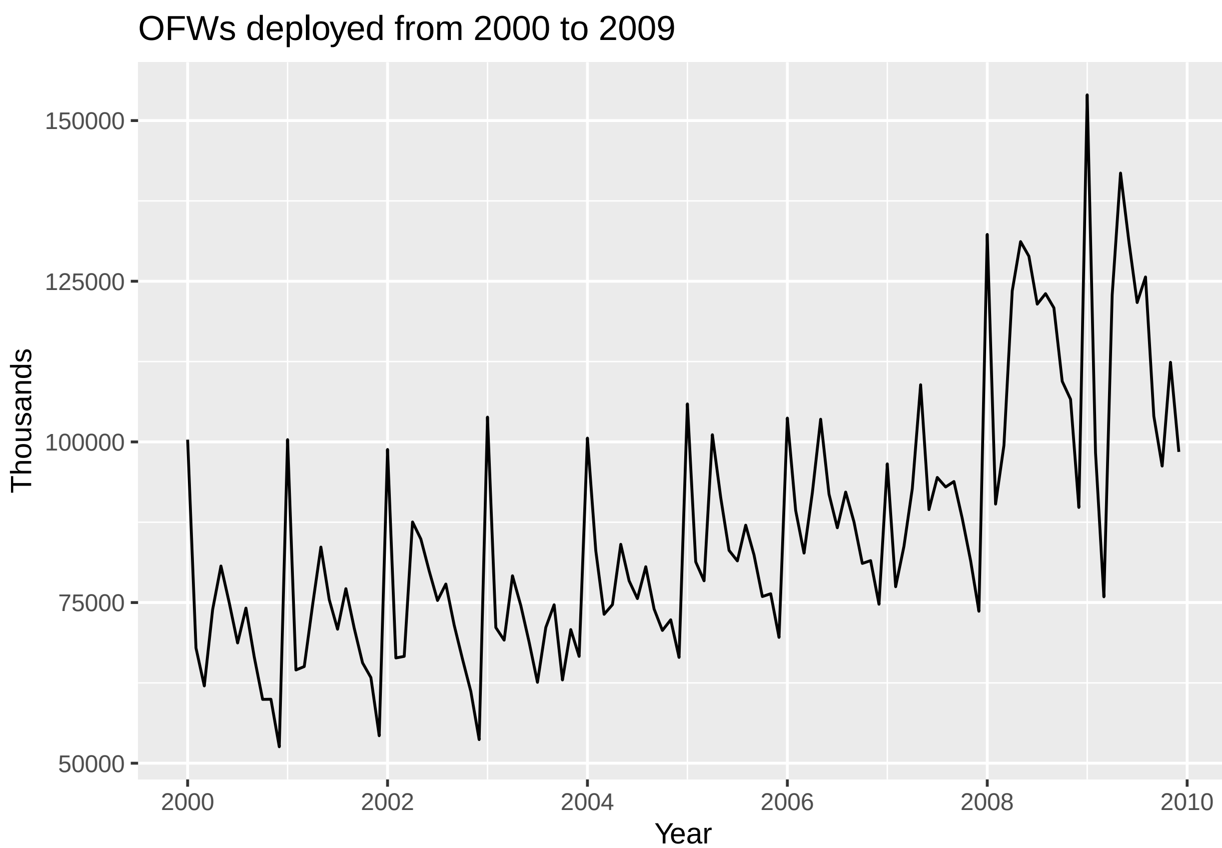
```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2000 100349.00  67935.00  62026.00  73905.00  80691.00  74926.00  68711.00
```

```
## 2001 100349.00  64507.00  65043.00  74619.00  83637.00  75446.00  70849.00
## 2002  98818.00  66380.00  66628.00  87547.00  84878.00  79918.00  75322.00
## 2003 103857.18  71113.00  69151.00  79172.00  74538.00  68822.00  62587.00
## 2004 100597.00  83062.00  73166.00  74674.00  84067.00  78381.00  75615.00
## 2005 105911.00  81334.00  78381.00 101120.00  91337.00  83118.00  81479.63
## 2006 103714.00  89319.00  82694.00  92076.00 103528.00  91872.00  86635.19
## 2007  96584.00  77462.00  83751.00  92723.00 108894.00  89458.00  94472.00
## 2008 132285.00  90323.00  99432.00 123491.00 131171.00 128894.00 121435.00
## 2009 154006.00  98308.00  75892.00 122871.00 141836.00 131235.00 121681.00
##            Aug       Sep       Oct       Nov       Dec
## 2000  74136.00  66509.00  59933.00  59953.00  52554.00
## 2001  77165.00  71007.00  65619.00  63331.00  54275.00
## 2002  77887.00  71482.00  66195.00  61153.00  53673.00
## 2003  71127.00  74655.00  62954.00  70797.00  66618.00
## 2004  80578.00  74007.00  70659.00  72319.00  66463.00
## 2005  87041.43  82394.83  75934.83  76373.43  69579.43
## 2006  92196.99  87550.39  81090.39  81528.99  74734.99
## 2007  92985.00  93836.00  88064.00  81530.00  73643.00
## 2008 123071.00 120860.00 109432.00 106630.00  89799.00
## 2009 125669.00 104007.00  96240.00 112388.00  98445.75
```

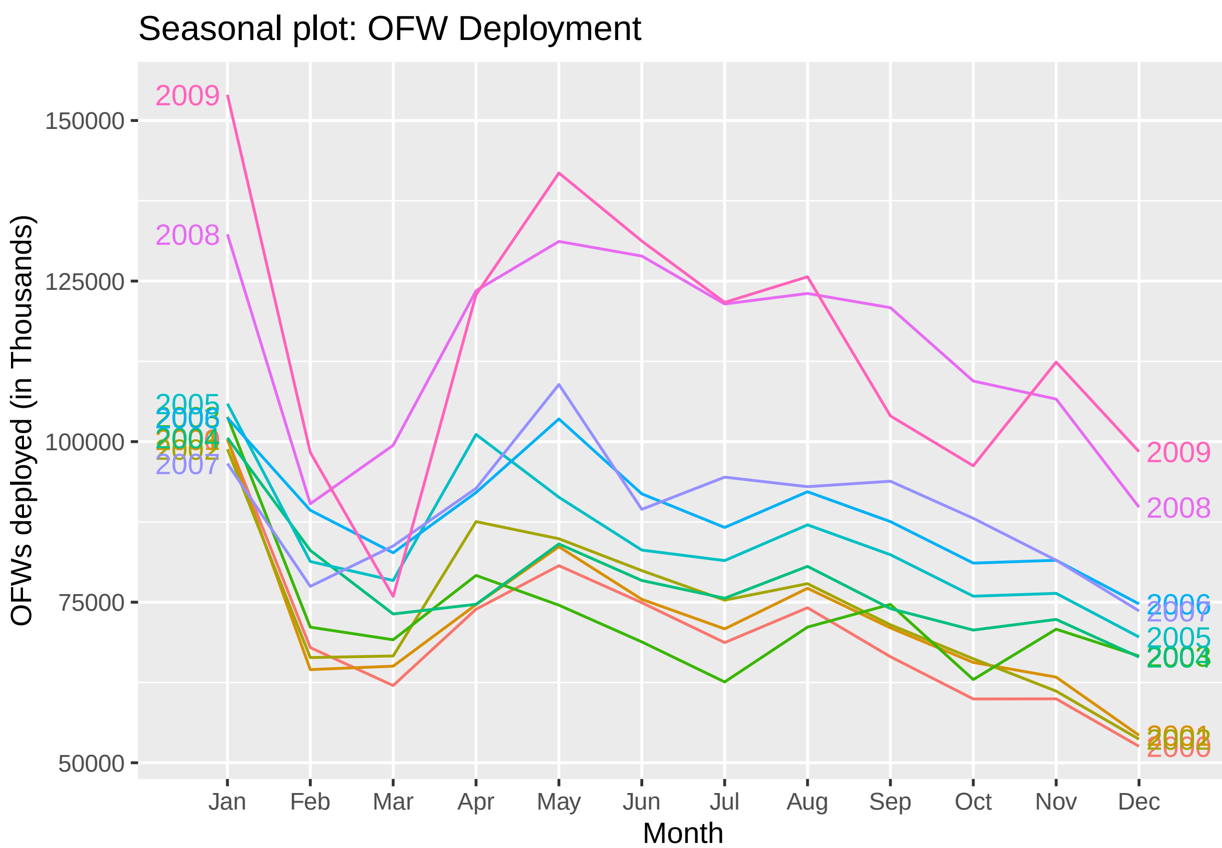
a. [1 pt] Using plots, describe in at least 2 sentences the trend and seasonality of the time series data.

Plotting the ofw deployed data

```
autoplot(ofwdeployed) +
  ggtitle("OFWs deployed from 2000 to 2009") +
  xlab("Year") +
  ylab("Thousands")
```

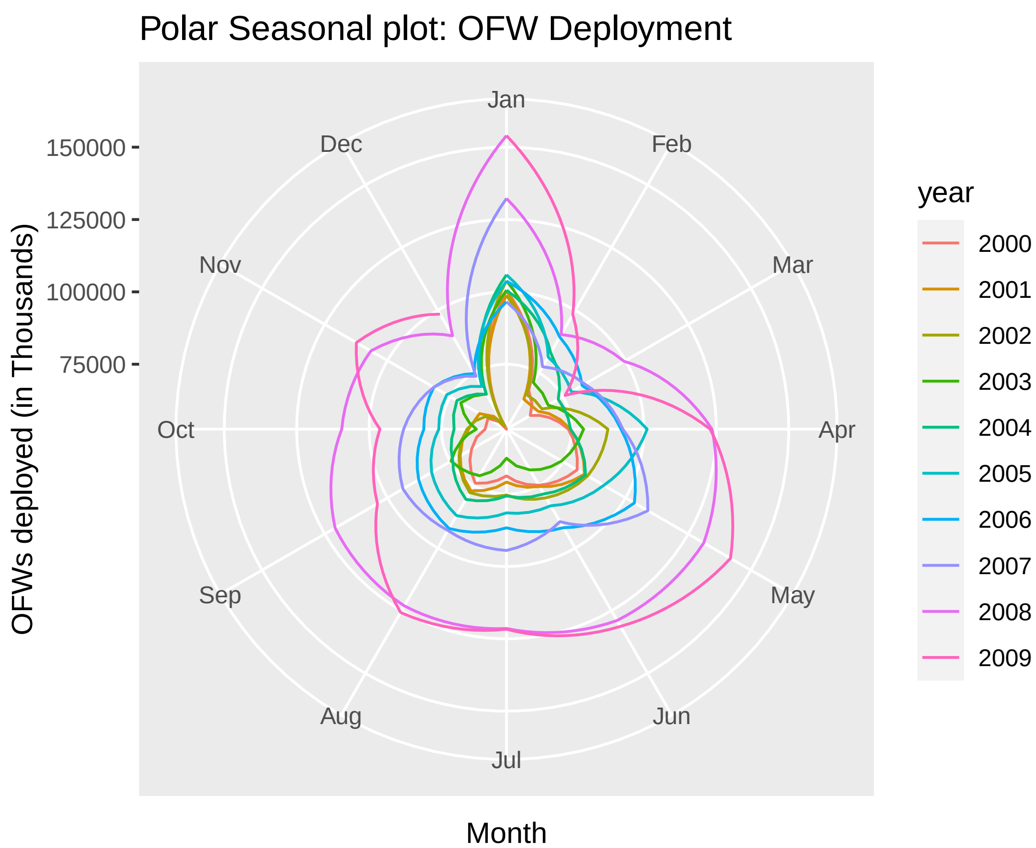



```r
ggseasonplot(ofwdeployed, year.labels=TRUE, year.labels.left=TRUE) +
  ylab("OFWs deployed (in Thousands)") +
  ggtitle("Seasonal plot: OFW Deployment")
```



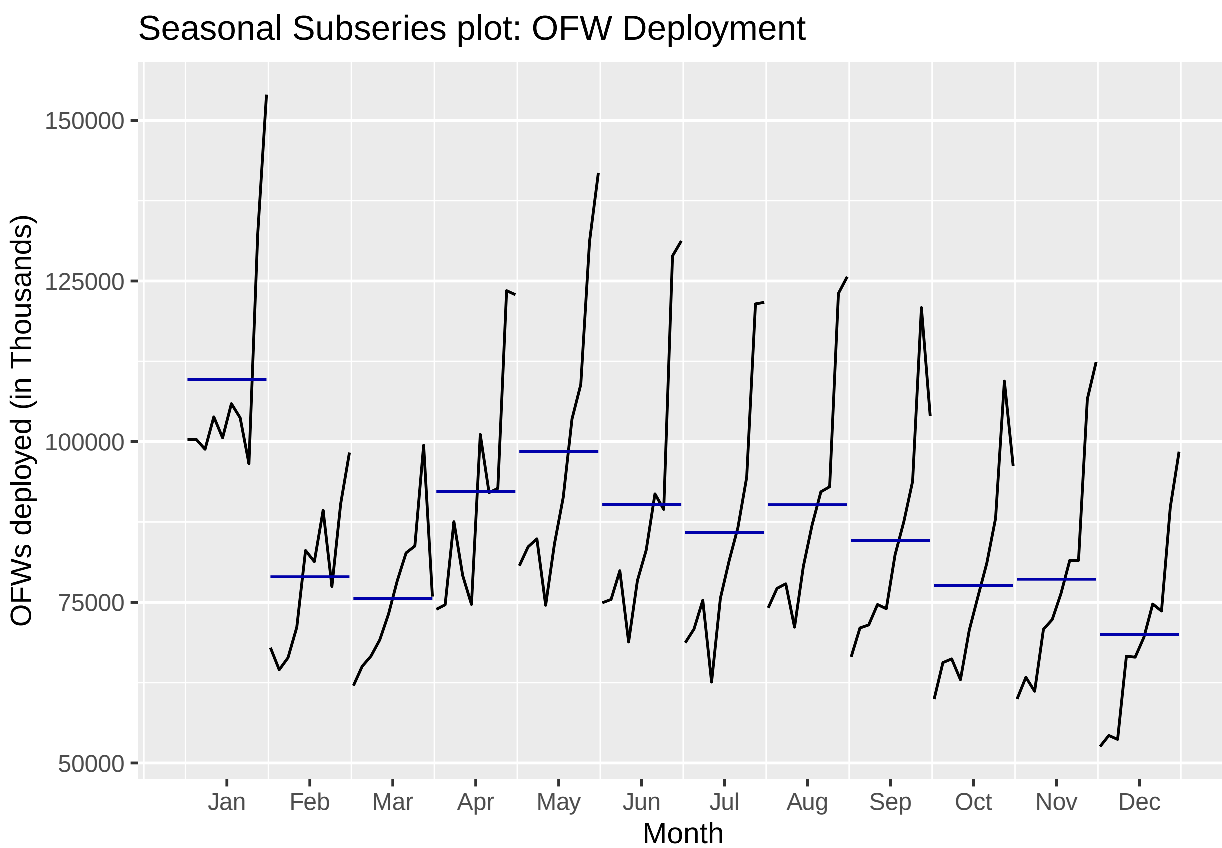
Polar Seasonal Plot

```
ggseasonplot(ofwdeployed, polar = TRUE) +
  ylab("OFWs deployed (in Thousands)") +
  ggtitle("Polar Seasonal plot: OFW Deployment")
```



Seasonal Subseries Plot

```
ggsubseriesplot(ofwdeployed) +
  ylab("OFWs deployed (in Thousands)") +
  ggtitle("Seasonal Subseries plot: OFW Deployment")
```



The number of OFWs deployed from 2000 to 2009 shows a gradually increasing trend with a seasonal pattern. Deployment of OFWs are typically peaks in January then goes down in February and March. It increases from April to May while gradually fluctuating until it reaches the lowest levels in December.

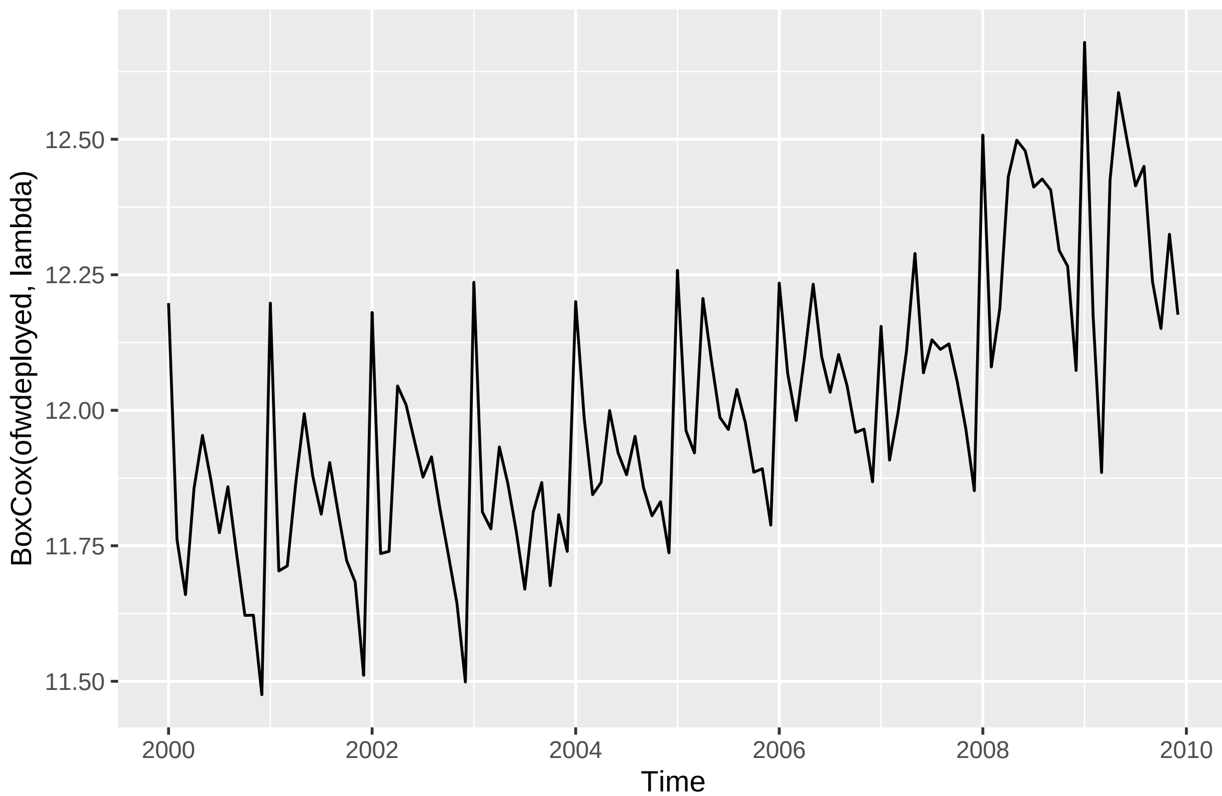
b. [1 pt] What Box-Cox transformation would achieve a stable variance for the data?

Getting the optimal value for the lambda

```
(lambda <- BoxCox.lambda(ofwdeployed))
```

```
## [1] 0.009887901
```

```
autoplot(BoxCox(ofwdeployed,lambda))
```



A lambda value of 0.009887901 in the Box Cox transformation will more or less give a stable variance in the data.

c. [3 pts] Split the data in which the most recent 2 years of data will be the test dataset. Using the forecasting approaches discussed in Chapter 3, which of the methods would best forecast the data? Explain your answer in at least 2 sentences.

Splitting the data where 2000-2007 will be the train data set and 2008 and 2009 will be the test dataset:

```r
#train dataset
ofw_train <- window(ofwdeployed,start=2000,end=c(2007,12))
ofw_train
```

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2000 100349.00  67935.00  62026.00  73905.00  80691.00  74926.00  68711.00
## 2001 100349.00  64507.00  65043.00  74619.00  83637.00  75446.00  70849.00
## 2002  98818.00  66380.00  66628.00  87547.00  84878.00  79918.00  75322.00
## 2003 103857.18  71113.00  69151.00  79172.00  74538.00  68822.00  62587.00
## 2004 100597.00  83062.00  73166.00  74674.00  84067.00  78381.00  75615.00
## 2005 105911.00  81334.00  78381.00 101120.00  91337.00  83118.00  81479.63
## 2006 103714.00  89319.00  82694.00  92076.00 103528.00  91872.00  86635.19
## 2007  96584.00  77462.00  83751.00  92723.00 108894.00  89458.00  94472.00
##            Aug       Sep       Oct       Nov       Dec
## 2000  74136.00  66509.00  59933.00  59953.00  52554.00
## 2001  77165.00  71007.00  65619.00  63331.00  54275.00
```

```
## 2002  77887.00  71482.00  66195.00  61153.00  53673.00
## 2003  71127.00  74655.00  62954.00  70797.00  66618.00
## 2004  80578.00  74007.00  70659.00  72319.00  66463.00
## 2005  87041.43  82394.83  75934.83  76373.43  69579.43
## 2006  92196.99  87550.39  81090.39  81528.99  74734.99
## 2007  92985.00  93836.00  88064.00  81530.00  73643.00
```

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

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2008 132285.00  90323.00  99432.00 123491.00 131171.00 128894.00 121435.00
## 2009 154006.00  98308.00  75892.00 122871.00 141836.00 131235.00 121681.00
##            Aug       Sep       Oct       Nov       Dec
## 2008 123071.00 120860.00 109432.00 106630.00  89799.00
## 2009 125669.00 104007.00  96240.00 112388.00  98445.75
```

Using the average method

```
ofw_average <- meanf(ofw_train, h = 24, level = c(0.8, 0.9, 0.95))
ofw_average
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 90    Hi 90    Lo 95    Hi 95
## Jan 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Feb 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Mar 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Apr 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## May 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Jun 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Jul 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Aug 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Sep 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Oct 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Nov 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Dec 2008        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Jan 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Feb 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Mar 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Apr 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## May 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Jun 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Jul 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Aug 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Sep 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Oct 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Nov 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
## Dec 2009        78757.1 62354.77 95159.43 57645.47 99868.73 53524.96 103989.2
```

Using the Naive method

```r
ofw_naive <- naive(ofw_train, h = 24)
ofw_naive
```

```
##          Point Forecast       Lo 80     Hi 80       Lo 95    Hi 95
## Jan 2008          73643  54214.0809  93071.92  43929.0324 103357.0
## Feb 2008          73643  46166.3591 101119.64  31621.1040 115664.9
## Mar 2008          73643  39991.1249 107294.88  22176.8984 125109.1
## Apr 2008          73643  34785.1618 112500.84  14215.0647 133070.9
## May 2008          73643  30198.6161 117087.38   7200.5485 140085.5
## Jun 2008          73643  26052.0619 121233.94    858.9411 146427.1
## Jul 2008          73643  22238.9118 125047.09  -4972.7688 152258.8
## Aug 2008          73643  18689.7181 128596.28 -10400.7920 157686.8
## Sep 2008          73643  15356.2426 131929.76 -15498.9029 162784.9
## Oct 2008          73643  12203.3631 135082.64 -20320.8160 167606.8
## Nov 2008          73643   9204.5652 138081.43 -24907.0817 172193.1
## Dec 2008          73643   6339.2499 140946.75 -29289.2033 176575.2
## Jan 2009          73643   3591.0359 143694.96 -33492.2339 180778.2
## Feb 2009          73643    946.6412 146339.36 -37536.4865 184822.5
## Mar 2009          73643  -1604.8802 148890.88 -41438.7018 188724.7
## Apr 2009          73643  -4072.6765 151358.68 -45212.8705 192498.9
## May 2009          73643  -6464.4857 153750.49 -48870.8271 196156.8
## Jun 2009          73643  -8786.9228 156072.92 -52422.6881 199708.7
## Jul 2009          73643 -11045.6950 158331.70 -55877.1821 203163.2
## Aug 2009          73643 -13245.7678 160531.77 -59241.9030 206527.9
## Sep 2009          73643 -15391.4926 162677.49 -62523.5059 209809.5
## Oct 2009          73643 -17486.7085 164772.71 -65727.8621 213013.9
## Nov 2009          73643 -19534.8228 166820.82 -68860.1827 216146.2
## Dec 2009          73643 -21538.8762 168824.88 -71925.1179 219211.1
```

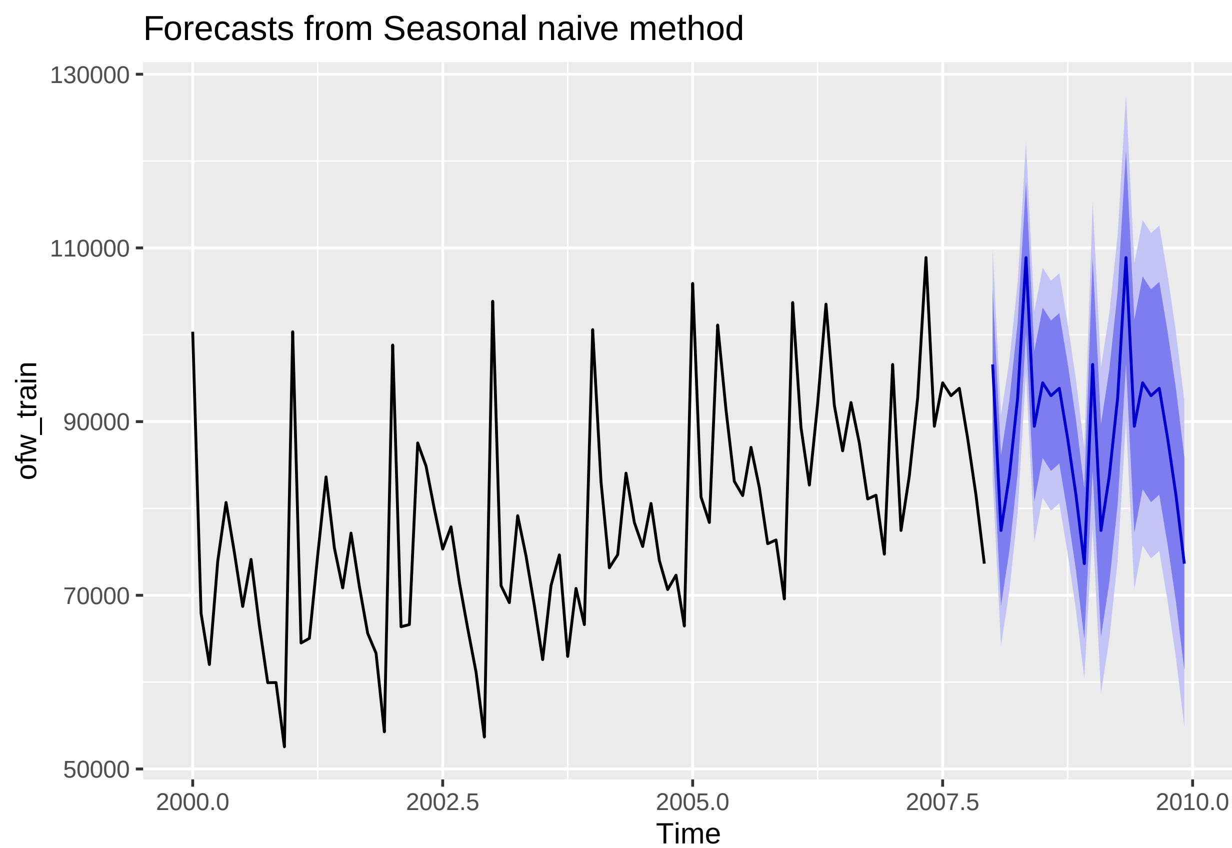
Using the seasonal Naive method

```r
ofw_seasonal_naive <- snaive(ofw_train, h = 24)
ofw_seasonal_naive
```

```
##          Point Forecast     Lo 80     Hi 80    Lo 95     Hi 95
## Jan 2008          96584  87922.53 105245.47 83337.43 109830.57
## Feb 2008          77462  68800.53  86123.47 64215.43  90708.57
## Mar 2008          83751  75089.53  92412.47 70504.43  96997.57
## Apr 2008          92723  84061.53 101384.47 79476.43 105969.57
## May 2008         108894 100232.53 117555.47 95647.43 122140.57
## Jun 2008          89458  80796.53  98119.47 76211.43 102704.57
## Jul 2008          94472  85810.53 103133.47 81225.43 107718.57
## Aug 2008          92985  84323.53 101646.47 79738.43 106231.57
## Sep 2008          93836  85174.53 102497.47 80589.43 107082.57
## Oct 2008          88064  79402.53  96725.47 74817.43 101310.57
## Nov 2008          81530  72868.53  90191.47 68283.43  94776.57
## Dec 2008          73643  64981.53  82304.47 60396.43  86889.57
## Jan 2009          96584  84334.84 108833.16 77850.52 115317.48
## Feb 2009          77462  65212.84  89711.16 58728.52  96195.48
## Mar 2009          83751  71501.84  96000.16 65017.52 102484.48
## Apr 2009          92723  80473.84 104972.16 73989.52 111456.48
## May 2009         108894  96644.84 121143.16 90160.52 127627.48
```

```
## Jun 2009          89458  77208.84 101707.16 70724.52 108191.48
## Jul 2009          94472  82222.84 106721.16 75738.52 113205.48
## Aug 2009          92985  80735.84 105234.16 74251.52 111718.48
## Sep 2009          93836  81586.84 106085.16 75102.52 112569.48
## Oct 2009          88064  75814.84 100313.16 69330.52 106797.48
## Nov 2009          81530  69280.84  93779.16 62796.52 100263.48
## Dec 2009          73643  61393.84  85892.16 54909.52  92376.48
```

```
autoplot(snaive(ofw_train, h = 24))
```



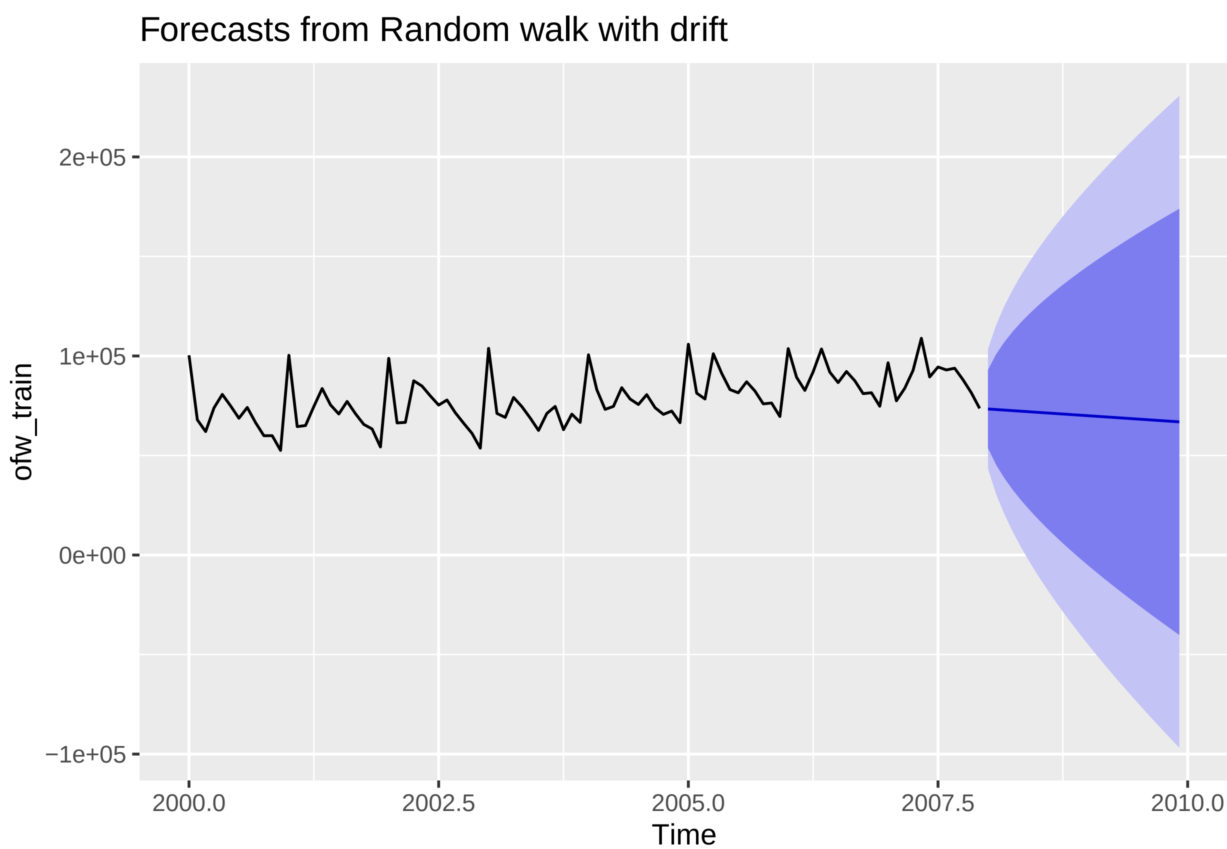
Using the Drift method

```
ofw_drift <- rwf(ofw_train, h = 24, drift = TRUE)
ofw_drift
```

```
##          Point Forecast       Lo 80     Hi 80      Lo 95    Hi 95
## Jan 2008       73361.88  53730.738  92993.03  43338.637 103385.1
## Feb 2008       73080.77  45173.913 100987.62  30400.916 115760.6
## Mar 2008       72799.65  38445.147 107154.16  20258.970 125340.3
## Apr 2008       72518.54  32647.491 112389.58  11541.034 133496.0
## May 2008       72237.42  27435.665 117039.18   3719.047 140755.8
## Jun 2008       71956.31  22633.661 121278.95  -3476.170 147388.8
## Jul 2008       71675.19  18137.559 125212.82 -10203.550 153553.9
## Aug 2008       71394.07  13880.064 128908.08 -16566.013 159354.2
## Sep 2008       71112.96   9814.724 132411.19 -22634.601 164860.5
```

```
## Oct 2008       70831.84   5907.929 135755.76 -28460.715 170124.4
## Nov 2008       70550.73   2134.469 138966.98 -34082.910 175184.4
## Dec 2008       70269.61  -1525.117 142064.34 -39530.950 180070.2
## Jan 2009       69988.49  -5086.203 145063.19 -44828.347 184805.3
## Feb 2009       69707.38  -8561.174 147975.93 -49994.043 189408.8
## Mar 2009       69426.26 -11960.169 150812.70 -55043.544 193896.1
## Apr 2009       69145.15 -15291.605 153581.90 -59989.721 198280.0
## May 2009       68864.03 -18562.553 156290.62 -64843.391 202571.5
## Jun 2009       68582.92 -21779.017 158944.85 -69613.735 206779.6
## Jul 2009       68301.80 -24946.144 161549.74 -74308.624 210912.2
## Aug 2009       68020.68 -28068.378 164109.75 -78934.856 214976.2
## Sep 2009       67739.57 -31149.590 166628.73 -83498.348 218977.5
## Oct 2009       67458.45 -34193.169 169110.07 -88004.287 222921.2
## Nov 2009       67177.34 -37202.103 171556.78 -92457.240 226811.9
## Dec 2009       66896.22 -40179.040 173971.48 -96861.259 230653.7
```

```
autoplot(rwf(ofw_train, h = 24, drift = TRUE))
```



Checking for the accuracy of the different forecasts

```
accuracy(ofw_average, ofw_test)
```

```
##                         ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -5.759462e-12 12578.09 10191.72 -2.575749 13.24343 1.943959
## Test set      3.621797e+04 40477.82 36456.73 29.662748 29.97735 6.953721
```

```
##                    ACF1 Theil's U
## Training set 0.2649739        NA
## Test set     0.0407695  1.467524
```

```
accuracy(ofw_naive, ofw_test)
```

```
##                      ME     RMSE      MAE       MPE     MAPE     MASE
## Training set  -281.1158 15160.47 10503.17 -2.021261 12.91051 2.003364
## Test set     41332.0730 45111.56 41332.07 34.230106 34.23011 7.883639
##                     ACF1 Theil's U
## Training set -0.4254810        NA
## Test set      0.0407695  1.634235
```

```
accuracy(ofw_seasonal_naive, ofw_test)
```

```
##                     ME      RMSE       MAE       MPE      MAPE     MASE
## Training set  2759.214  6758.578  5242.766  3.228857  6.524772 1.000000
## Test set     25524.906 28484.895 26179.823 21.107609 21.970568 4.993513
##                     ACF1 Theil's U
## Training set  0.3063717        NA
## Test set     -0.0721935    1.0501
```

```
accuracy(ofw_drift, ofw_test)
```

```
##                          ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -1.685080e-12 15157.86 10446.95 -1.654222 12.81283 1.992640
## Test set      4.484602e+04 48303.43 44846.02 37.402216 37.40222 8.553885
##                      ACF1 Theil's U
## Training set -0.42548097        NA
## Test set      0.04081535  1.756954
```

Putting the results in a table,

	RMSE	MAE	MAPE	MASE
Mean	40,478	36,457	30	7
Naïve	45,112	41,332	34	8
Seasonal Naïve	28,485	26,180	22	5
Drift	48,303	44,846	37	9

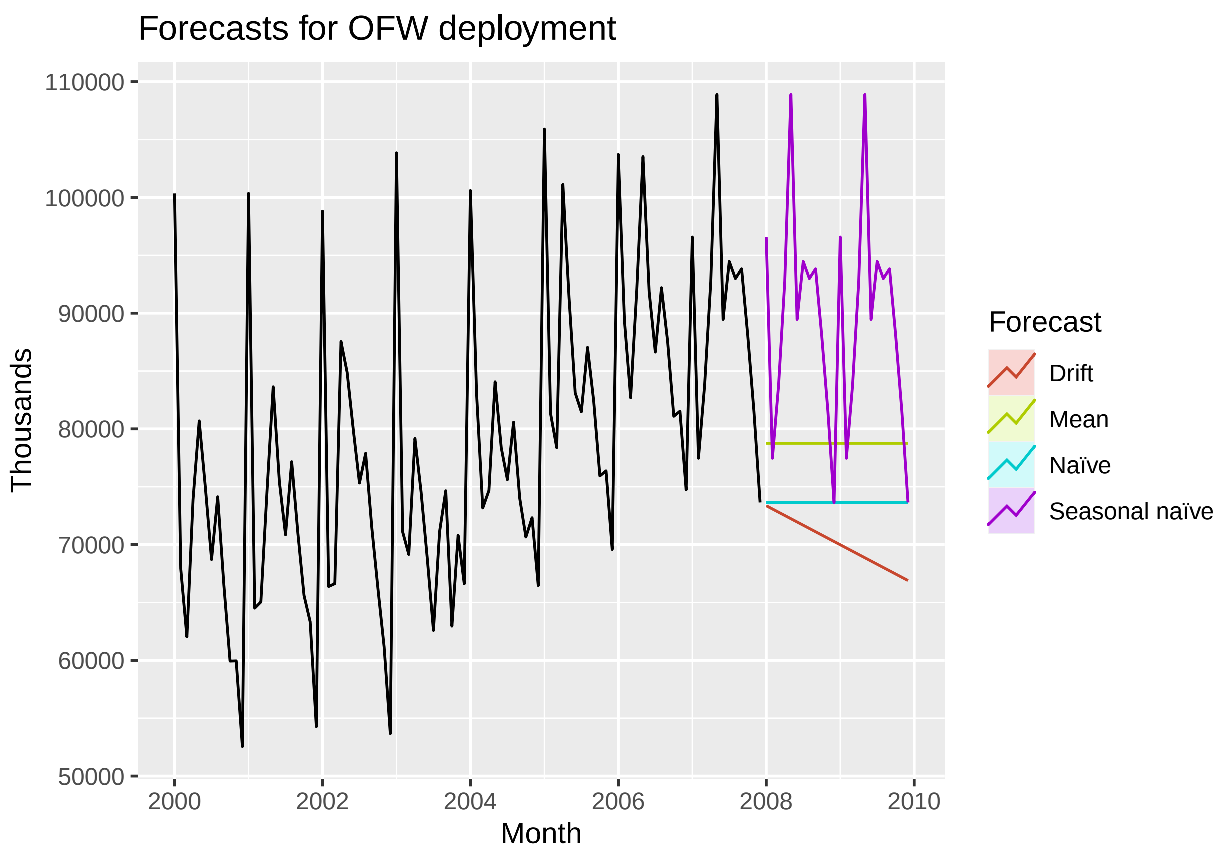
Figure 1: results

The best performing forecast is the Seasonal Naive method because it has the lowest RMSE, MAE, MAPE and MASE. This can be visually validated in the line graph below where the Seasonal Naive FC seems to give the best prediction and to capture the seasonality best as compared to the other forecasts.

Visualizing all of the forecasts in one graph

```
autoplot(ofw_train) +
  autolayer(meanf(ofw_train, h=24),
    series="Mean", PI=FALSE) +
  autolayer(naive(ofw_train, h=24),
```

```
    series="Naïve", PI=FALSE) +
  autolayer(snaive(ofw_train, h=24),
    series="Seasonal naïve", PI=FALSE) +
  autolayer(rwf(ofw_train, h = 24, drift = TRUE),
    series="Drift", PI=FALSE) +
  ggtitle("Forecasts for OFW deployment") +
  xlab("Month") + ylab("Thousands") +
  guides(colour=guide_legend(title="Forecast"))
```



Selecting the best forecast using the lowest RMSE in the time series cross validation,

RMSE for the Mean FC

```
e_average <- tsCV(ofwdeployed,meanf,h=24)

sqrt(mean(e_average^2, na.rm = TRUE))
```

```
## [1] 22370.42
```

RMSE for the Naive FC

```
e_naive <- tsCV(ofwdeployed,naive,h=24)

sqrt(mean(e_naive^2, na.rm = TRUE))
```

```
## [1] 20039.54
```

RMSE for the seasonal Naive FC

```r
e_snaive <- tsCV(ofwdeployed,snaive,h=24)

sqrt(mean(e_snaive^2, na.rm = TRUE))
```

```
## [1] 14498.97
```

RMSE for the drift FC

```r
e_drift <- tsCV(ofwdeployed, rwf, drift=TRUE, h=24)

sqrt(mean(e_drift^2, na.rm=TRUE))
```

```
## [1] 62519.92
```

The forecast with the lowest RMSE using time series cross validation is still the seasonal naive forecast which is consistent with the results of the RMSE, MAE, MAPE and MASE that were previously obtained.

2. [5pts] Quarterly Data (PhilQuarterData.csv, available at UVLe): Please use from Quarter 1 1994 to Quarter 4 2008. Using agri (Gross Value Added of Agriculture, Forestry, and Fisheries in the Philippines, in Million Php), answer for the following questions:

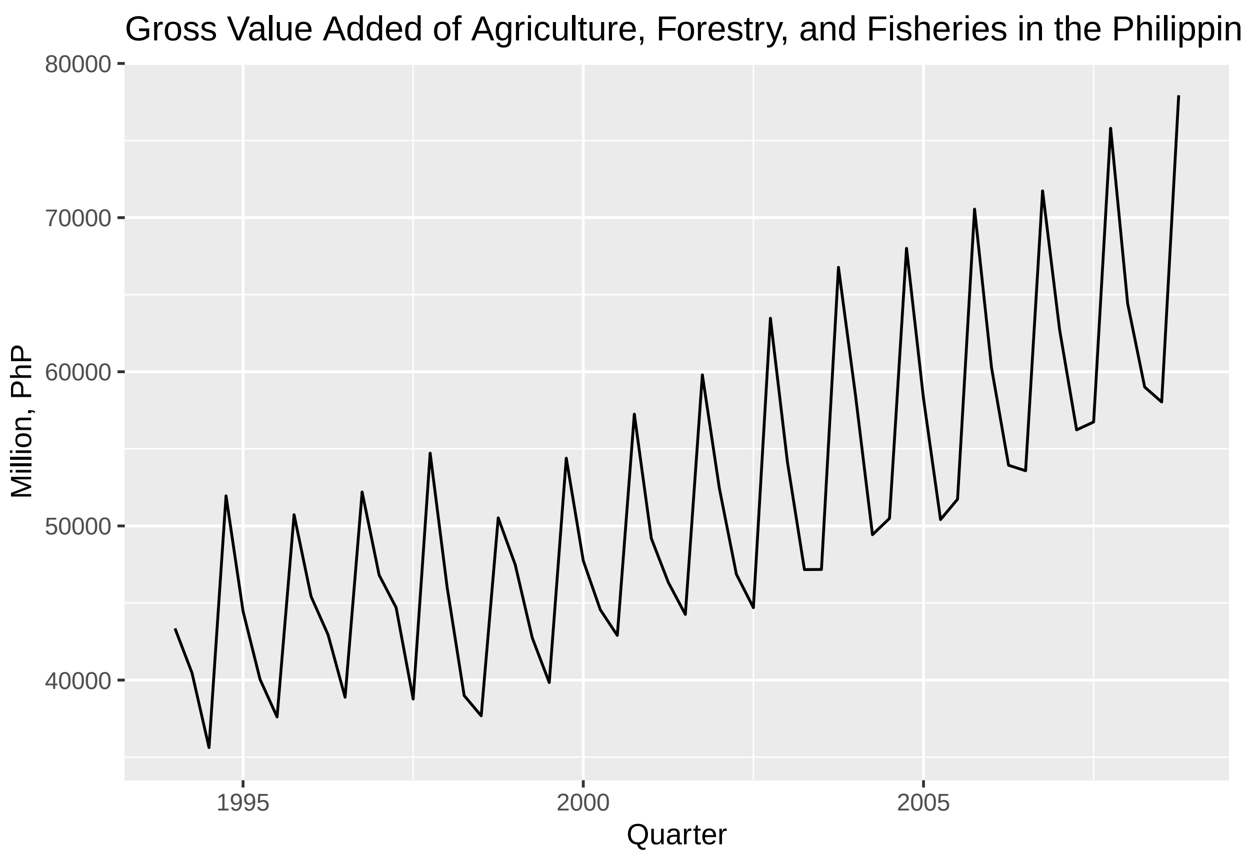
Loading and subsetting the PH quarterly data

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

#Quarterly Agri data

PH_Quarter <- ts(PhilQuarterlyData$agri, start=1981, frequency=4)
# PH_Quarter

quarter_agri <- window(PH_Quarter, start=1994, end=c(2008,4))
quarter_agri
```

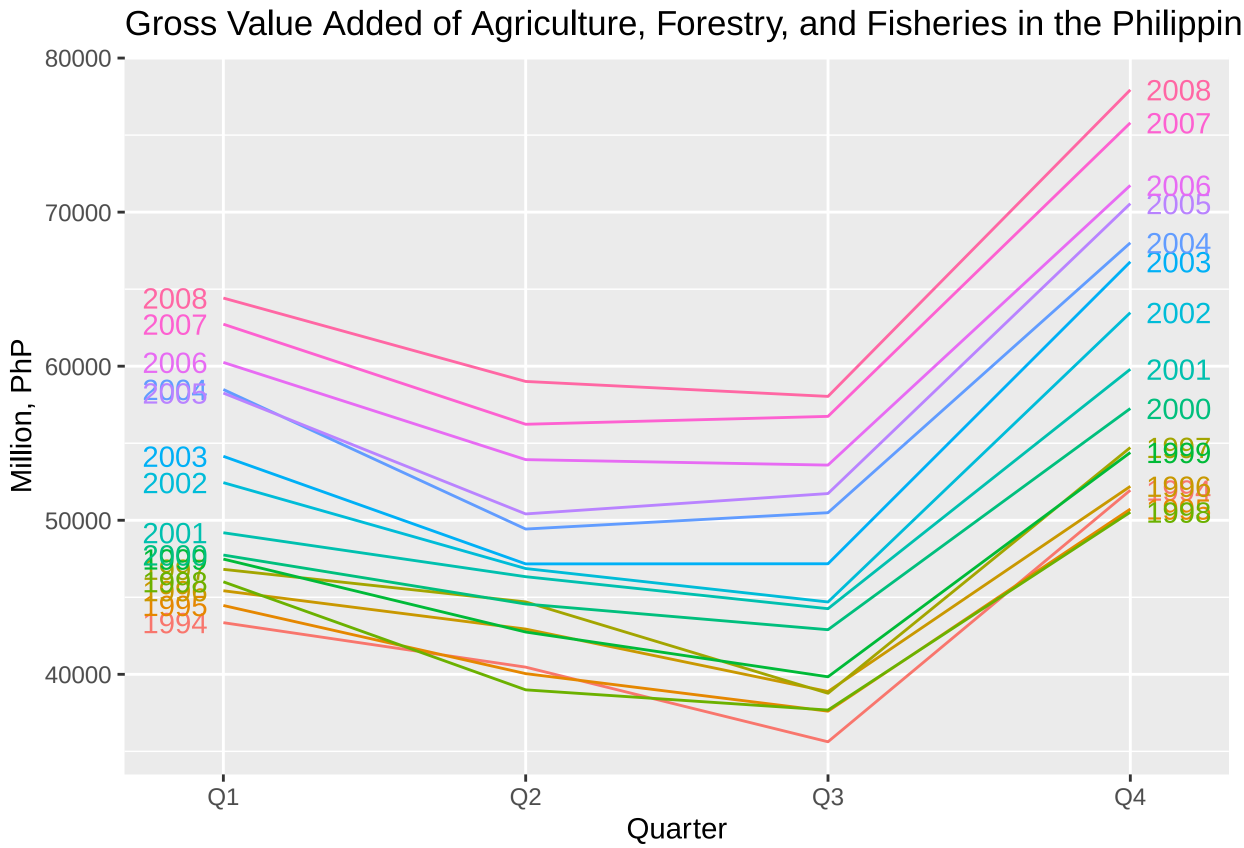
```
##          Qtr1     Qtr2     Qtr3     Qtr4
## 1994 43353.00 40466.00 35620.00 51951.00
## 1995 44467.00 40045.00 37608.00 50728.00
## 1996 45425.00 42938.00 38885.00 52203.00
## 1997 46814.00 44699.00 38769.00 54722.00
## 1998 46004.00 38992.00 37680.00 50525.00
## 1999 47481.00 42746.00 39839.00 54398.00
## 2000 47743.00 44564.00 42896.00 57254.00
## 2001 49190.00 46336.00 44260.00 59803.00
## 2002 52441.63 46866.94 44696.39 63475.28
## 2003 54151.02 47167.58 47176.77 66777.39
## 2004 58488.27 49429.20 50491.39 68008.59
## 2005 58256.02 50411.35 51733.14 70553.86
## 2006 60246.60 53934.64 53580.46 71737.04
## 2007 62726.00 56230.00 56742.00 75797.00
## 2008 64422.00 59010.00 58040.00 77938.00
```

a. [1 pt] Using plots, describe in at least 2 sentences the trend and seasonality of the time series data.

```
autoplot(quarter_agri) +
  ggtitle("Gross Value Added of Agriculture, Forestry, and Fisheries in the Philippines from 1994-2008")
  xlab("Quarter") +
  ylab("Million, PhP")
```

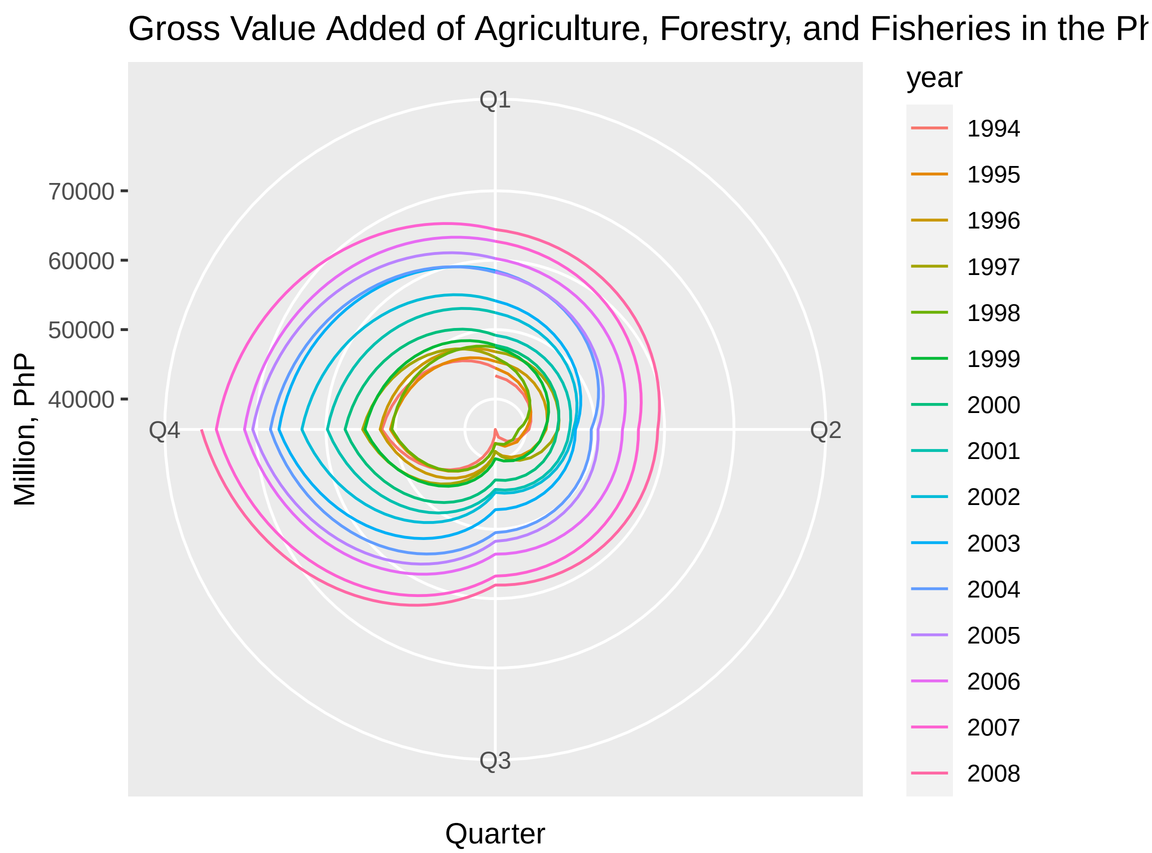



```
ggseasonplot(quarter_agri, year.labels=TRUE, year.labels.left=TRUE) +
  ylab("Million, PhP") +
  ggtitle("Gross Value Added of Agriculture, Forestry, and Fisheries in the Philippines from 1994-2008")
```

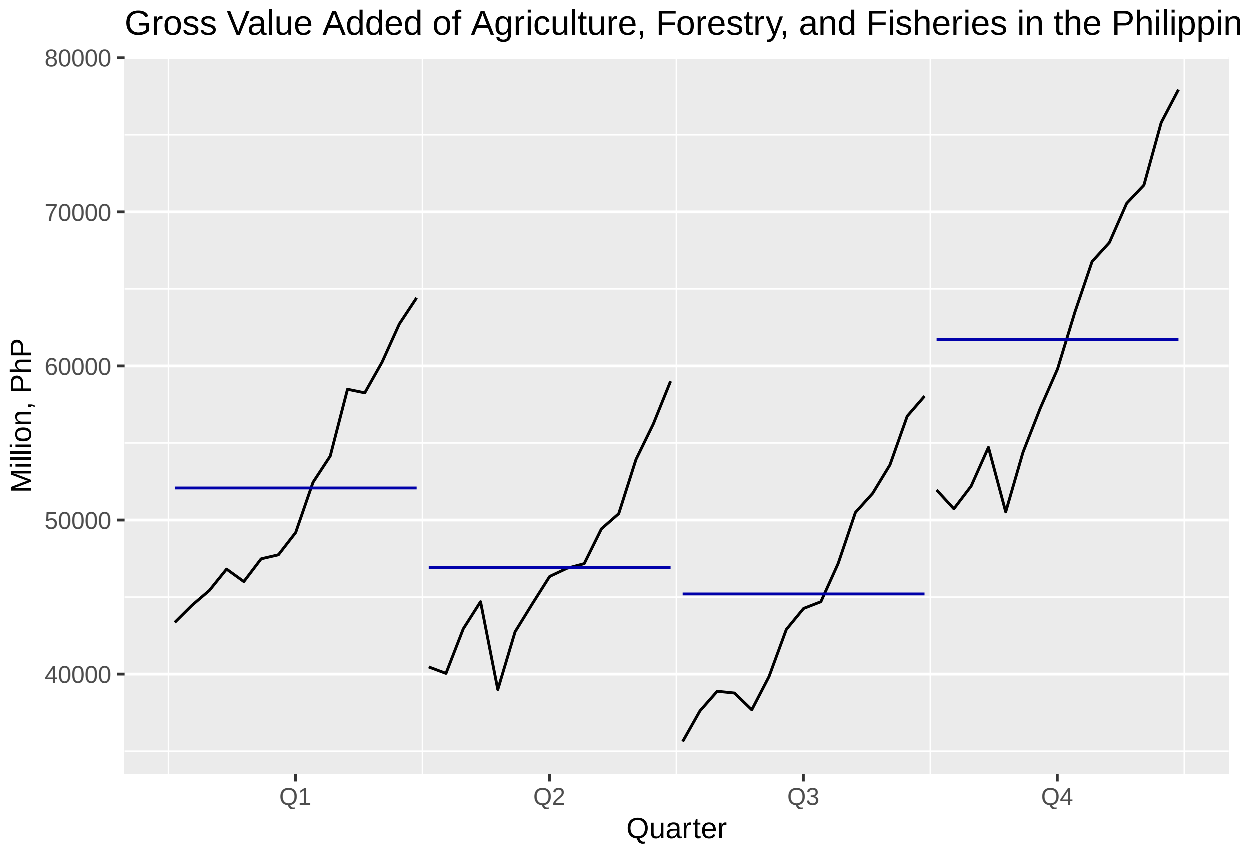


Polar Seasonal Plot

```
ggseasonplot(quarter_agri, polar = TRUE) +
  ylab("Million, PhP") +
  ggtitle("Gross Value Added of Agriculture, Forestry, and Fisheries in the Philippines from 1994-2008")
```




```
ggsubseriesplot(quarter_agri) +
  ylab("Million, PhP") +
  ggtitle("Gross Value Added of Agriculture, Forestry, and Fisheries in the Philippines from 1994-2008")
```



There is an increasing trend in the Gross Value Added of Agriculture, Forestry, and Fisheries in the Philippines from 1994-2008. There is a strong seasonal pattern which is consistent throughout 1994 to 2008 where the production gradually goes down from Q1 to Q3 then there is a sudden increase in Q4.

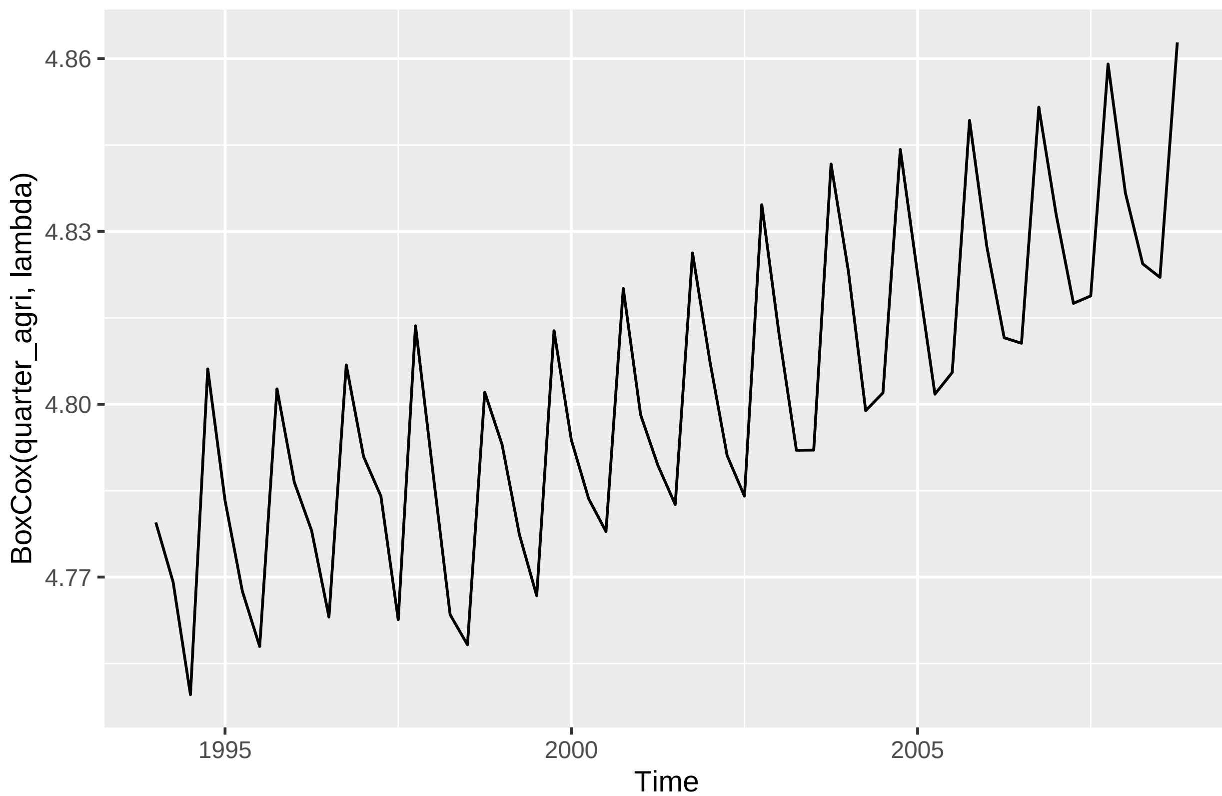
b. [1 pts] What Box-Cox transformation would achieve a stable variance for the data?

Getting the optimal value for the lambda

```
(lambda <- BoxCox.lambda(quarter_agri))
```

```
## [1] -0.1779247
```

```
autoplot(BoxCox(quarter_agri,lambda))
```



A lambda value of -0.1779247 in the Box Cox transformation will more or less give a stable variance in the data.

c. [3 pts] Split the data in which the most recent 4 years of data will be the test dataset. Using the forecasting approaches discussed in Chapter 3, which of the methods would best forecast the data? Explain your answer in at least 2 sentences.

Splitting the data where 1994-2004 will be the train dataset and 2005-2008 will be the test dataset.

```
#train dataset
agri_train <- window(quarter_agri,start=1994,end=c(2004,4))
agri_train
```

```
##          Qtr1     Qtr2     Qtr3     Qtr4
## 1994 43353.00 40466.00 35620.00 51951.00
## 1995 44467.00 40045.00 37608.00 50728.00
## 1996 45425.00 42938.00 38885.00 52203.00
## 1997 46814.00 44699.00 38769.00 54722.00
## 1998 46004.00 38992.00 37680.00 50525.00
## 1999 47481.00 42746.00 39839.00 54398.00
## 2000 47743.00 44564.00 42896.00 57254.00
## 2001 49190.00 46336.00 44260.00 59803.00
## 2002 52441.63 46866.94 44696.39 63475.28
## 2003 54151.02 47167.58 47176.77 66777.39
## 2004 58488.27 49429.20 50491.39 68008.59
```

```r
#test dataset
agri_test <- window(quarter_agri,start=2005,end=c(2008,4))
agri_test
```

```
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2005 58256.02 50411.35 51733.14 70553.86
## 2006 60246.60 53934.64 53580.46 71737.04
## 2007 62726.00 56230.00 56742.00 75797.00
## 2008 64422.00 59010.00 58040.00 77938.00
```

Using the average method

```r
agri_average <- meanf(agri_train, h = 16, level = c(0.8, 0.9, 0.95))
agri_average
```

```
##         Point Forecast    Lo 80    Hi 80    Lo 90    Hi 90    Lo 95    Hi 95
## 2005 Q1       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2005 Q2       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2005 Q3       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2005 Q4       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2006 Q1       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2006 Q2       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2006 Q3       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2006 Q4       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2007 Q1       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2007 Q2       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2007 Q3       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2007 Q4       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2008 Q1       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2008 Q2       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2008 Q3       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
## 2008 Q4       47899.42 37843.09 57955.75 34910.76 60888.08 32317.61 63481.23
```

Using the Naive method

```r
agri_naive <- naive(agri_train, h = 16)
agri_naive
```

```
##         Point Forecast    Lo 80     Hi 80      Lo 95     Hi 95
## 2005 Q1       68008.59 56199.57  79817.61 49948.2586  86068.93
## 2005 Q2       68008.59 51308.12  84709.07 42467.4234  93549.76
## 2005 Q3       68008.59 47554.77  88462.41 36727.1766  99290.01
## 2005 Q4       68008.59 44390.56  91626.63 31887.9247 104129.26
## 2006 Q1       68008.59 41602.83  94414.36 27624.4582 108392.73
## 2006 Q2       68008.59 39082.53  96934.66 23769.9899 112247.20
## 2006 Q3       68008.59 36764.87  99252.32 20225.4404 115791.74
## 2006 Q4       68008.59 34607.65 101409.54 16926.2542 119090.93
## 2007 Q1       68008.59 32581.54 103435.65 13827.5908 122189.59
## 2007 Q2       68008.59 30665.20 105351.98 10896.8021 125120.38
## 2007 Q3       68008.59 28842.51 107174.67  8109.2414 127907.94
## 2007 Q4       68008.59 27100.96 108916.23  5445.7607 130571.42
```

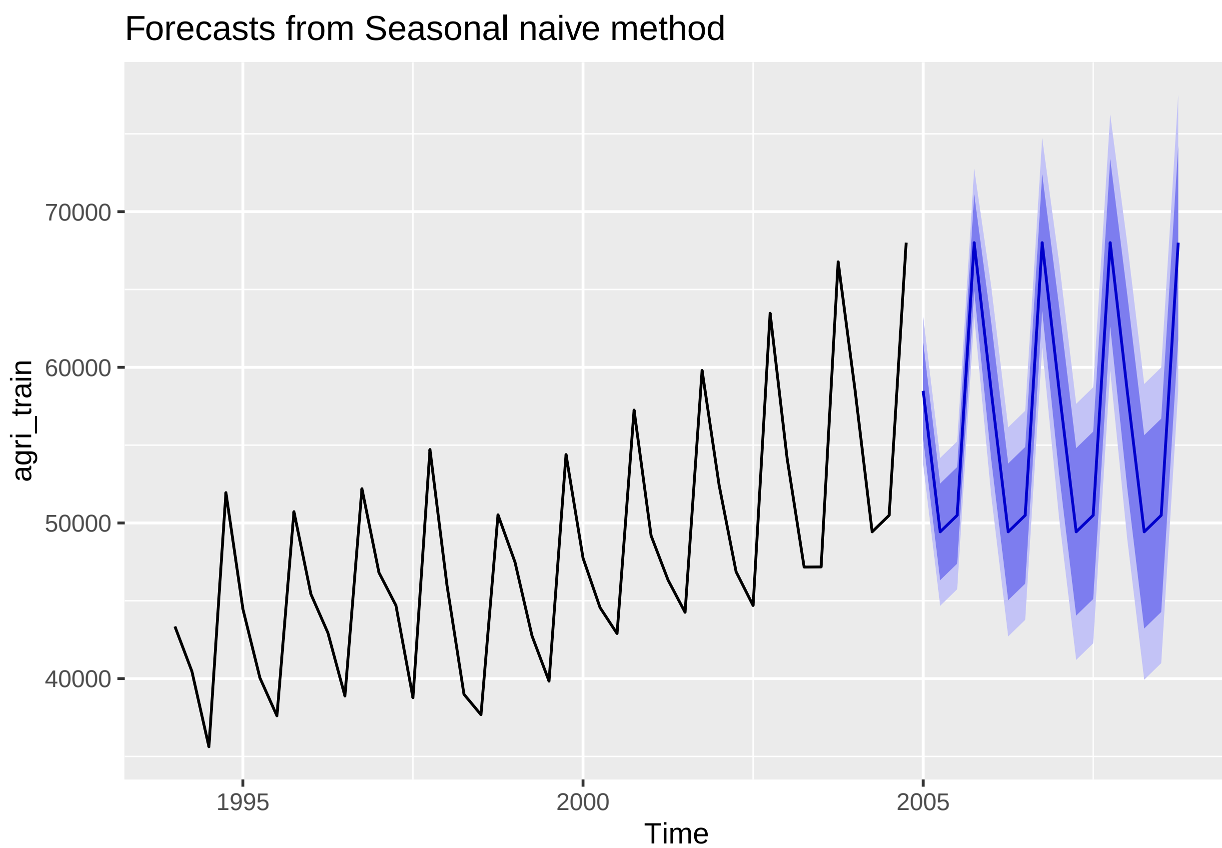
```
## 2008 Q1       68008.59 25430.57 110586.61  2891.1326 133126.05
## 2008 Q2       68008.59 23823.29 112193.89   433.0108 135584.17
## 2008 Q3       68008.59 22272.46 113744.72 -1938.7799 137955.96
## 2008 Q4       68008.59 20772.52 115244.66 -4232.7431 140249.93
```

Using the seasonal Naive method

```
agri_seasonal_naive <- snaive(agri_train, h = 16)
agri_seasonal_naive
```

```
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2005 Q1       58488.27 55383.26 61593.27 53739.58 63236.96
## 2005 Q2       49429.20 46324.20 52534.21 44680.51 54177.89
## 2005 Q3       50491.39 47386.38 53596.39 45742.69 55240.08
## 2005 Q4       68008.59 64903.59 71113.60 63259.90 72757.28
## 2006 Q1       58488.27 54097.13 62879.41 51772.60 65203.93
## 2006 Q2       49429.20 45038.07 53820.34 42713.54 56144.87
## 2006 Q3       50491.39 46100.25 54882.52 43775.72 57207.05
## 2006 Q4       68008.59 63617.46 72399.73 61292.93 74724.26
## 2007 Q1       58488.27 53110.25 63866.29 50263.29 66713.24
## 2007 Q2       49429.20 44051.18 54807.23 41204.23 57654.18
## 2007 Q3       50491.39 45113.36 55869.41 42266.41 58716.36
## 2007 Q4       68008.59 62630.57 73386.62 59783.62 76233.57
## 2008 Q1       58488.27 52278.26 64698.27 48990.88 67985.65
## 2008 Q2       49429.20 43219.20 55639.21 39931.82 58926.59
## 2008 Q3       50491.39 44281.38 56701.39 40994.00 59988.77
## 2008 Q4       68008.59 61798.59 74218.60 58511.21 77505.98
```

```
autoplot(snaive(agri_train, h = 16))
```

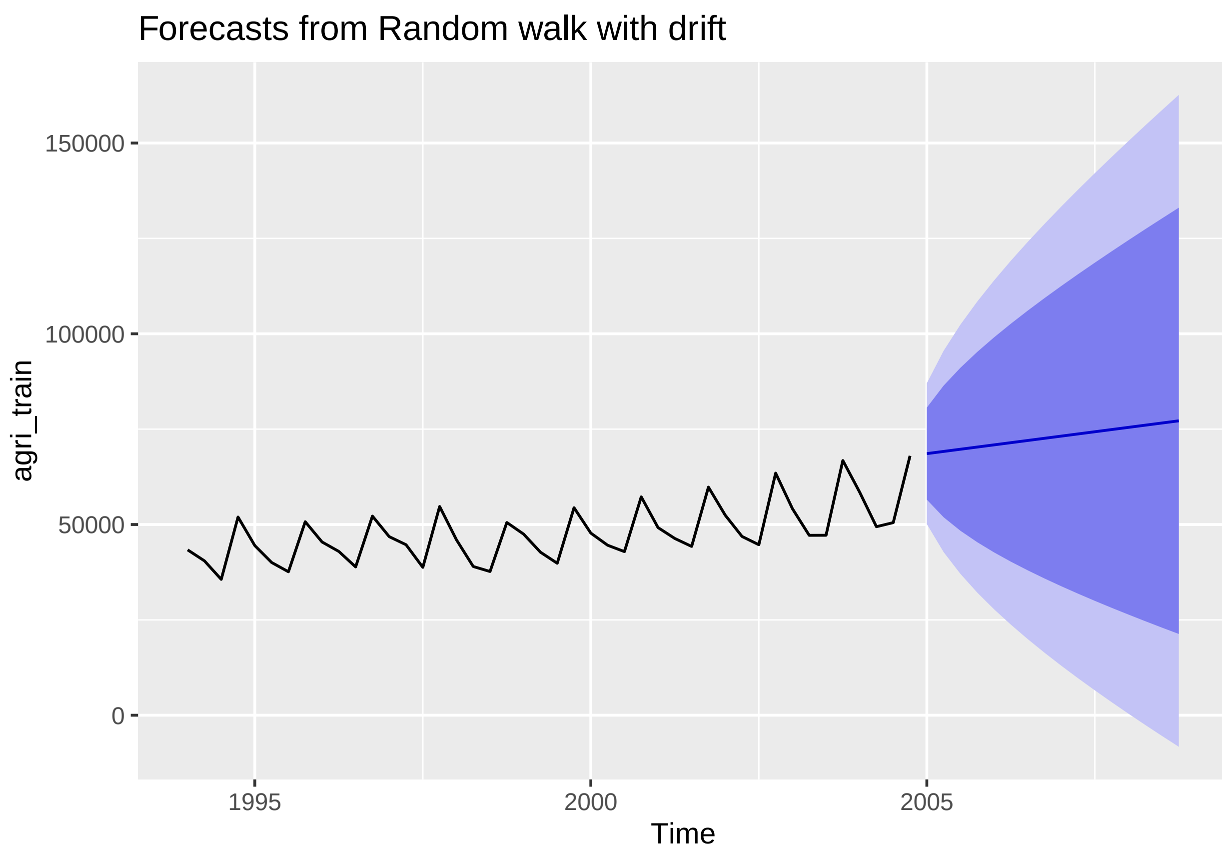


Using the Drift method

```
agri_drift <- rwf(agri_train, h = 16, drift = TRUE)
agri_drift
```

```
##         Point Forecast     Lo 80     Hi 80      Lo 95     Hi 95
## 2005 Q1       68581.98 56518.49  80645.47 50132.4599  87031.50
## 2005 Q2       69155.36 51902.23  86408.50 42768.9758  95541.75
## 2005 Q3       69728.75 48364.57  91092.93 37055.0559 102402.44
## 2005 Q4       70302.14 45366.20  95238.07 32165.9179 108438.35
## 2006 Q1       70875.52 42701.28  99049.77 27786.7299 113964.31
## 2006 Q2       71448.91 40265.73 102632.08 23758.3543 119139.46
## 2006 Q3       72022.29 37998.66 106045.93 19987.6338 124056.95
## 2006 Q4       72595.68 35860.96 109330.40 16414.7682 128776.59
## 2007 Q1       73169.07 33825.87 112512.26 12998.8419 133339.29
## 2007 Q2       73742.45 31874.22 115610.68  9710.5153 137774.39
## 2007 Q3       74315.84 29991.74 118639.94  6527.9777 142103.70
## 2007 Q4       74889.22 28167.52 121610.93  3434.5452 146343.90
## 2008 Q1       75462.61 26393.02 124532.19   417.1552 150508.06
## 2008 Q2       76035.99 24661.43 127410.56 -2534.6186 154606.61
## 2008 Q3       76609.38 22967.21 130251.55 -5429.2352 158648.00
## 2008 Q4       77182.77 21305.82 133059.72 -8273.6535 162639.19
```

```
autoplot(rwf(agri_train, h = 16, drift = TRUE))
```



Checking for the accuracy of the different forecasts

```
accuracy(agri_average, agri_test)
```

```
##                         ME      RMSE       MAE       MPE     MAPE     MASE
## Training set 1.157054e-12  7552.769  5897.575 -2.341148 12.29626 2.871492
## Test set     1.343546e+04 15768.153 13435.463 20.570480 20.57048 6.541643
##                     ACF1 Theil's U
## Training set  0.18800195        NA
## Test set     -0.01875492  1.402281
```

```
accuracy(agri_naive, agri_test)
```

```
##                     ME      RMSE      MAE         MPE     MAPE     MASE
## Training set   573.3859  9214.625 7472.898  -0.5860967 14.63427 3.638507
## Test set     -6673.7104 10614.207 9672.652 -12.7756862 16.75324 4.709554
##                     ACF1 Theil's U
## Training set -0.29335175        NA
## Test set     -0.01875492 0.9429065
```

```
accuracy(agri_seasonal_naive, agri_test)
```

```
##                    ME     RMSE      MAE      MPE     MAPE     MASE      ACF1
## Training set 1375.686 2422.847 2053.836 2.666126 4.233895 1.000000 0.3388125
## Test set     4730.520 5603.501 4759.550 7.570412 7.620245 2.317395 0.6483551
```

```
##                Theil's U
## Training set          NA
## Test set       0.4855898
```

```
accuracy(agri_drift, agri_test)
```

```
##                          ME      RMSE       MAE        MPE     MAPE     MASE
## Training set -2.707498e-12  9196.768  7725.827  -1.808916 15.28549 3.761657
## Test set     -1.154749e+04 13750.023 11786.832 -20.603121 20.91855 5.738935
##                    ACF1 Theil's U
## Training set -0.2933517        NA
## Test set     -0.1614651  1.200096
```

Putting the results in a table,

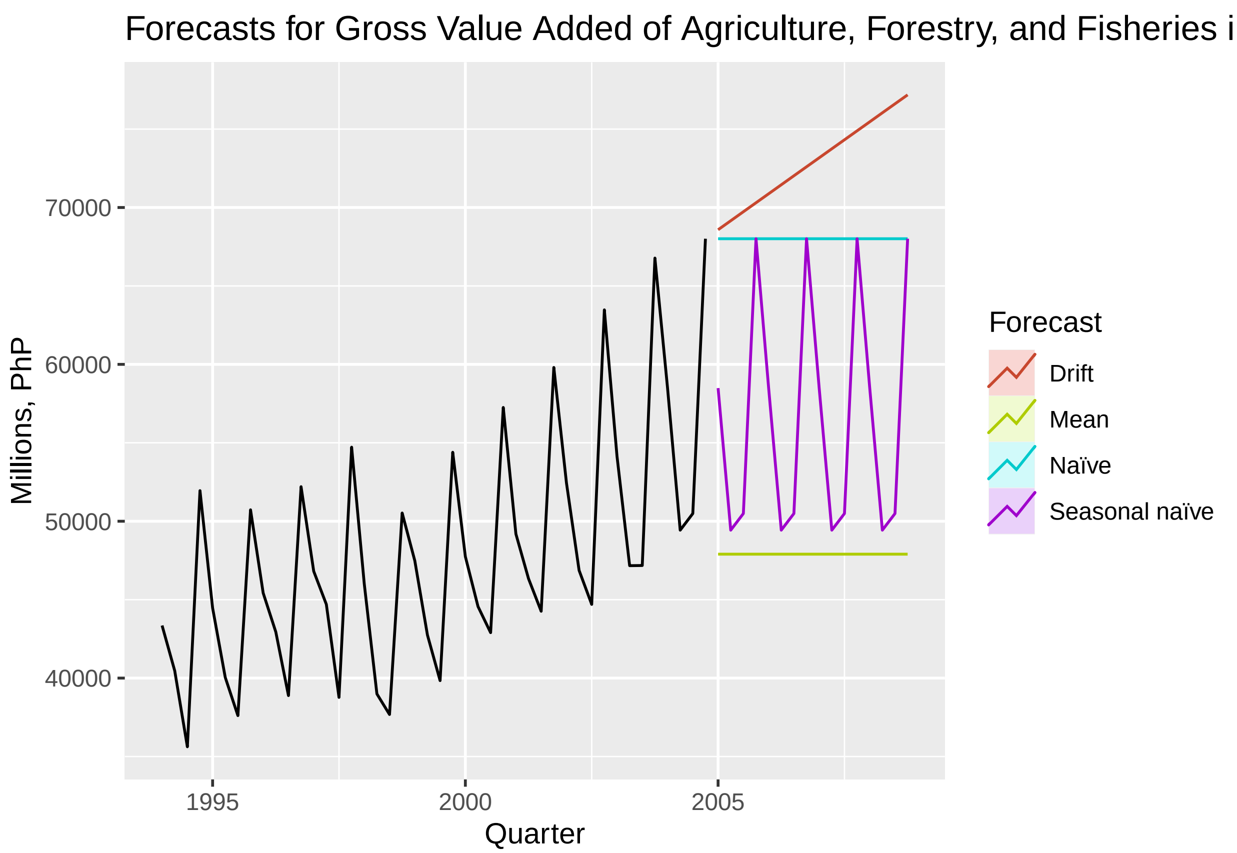
	RMSE	MAE	MAPE	MASE
Mean	15,768	13,435	21	7
Naïve	10,614	9,673	17	5
Seasonal Naïve	5,604	4,760	8	2
Drift	13,750	11,787	21	6

Figure 2: results

The best performing forecast is the Seasonal Naive method because it has the lowest RMSE, MAE, MAPE and MASE. This can be visually validated in the line graph below where the Seasonal Naive FC seems to give the best prediction and to capture the seasonality best as compared to the other forecasts.

Visualizing all of the forecasts in one graph

```
autoplot(agri_train) +
  autolayer(meanf(agri_train, h=16),
    series="Mean", PI=FALSE) +
  autolayer(naive(agri_train, h=16),
    series="Naïve", PI=FALSE) +
  autolayer(snaive(agri_train, h=16),
    series="Seasonal naïve", PI=FALSE) +
  autolayer(rwf(agri_train, h = 16, drift = TRUE),
    series="Drift", PI=FALSE) +
  ggtitle("Forecasts for Gross Value Added of Agriculture, Forestry, and Fisheries in the Philippines")
  xlab("Quarter") + ylab("Millions, PhP") +
  guides(colour=guide_legend(title="Forecast"))
```



Selecting the best forecast using the lowest RMSE in the time series cross validation,

RMSE for the Mean FC

```
e_average_agri <- tsCV(quarter_agri,meanf,h=16)

sqrt(mean(e_average_agri^2, na.rm = TRUE))
```

```
## [1] 11315.02
```

RMSE for the Naive FC

```
e_naive_agri <- tsCV(quarter_agri,naive,h=16)

sqrt(mean(e_naive_agri^2, na.rm = TRUE))
```

```
## [1] 9981.994
```

RMSE for the seasonal Naive FC

```
e_snaive_agri <- tsCV(quarter_agri,snaive,h=16)

sqrt(mean(e_snaive_agri^2, na.rm = TRUE))
```

```
## [1] 4957.234
```

RMSE for the drift FC

```
e_drift_agri <- tsCV(quarter_agri, rwf, drift=TRUE, h=16)

sqrt(mean(e_drift_agri^2, na.rm=TRUE))
```

```
## [1] 14302.88
```

The forecast with the lowest RMSE using time series cross validation is still the seasonal naive forecast which is consistent with the results of the RMSE, MAE, MAPE and MASE that were previously obtained.