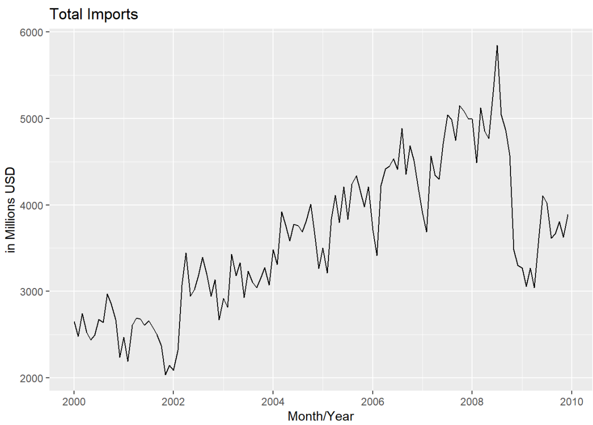
Stat 280 Forecasting: Analytics Project 2

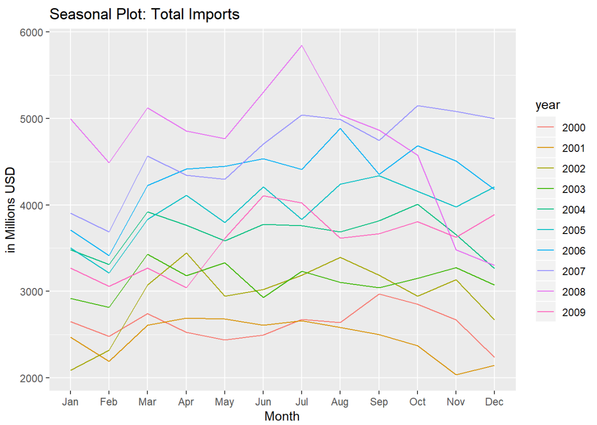
Cennen del Rosario and Arcel Galvez

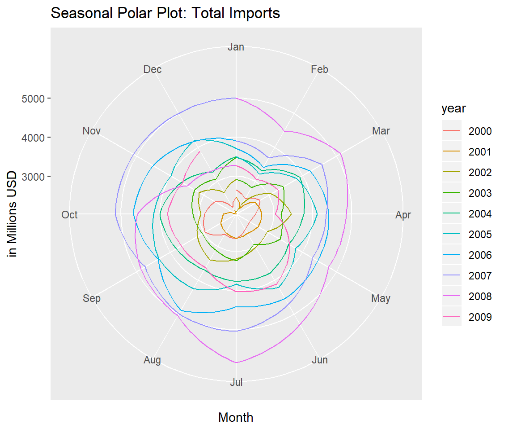
12/3/2019

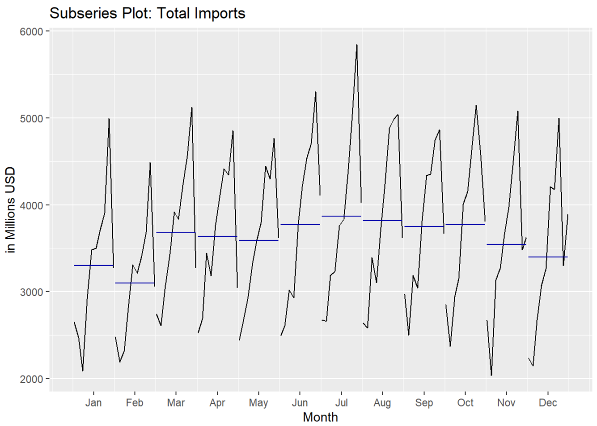
Part I - Visualization

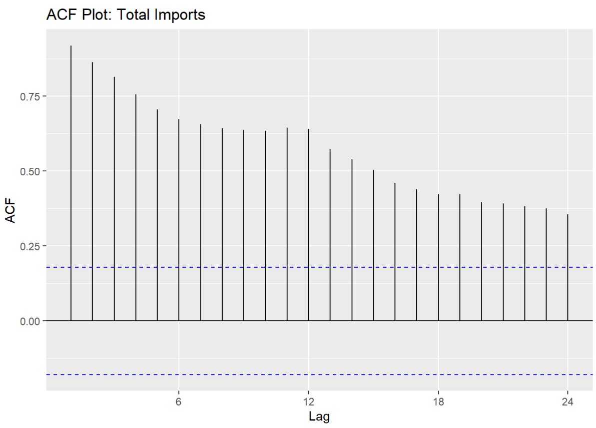
Before we proceed to the model building, let’s have an impression of the single time series data (that we have) through autoplots, seasonal plots, subseries plots and autocorrelation plots.





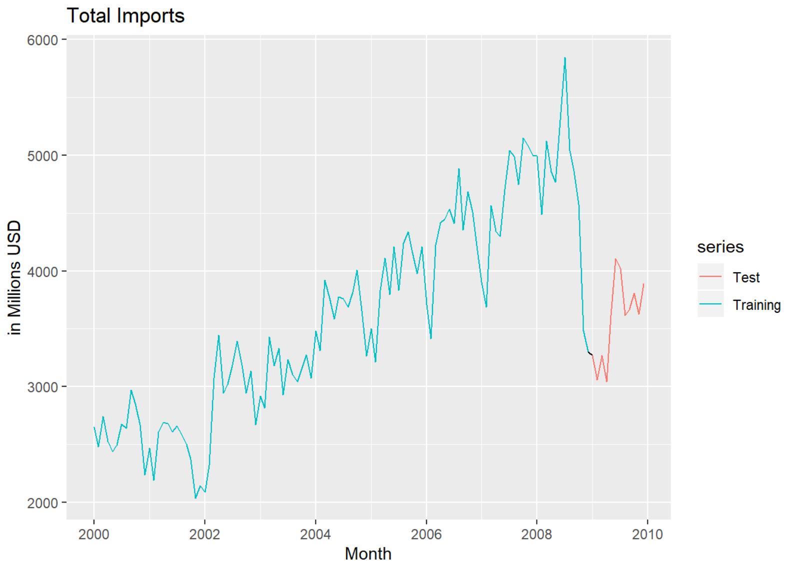


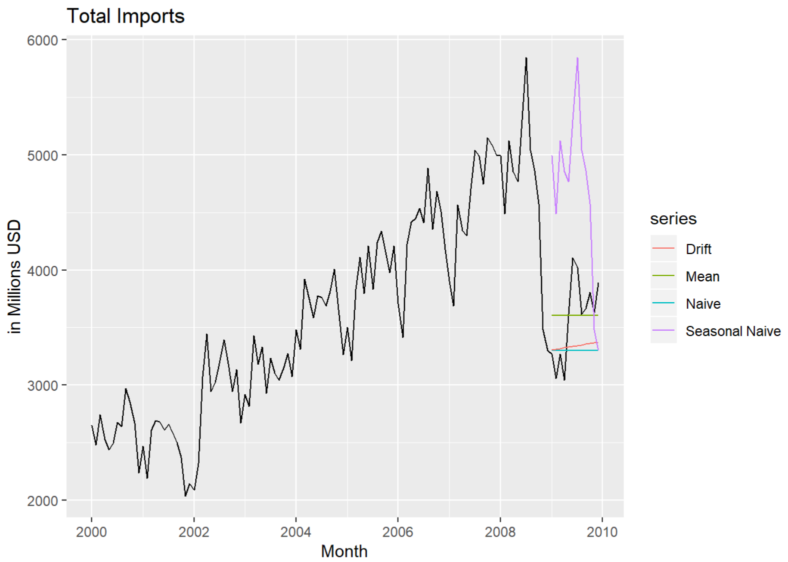




AS we observe on the autoplots and ACF plot,the total imports follows a downward trend from 2000 to 2002 and goes on a positive trend on the total imports from 2002. Seasonality on the data is very evident due to the slight “scalloping” of the graph as seen on the ACF plot (also seen on the seasonal plots). Also, we can say that the peak of the imports is every July of a particular year. On the other side of the story, February has the lowest number of imports in a particular year. One possible reason is because of the number of days in a month that affects the number of imports, though the difference of the numbers of days in February from the other months is 2 days (or 3 days), but it makes a huge difference.

Part II - Benchmark Forecasting





Visually, it appears that seasonal naive is the best performing forecast. However, in order to validate our observation and/or insights, let’s compute for the accuracy of the forecasting models.

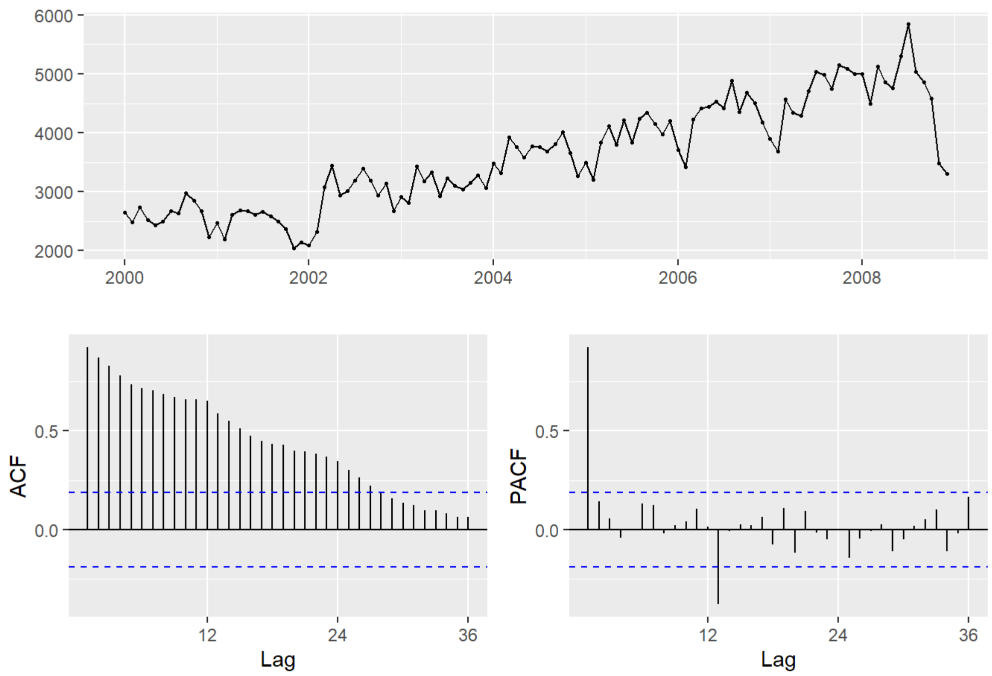
```
## Here, we the accuracy of the four benchmarking models to the test dataset,
mean.accuracy <- accuracy(imports.forecast.mean, imports.test)
naive.accuracy <- accuracy(imports.forecast.naive, imports.test)
drift.accuracy <- accuracy(imports.forecast.drift, imports.test)
snaive.accuracy <- accuracy(imports.forecast.snaive, imports.test)

rbind(mean.accuracy, naive.accuracy, drift.accuracy, snaive.accuracy)
```

```
##                         ME      RMSE       MAE         MPE      MAPE
## Training set  5.106718e-14  888.1245  759.6975  -6.4650201 22.680565
## Test set     -2.063889e+01  341.5312  275.6759  -1.5196181  7.950372
## Training set  6.074766e+00  337.7144  269.4766  -0.2337714  7.562484
## Test set      2.825833e+02  442.7990  376.7500   7.0211994 10.085834
## Training set  5.949356e-14  337.6597  270.2147  -0.4127492  7.589280
## Test set      2.430974e+02  407.7329  347.3886   5.9489349  9.336816
## Training set  2.631042e+02  528.6438  426.7708   6.1475570 11.740464
## Test set     -1.136917e+03 1361.2256 1259.4167 -32.9145508 36.107270
##                   MASE       ACF1 Theil's U
## Training set 1.7801065  0.9224994        NA
## Test set     0.6459577  0.5673864  1.113630
## Training set 0.6314317 -0.1982187        NA
## Test set     0.8827923  0.5673864  1.402207
## Training set 0.6331611 -0.1982187        NA
## Test set     0.8139934  0.5488508  1.293973
## Training set 1.0000000  0.6237613        NA
## Test set     2.9510373  0.5646037  4.291638
```

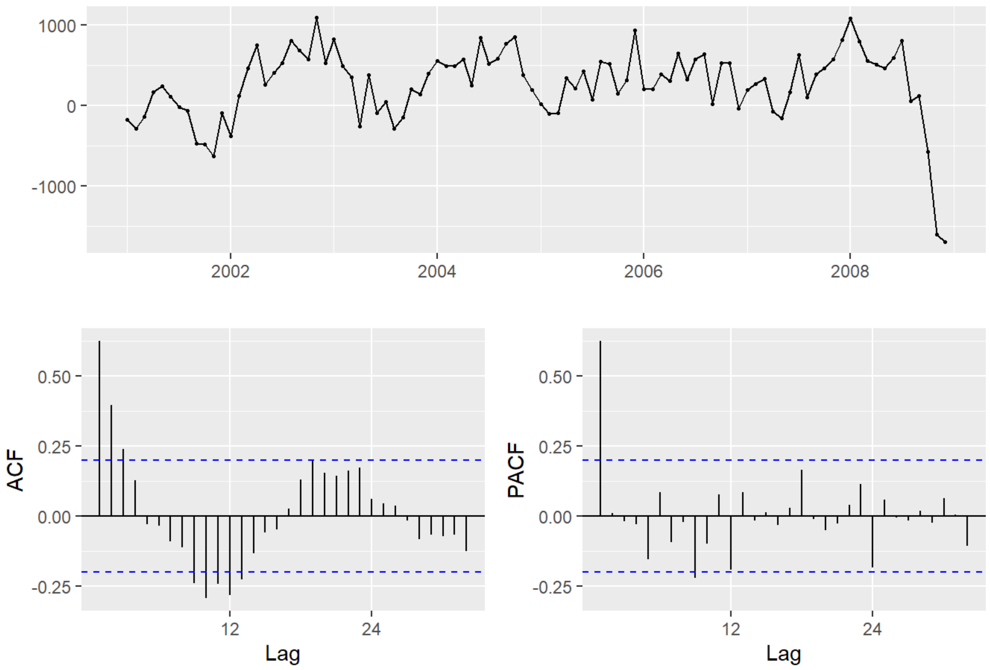
As we check on the accuracy measures (RMSE, MAPE, MAE), it appears that the forecasting by naive and drift method as the best forecasts. Although the two model do not really capture the wiggly behavior of the data.

Part III - SARIMA Forecasting Model

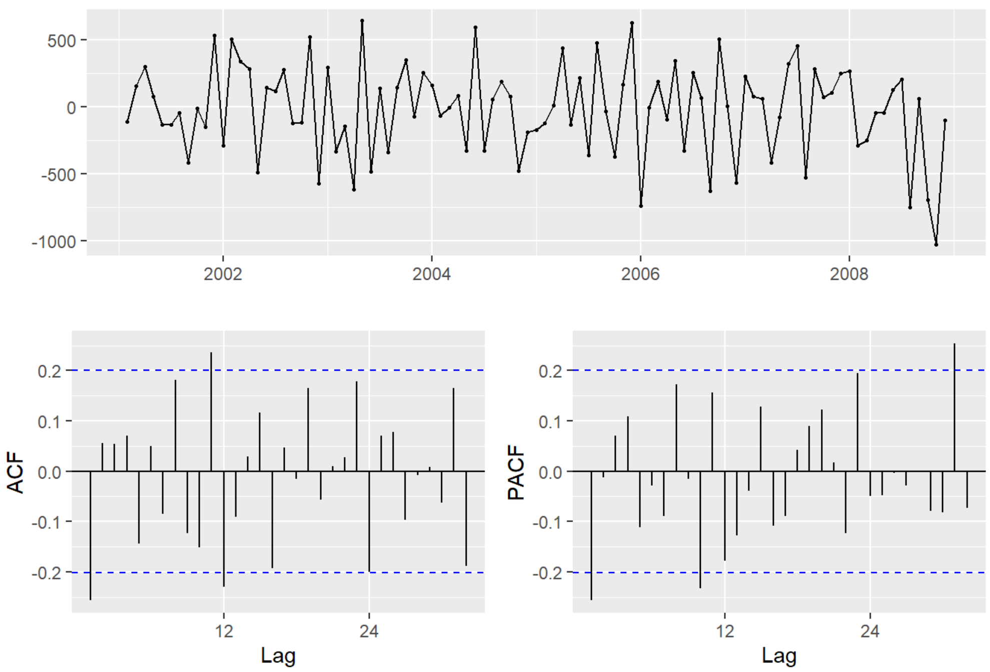
```
# Here, we try to identify the possible specifications of the model.
## As part of describing the possible specifications of the model, we plot the model.
imports.train %>% ggtsdisplay()
```




```
# It seems that the data is non-stationary, so we get the first seasonal differencing of the
 time series.

imports.train %>% diff(lag=12) %>% ggtsdisplay()
```




```r
# After the first seasonal differencing, it appears that the time series data is still non-stationary. From there, we try to aplly non-seasonal differencing to the data.

imports.train %>% diff(lag=12) %>% diff() %>% ggtsdisplay()
```




```r
# After one seasonal and one non-seasonal differencing, it appears that our data is somewhat stationary. As we can see, there are some significant spikes at lags 1 and 11 in the ACF plot. This means that it suggests an MA(1) or MA(11) term; however, we consider only adding MA(1) term.
# Also, the spike is significant at lag 12, since this time seriesdata has a monthly seasonality, it means that it also suggest a seasonal MA(1) term.
# Checking the PACF plot, there is a significant spike at lag 1, means it suggest an AR(1) term and remaining lags tapers off and are not significant.
```

As we take a look on the time series data (in terms of the autoplot, ACF and PACF plots), it seems that the data is non-stationary, so we get the first seasonal differencing of the time series. After the first seasonal differencing, it appears that the time series data is still non-stationary. From there, we try to apply non-seasonal differencing to the data.

After one seasonal and one non-seasonal differencing, it appears that our data is somewhat stationary. As we can see, there are some significant spikes at lags 1 and 11 in the ACF plot. This means that it suggests an MA(1) or MA(11) term; however, we consider only adding MA(1) term. Also, the spike is significant at lag 12, since this time series data has a monthly seasonality, it means that it also suggest a seasonal MA(1) term. Checking the PACF plot, there is a significant spike at lag 1, means it suggest an AR(1) term and remaining lags tapers off and are not significant.

With all those insights, we can say that the possible specifications of the SARIMA model may include the following: * p = {0, 1} [a non-seasonal AR term] * q = {0, 1} [a non-seasonal MA term] * Q = {0, 1} [a seasonal MA term].

From those insights, we can have 6 SARIMA models (actually 8 models, however, we remove the non-feasible/irrelevant models). Also, we fit a SARIMA model using the auto.arima() function, which is the automated version of the Arima().

```
# ARIMA Model Fitting
imports.sarima.fit1 <- Arima(imports.train, order = c(1, 1, 1), seasonal = c(0, 1, 1))
summary(imports.sarima.fit1)
```

```
## Series: imports.train
## ARIMA(1,1,1)(0,1,1)[12]
##
## Coefficients:
##           ar1      ma1     sma1
##       -0.1423  -0.0764  -0.7063
## s.e.   0.4295   0.4315   0.1425
##
## sigma^2 estimated as 86847:  log likelihood=-677.6
## AIC=1363.19   AICc=1363.64   BIC=1373.41
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE    MAPE      MASE
## Training set -3.863338 271.9948 204.4786 -0.2230936 5.60717 0.4791297
##                       ACF1
## Training set -0.0005348731
```

```
imports.sarima.fit2 <- Arima(imports.train, order = c(1, 1, 1), seasonal = c(0, 0, 1))
summary(imports.sarima.fit2)
```

```
## Series: imports.train
## ARIMA(1,1,1)(0,0,1)[12]
##
## Coefficients:
##           ar1      ma1    sma1
##       -0.1638  -0.0252  0.4134
## s.e.   0.4220   0.4273  0.0889
##
## sigma^2 estimated as 93340:  log likelihood=-763.7
## AIC=1535.4   AICc=1535.8   BIC=1546.09
##
## Training set error measures:
##                     ME     RMSE     MAE        MPE     MAPE      MASE
## Training set -1.234925 299.8049 236.782 -0.3837376 6.760117 0.5548223
##                      ACF1
## Training set -0.001042758
```

```
imports.sarima.fit3 <- Arima(imports.train, order = c(1, 1, 1), seasonal = c(0, 1, 0))
summary(imports.sarima.fit3)
```

```
## Series: imports.train
## ARIMA(1,1,1)(0,1,0)[12]
##
## Coefficients:
##           ar1      ma1
##       -0.2244  -0.0300
## s.e.   0.3256   0.3301
##
## sigma^2 estimated as 111993:  log likelihood=-686.07
## AIC=1378.13   AICc=1378.39   BIC=1385.79
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -17.49048 310.5454 232.4617 -0.568435 6.519943 0.544699
##                      ACF1
## Training set -0.005497597
```

```
imports.sarima.fit4 <- Arima(imports.train, order = c(1, 1, 0), seasonal = c(0, 1, 1))
summary(imports.sarima.fit4)
```

```
## Series: imports.train
## ARIMA(1,1,0)(0,1,1)[12]
##
## Coefficients:
##           ar1     sma1
##       -0.2140  -0.7059
## s.e.   0.0999   0.1422
##
## sigma^2 estimated as 85953:  log likelihood=-677.61
## AIC=1361.22   AICc=1361.49   BIC=1368.88
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -3.911582 272.0566 204.5976 -0.2236928 5.606078 0.4794086
##                      ACF1
## Training set -0.005280876
```

```
imports.sarima.fit5 <- Arima(imports.train, order = c(1, 1, 0), seasonal = c(0, 0, 1))
summary(imports.sarima.fit5)
```

```
## Series: imports.train
## ARIMA(1,1,0)(0,0,1)[12]
##
## Coefficients:
##           ar1    sma1
##       -0.1879  0.4140
## s.e.   0.0948  0.0883
##
## sigma^2 estimated as 92448:  log likelihood=-763.7
## AIC=1533.41   AICc=1533.64   BIC=1541.43
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE     MASE
## Training set -1.303375 299.7997 236.7473 -0.3843604 6.757933 0.554741
##                      ACF1
## Training set -0.002097896
```

```
imports.sarima.fit6 <- Arima(imports.train, order = c(1, 1, 0), seasonal = c(0, 1, 0))
summary(imports.sarima.fit6)
```

```
## Series: imports.train
## ARIMA(1,1,0)(0,1,0)[12]
##
## Coefficients:
##           ar1
##       -0.2522
## s.e.   0.0988
##
## sigma^2 estimated as 110813:  log likelihood=-686.07
## AIC=1376.14   AICc=1376.27   BIC=1381.25
##
## Training set error measures:
##                     ME     RMSE     MAE        MPE    MAPE      MASE
## Training set -17.43637 310.5608 232.475 -0.5665305 6.51744 0.5447303
##                      ACF1
## Training set -0.007936389
```

```
# Trying to let R select the optimal values of (p, d, q)(P, D, Q)[12]
imports.sarima.fit7 <- auto.arima(imports.train)
summary(imports.sarima.fit7)
```

```
## Series: imports.train
## ARIMA(1,1,0)(1,0,0)[12]
##
## Coefficients:
##           ar1    sar1
##       -0.2123  0.4960
## s.e.   0.0943  0.0933
##
## sigma^2 estimated as 87227:  log likelihood=-761.16
## AIC=1528.33   AICc=1528.56   BIC=1536.35
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -5.360146 291.2113 232.5008 -0.4395008 6.637717 0.5447908
##                      ACF1
## Training set -0.005124402
```

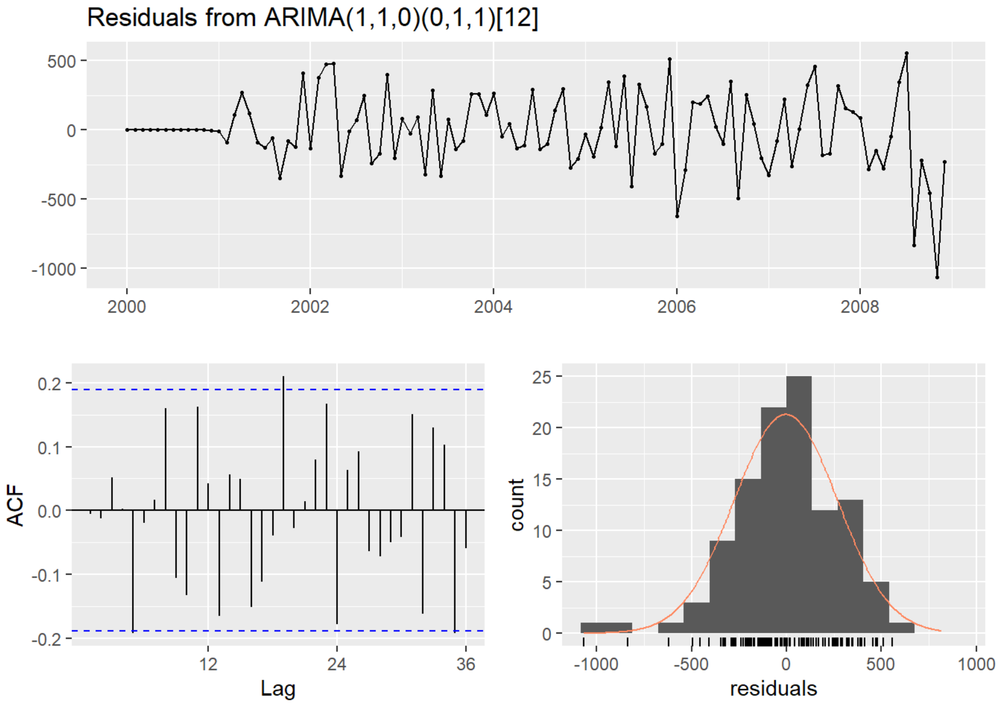
Summarizing the models that we fitted earlier,

Models	AICc	AIC	BIC
ARIMA(1,1,1)(0,1,1)[12]	1363.638	1363.193	1373.409
ARIMA(1,1,1)(0,0,1)[12]	1535.796	1535.404	1546.095
ARIMA(1,1,1)(0,1,0)[12]	1378.394	1378.130	1385.792
ARIMA(1,1,0)(0,1,1)[12]	**1361.485**	**1361.221**	**1368.883**
ARIMA(1,1,0)(0,0,1)[12]	1533.640	1533.407	1541.426
ARIMA(1,1,0)(0,1,0)[12]	1376.270	1376.139	1381.247

Models	AICc	AIC	BIC
ARIMA(1,1,0)(1,0,0)[12]	1528.563	1528.330	1536.348

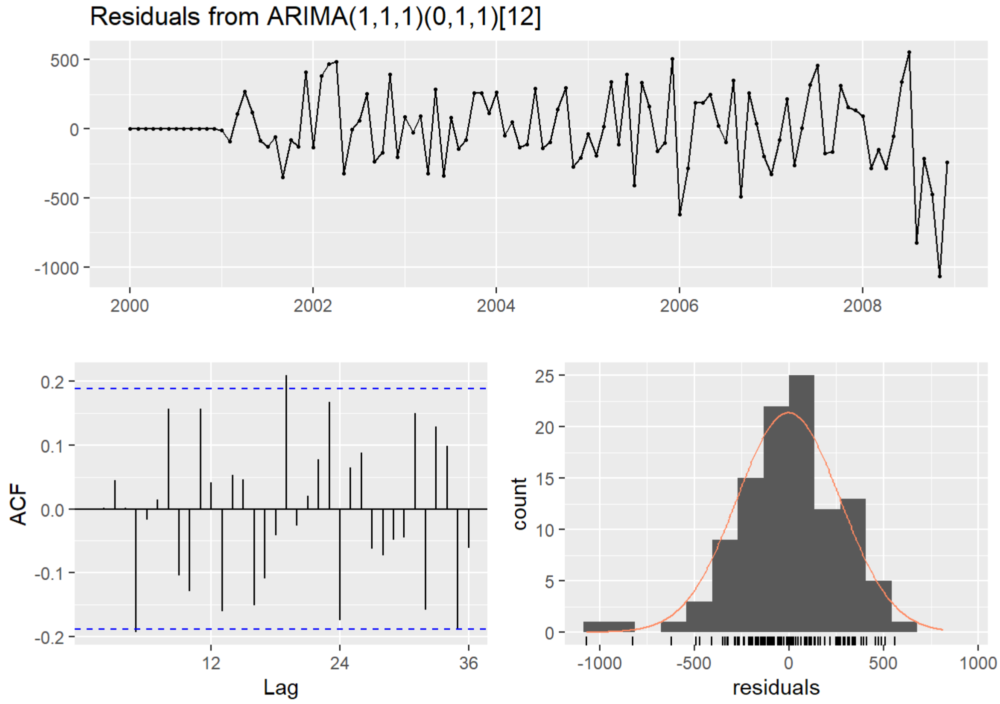
As we can see on the summary of the seven fitted SARIMA models on the time series data, checking the AICc values of the models, we can say that the SARIMA$(1,1,0)(0,1,1)_{12}$ is the best fitting model (lowest AICc, better model). Using that model, we evaluate and assess its residuals.

```
# Checking the residuals of the best model.
imports.sarima.fit4 %>% checkresiduals()
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0)(0,1,1)[12]
## Q* = 30.342, df = 20, p-value = 0.06449
##
## Model df: 2.   Total lags used: 22
```

```
# Evaluating the residuals of the second best model.
imports.sarima.fit1 %>% checkresiduals()
```

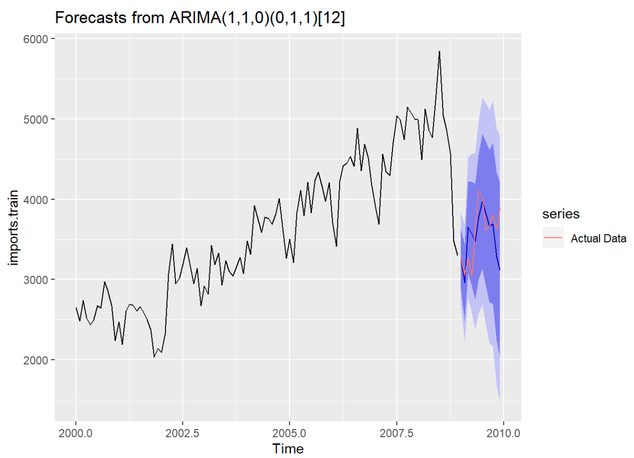


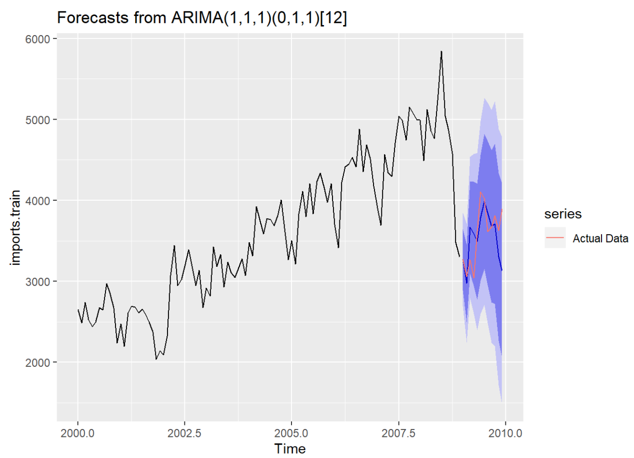

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,1)(0,1,1)[12]
## Q* = 29.458, df = 19, p-value = 0.05911
##
## Model df: 3.   Total lags used: 22
```

As we can see, the residuals appear to be “somewhat” white noise, since there is a significant spike at the 19th lag (as seen on the ACF plot, only lag that goes outside the confidence bands). However, as tested using the Ljung Box Test, it agrees to the fact that the residuals are not distinguishable from a white noise.

In order to have another comparison, we consider the second best model that is SARIMA$(1,1,1)(0,1,1)_{12}$. Based on the assessment of the residuals of the model, it appears that they have almost the same results to our best SARIMA model. It also has a significant spike at the 19th lag; however, by Ljung-Box test, has same results as the best model.

Therefore, we can now proceed to the forecasting. Here we use the two models in forecasting 12 months ahead.



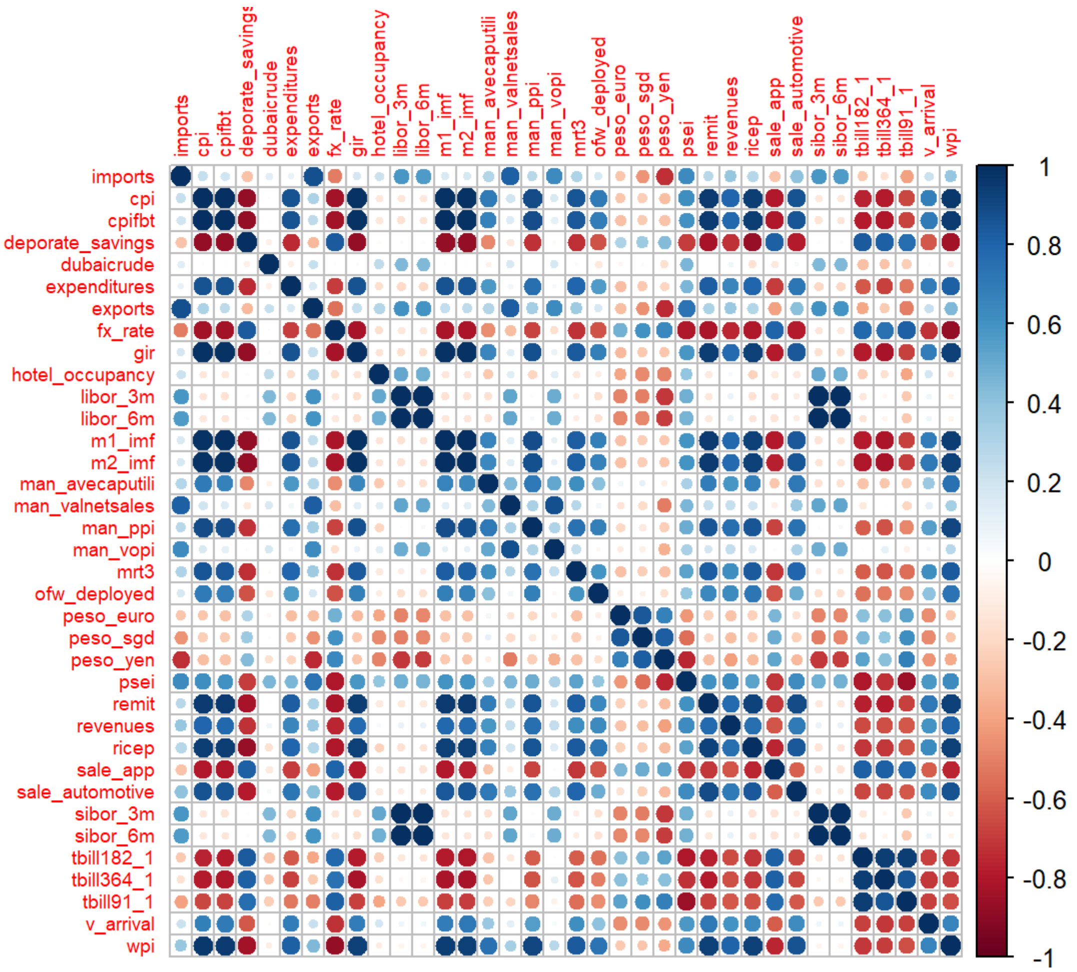


Note that ARIMA$(1,1,0)(0,1,1)_{12}$ model has a slightly better performance. But both could forecast the actual data well.

Part IV - Dynamic Regression Forecasting Model

4.a. Linear Regression with ARIMA errors

First, we check the correlation of all variables for the time period where there are no missing data. For this work, data starting from April 2001 until November 2009 will be considered.



From the heatmap and table below, we can identify variables that are correlated. In the table, highly correlated groups are identified by “cpi”, “exports”, and “peso_sgd”.

Clustering of predictors

Variable	Correlated with	Variable	Correlated with

[a] Has correlation with other variables

Variable	Correlated with	Variable	Correlated with
cpi	cpifbt deporate_savings expenditures fx_rate gir m1_imf m2_imf man_avecaputili man_ppi mrt3 ofw_deployed psei remit revenues ricep sale_app sale_automotive tbill182_1 tbill364_1 tbill91_1 v_arrival wpi	peso_sgd	fx_rate[a] libor_3m[a] libor_6m[a] peso_euro peso_yen[a] psei[a] sibor_3m[a] sibor_6m[a] v_arrival[a]
dubaicrude	•	hotel_occupancy	•
exports	fx_rate[a] libor_3m libor_6m man_valnetsales man_vopi peso_yen psei[a] sibor_3m sibor_6m	imports	exports fx_rate libor_3m libor_6m man_valnetsales man_vopi peso_yen psei sibor_3m sibor_6m

[a] Has correlation with other variables

Initially, we reduce our scope to the following predictors because they are more likely to affect the total imports. But further consideration shows that “peso_yen”, “peso_euro” and “peso_sgd” are highly correlated; hence, we will only consider “peso_yen” among them. Although it is highly correlated with “exports”, we will also consider the latter because it is highly correlated with the imports. On the other hand, “man_vopi” will be included as well. For the “cpi” group, we will only consider “fx_rate” and “sale_app” as indicators because the rest are highly correlated.

Hence, our candidate predictors are “fx_rate”, “sale_app”, “exports”, “peso_yen”, “man_vopi”, “dubaicrude”, and “hotel_occupancy”.

Predictors that would make sense

Variable	Correlated.with
cpi	fx_rate, man_ppi, revenues, ricep, sale_app
exports	man_vopi, peso_yen
peso_sgd	peso_euro, peso_yen
dubaicrude	–
hotel_occupancy	–

Because of the limitation of the data for “man_vopi”, we use only the part starting from January, 2001.

Possible models for linear regression with ARIMA errors

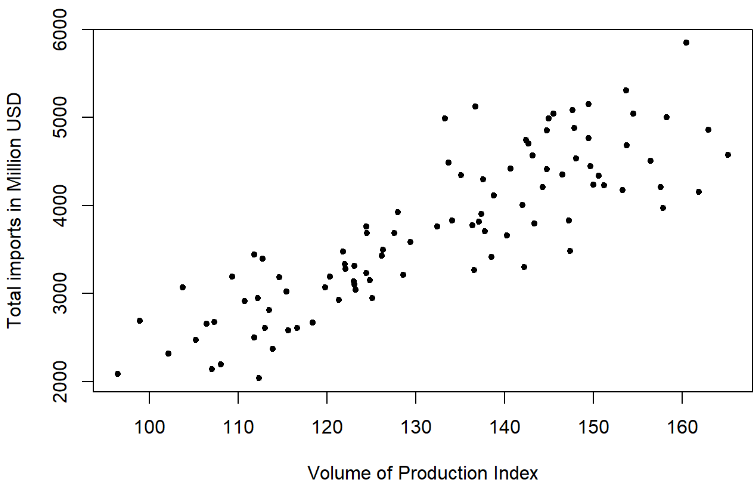
fx_rate	sale_app	exports	peso_yen	man_vopi	dubaicrude	hotel_occupancy	AICc
0	0	0	0	0	0	1	1207.105
0	0	0	0	0	1	1	1207.629
0	1	0	1	0	0	1	1207.886
1	1	0	0	0	1	1	1208.041
0	0	0	1	0	0	1	1208.206
1	0	0	0	0	1	1	1208.600
0	1	0	1	0	1	1	1208.949
0	0	0	1	0	1	1	1209.475
1	1	0	1	0	0	1	1209.520
1	0	0	0	0	0	1	1209.717
1	1	0	1	0	1	1	1209.907
0	1	0	0	0	1	1	1210.112
0	1	0	0	0	0	1	1210.560
1	1	0	0	0	0	1	1211.681
1	0	0	1	0	0	1	1212.196
1	**0**	**0**	**1**	**0**	**1**	**1**	**1214.619**
0	**0**	**0**	**0**	**1**	**0**	**0**	**1349.424**
0	0	0	0	1	1	0	1351.594

fx_rate	sale_app	exports	peso_yen	man_vopi	dubaicrude	hotel_occupancy	AICc
0	0	1	0	1	0	0	1354.374
1	0	1	0	1	0	1	1355.192

It is interesting to note that the lowest AICc is achieved when only "hotel_occupancy" is the predictor variable. However, looking at the table above, there is sudden increase in the AICc if this variable is removed. In that case, "man_vopi" (Manufacturing Volume of Production Index) as the sole predictor would be the best performing model. This is shown in the following table. For this reason, we select "man_vopi" as the sole predictor for linear regression.

Possible linear regression with ARIMA errors without hotel_occupancy

	fx_rate	sale_app	exports	peso_yen	man_vopi	dubaicrude	hotel_occupancy	AICc
17	0	0	0	0	1	0	0	1349.424
18	0	0	0	0	1	1	0	1351.594
19	0	0	1	0	1	0	0	1354.374
21	1	0	1	0	1	0	0	1355.542
23	0	1	1	0	1	0	0	1356.156
24	1	0	1	1	1	0	0	1356.450
25	0	0	1	1	1	0	0	1356.645
26	0	0	1	0	1	1	0	1356.691
30	1	1	1	0	1	0	0	1357.482
31	1	0	1	0	1	1	0	1357.914



Considering the scatterplot of the Volume of Production Index and Total Imports, there seems to be a linear relation between the two. We then fit an ordinary linear regression with ARIMA error.

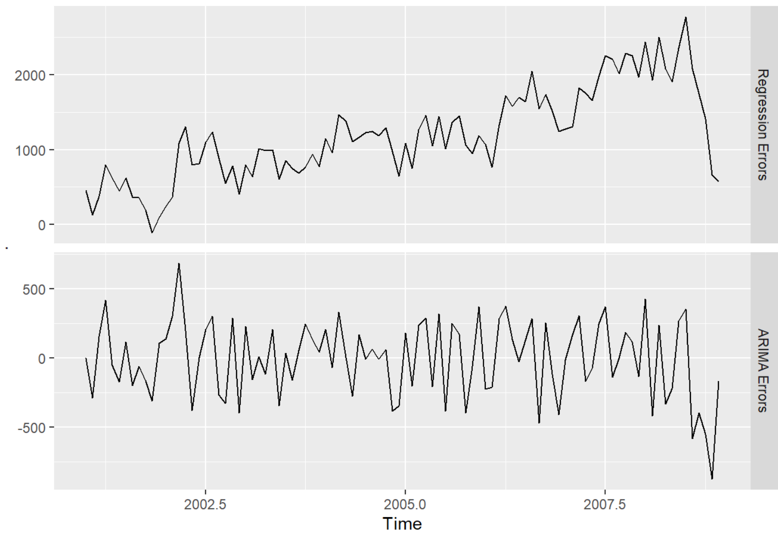
```
best.lm.model <- auto.arima(lm.train.ts[, "imports"],
                            xreg = lm.train.ts[, "man_vopi"], approximation = FALSE)
best.lm.model
```

```
## Series: lm.train.ts[, "imports"]
## Regression with ARIMA(0,1,1)(1,0,0)[12] errors
##
## Coefficients:
##           ma1    sar1     xreg
##       -0.2478  0.4499  19.1429
## s.e.   0.1014  0.1047   4.4820
##
## sigma^2 estimated as 79266:  log likelihood=-670.49
## AIC=1348.98   AICc=1349.42   BIC=1359.2
```

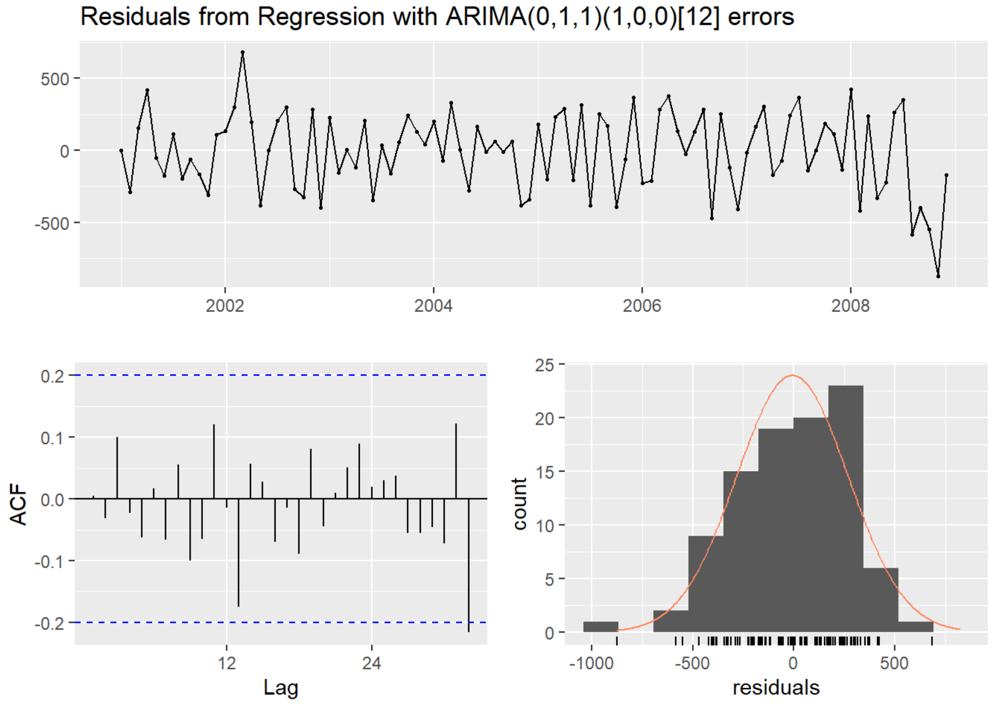
The model took care of non-stationary proprty of the target variable. The fitted model is

$$\begin{aligned} y_t &= 19.14\, x_t + \eta_t \\ (1 - 0.45B^{12})(1 - B)\eta_t &= (1 - 0.25B)\varepsilon_t \\ \varepsilon_t &\sim \text{NID}(0, 79266) \end{aligned}$$

where y_t is the total imports, x_t is the manufacturing production index, η_t is the regression error, and ε_t is the ARIMA error at month t.

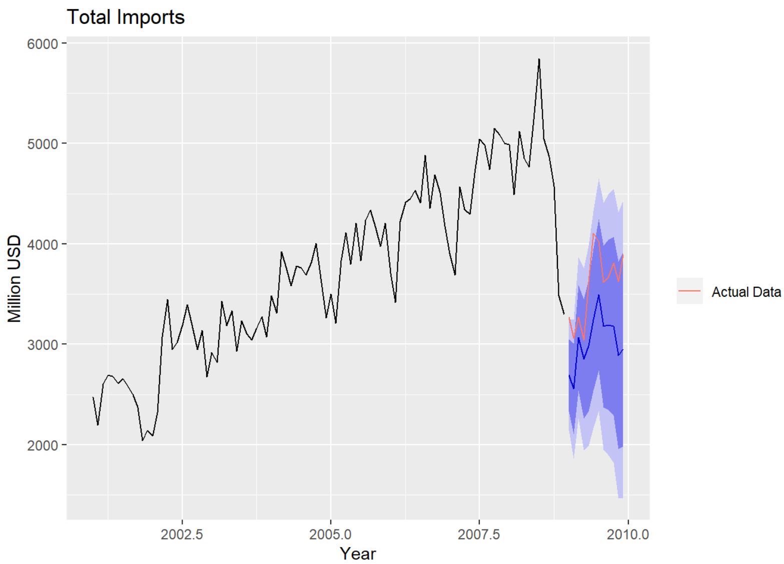


It is the ARIMA errors that should resemble a white noise. Based on the graph and Ljung-Box test below, they seem to resemble a white noise.




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(0,1,1)(1,0,0)[12] errors
## Q* = 11.827, df = 16, p-value = 0.7558
##
## Model df: 3.   Total lags used: 19
```

So, We forecast using the test data set. The prediction interval prettily covers the actual values for the year 2009. The accuracy measures are shown below. We will use this later on when choosing the superior model.




```
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set  -7.115456 275.6136 228.0441 -0.3858791  6.32439 0.5029461
## Test set     559.876774 600.6456 559.8768 15.3565211 15.35652 1.2347953
```

4.b. Stochastic and deterministic trends

We proceed with another models, the deterministic and stochastic trend models. This time, we maximime and use the Total Imports data from 2000 to 2008. (No dependency in other variables is needed.)

The fitted deterministic trend is obtained as follows.

```
trend <- seq_along(imports.train)
deterministic.model <- auto.arima(imports.train, d = 0, xreg=trend, approximation = F)
deterministic.model
```

```
## Series: imports.train
## Regression with ARIMA(1,0,0)(1,0,0)[12] errors
##
## Coefficients:
##          ar1    sar1  intercept     xreg
##       0.7418  0.4476  2458.0930  20.1977
## s.e.  0.0788  0.0989   311.0129   4.8639
##
## sigma^2 estimated as 84911:  log likelihood=-765.83
## AIC=1541.65   AICc=1542.24   BIC=1555.06
```

$$\begin{aligned} y_t &= 2458.09 + 0.17t + \eta_t \\ (1 - 0.74B)(1 - 0.45B^{12})\eta_t &= \varepsilon_t \\ \varepsilon_t &\sim \text{NID}(0, 84911) \end{aligned}$$

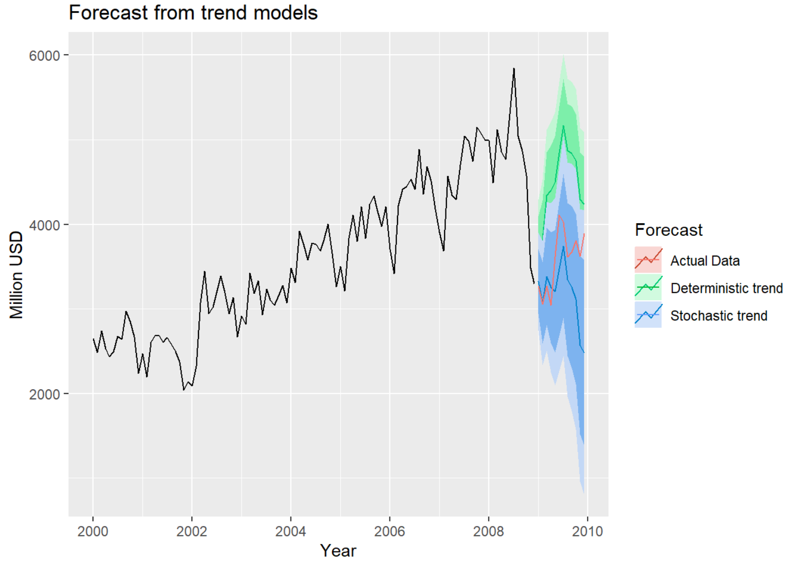
where y_t is the total imports, x_t is the manufacturing production index, η_t is the regression error, and ε_t is the ARIMA error at month t.

Similarly, we fit a stochastic model with

```
## Series: imports.train
## ARIMA(1,1,0)(1,0,0)[12]
##
## Coefficients:
##           ar1    sar1
##       -0.2123  0.4960
## s.e.   0.0943  0.0933
##
## sigma^2 estimated as 87227:  log likelihood=-761.16
## AIC=1528.33   AICc=1528.56   BIC=1536.35
```

$$\begin{aligned} y_t - y_{t-1} &= \eta_t^{'} \\ (1 + 0.21B)(1 - 0.50B^{12})\eta_t^{'} &= \varepsilon_t \\ \varepsilon_t &\sim \text{NID}(0, 87227) \end{aligned}$$

Forecasting the next 12 months for both trend models, we get the following.



The deterministic trend model produces larger forecasts than the stochastic model. This is apparent in the wider prediction intervals in the graph above. We also check the accuracy measures for each trend model.

```
deterministic.accuracy <- accuracy(deterministic.forecast, imports.test)
deterministic.accuracy[,1:6]
```

```
##                      ME     RMSE      MAE         MPE      MAPE      MASE
## Training set   -3.200033 285.9476 228.6536  -0.7760262  6.565312 0.5357762
## Test set     -898.632180 948.5689 898.6322 -25.4536149 25.453615 2.1056551
```

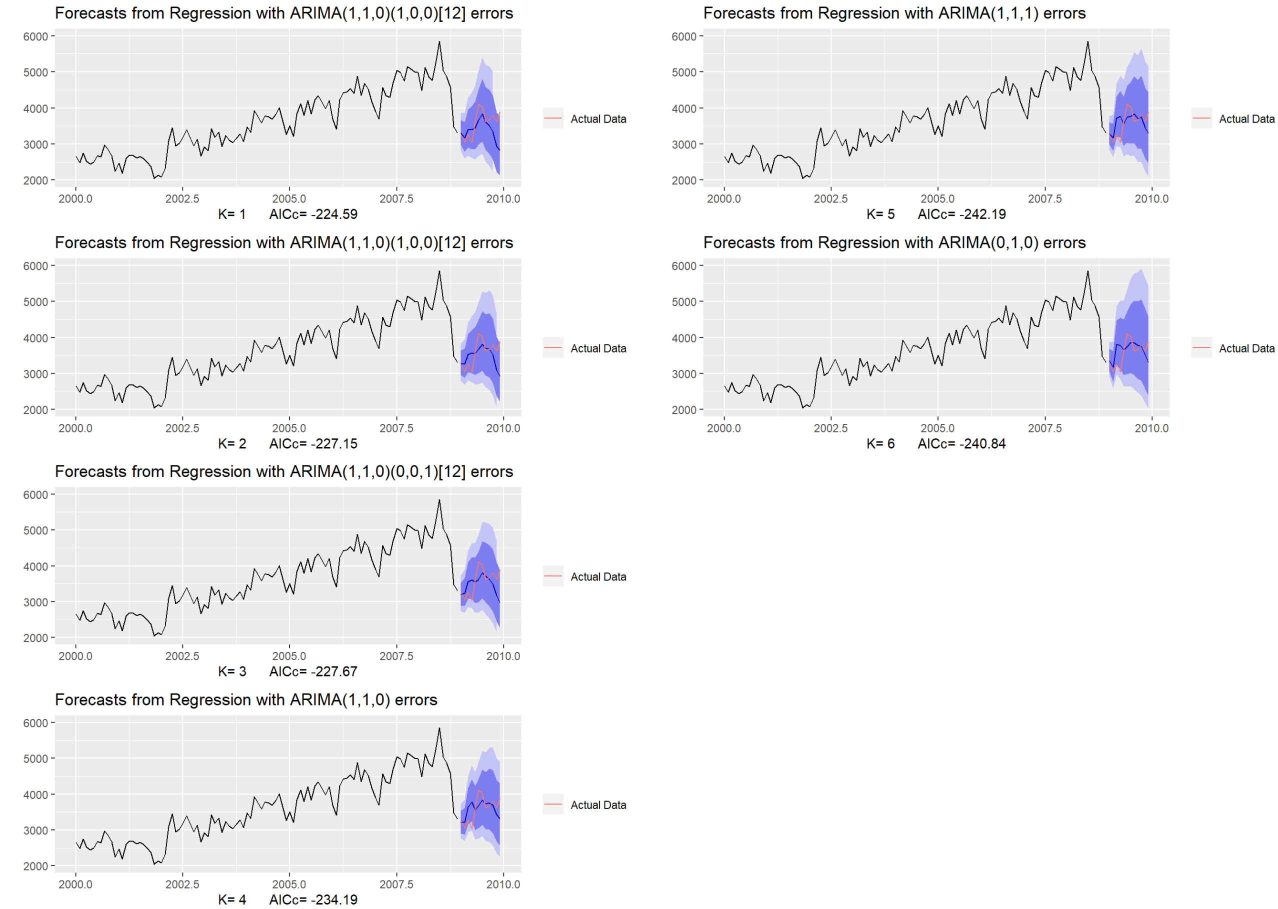
```
stochastic.accuracy <- accuracy(stochastic.forecast, imports.test)
stochastic.accuracy[,1:6]
```

```
##                     ME     RMSE      MAE        MPE      MAPE      MASE
## Training set  -5.360146 291.2113 232.5008 -0.4395008  6.637717 0.5447908
## Test set     396.876239 614.3253 463.2717 10.2631795 12.379731 1.0855280
```

Note that up to this point, the models fit ARIMA error with very large variance. The standard deviation goes around 200 to 300 for around 2000 to 6000 values (in millions) of the total imports.

4.c. Dynamic harmonic regression

Using the harmonic terms, we fit models with varying number of harmonic terms. For $K = 1$ to $K = 6$, we plot the fit a linear regression with ARIMA errors and forecast for the next 12 months (as shown in the proceeding graphs). $K = 6$ is the maximum allowed number of terms to respect the Nyquist criterion. In turns out that $K = 5$ produces the optimal result based on the lowest AICc.



So we fit again the model and get the accuracy measures. We will use these measures later.

```
## Series: imports.train
## Regression with ARIMA(1,1,1) errors
## Box Cox transformation: lambda= 0
##
## Coefficients:
##          ar1     ma1    S1-12    C1-12    S2-12    C2-12    S3-12   C3-12
##      -0.8629  0.6998  -0.0355  -0.0628  -0.0206  -0.0288  -0.0183  0.0121
## s.e.  0.1084  0.1530   0.0165   0.0163   0.0085   0.0085   0.0061  0.0061
##        S4-12   C4-12   S5-12    C5-12
##       0.0231  0.0078  0.0192  -0.0126
## s.e.  0.0051  0.0051  0.0051   0.0051
##
## sigma^2 estimated as 0.005177:  log likelihood=136.05
## AIC=-246.11   AICc=-242.19   BIC=-211.36
```

```
##                     ME     RMSE      MAE          MPE     MAPE      MASE
## Training set  8.271733 249.4999 201.6161 -0.003071587 5.576229 0.4724224
## Test set     -2.393079 340.8513 255.1107 -0.806206026 7.260524 0.5977698
```

4.d. Lagged predictors

Lastly, we consider modeling with lagged predictor. For this case, we will only consider “man_vopi” for the sake of comparison with ordinary linear regression with ARIMA errors from the past section. We consider at most 6-month lags of the said predictor. Data from 2001 up to 2008 are used again here because of the limitations of “man_vopi” predictor.

```
ProductionIndex <- cbind(
    ProdLag0 = lag.train.ts[,"man_vopi"],
    ProdLag1 = stats::lag(lag.train.ts[,"man_vopi"], -1),
    ProdLag2 = stats::lag(lag.train.ts[,"man_vopi"], -2),
    ProdLag3 = stats::lag(lag.train.ts[,"man_vopi"], -3),
    ProdLag4 = stats::lag(lag.train.ts[,"man_vopi"], -4),
    ProdLag5 = stats::lag(lag.train.ts[,"man_vopi"], -5),
    ProdLag6 = stats::lag(lag.train.ts[,"man_vopi"], -6)) %>%
        head(NROW(lag.train.ts))

length.lag.data <- nrow(lag.train.ts)
lag.fit1 <- auto.arima(lag.train.ts[7:length.lag.data,"imports"],
                       xreg = ProductionIndex[7:length.lag.data, 1],
                       stationary = FALSE)
lag.fit2 <- auto.arima(lag.train.ts[7:length.lag.data,"imports"],
                       xreg = ProductionIndex[7:length.lag.data, 1:2],
                       stationary = FALSE)
lag.fit3 <- auto.arima(lag.train.ts[7:length.lag.data,"imports"],
                       xreg = ProductionIndex[7:length.lag.data, 1:3],
                       stationary = FALSE)
lag.fit4 <- auto.arima(lag.train.ts[7:length.lag.data,"imports"],
                       xreg = ProductionIndex[7:length.lag.data, 1:4],
                       stationary = FALSE)
lag.fit5 <- auto.arima(lag.train.ts[7:length.lag.data,"imports"],
                       xreg = ProductionIndex[7:length.lag.data, 1:5],
                       stationary = FALSE)
lag.fit6 <- auto.arima(lag.train.ts[7:length.lag.data,"imports"],
                       xreg = ProductionIndex[7:length.lag.data, 1:6],
                       stationary = FALSE)
lag.fit7 <- auto.arima(lag.train.ts[7:length.lag.data,"imports"],
                       xreg = ProductionIndex[7:length.lag.data, 1:7],
                       stationary = FALSE)
```

```
## [1] 1278.068 1280.258 1269.946 1271.280 1272.334 1270.574 1290.384
```

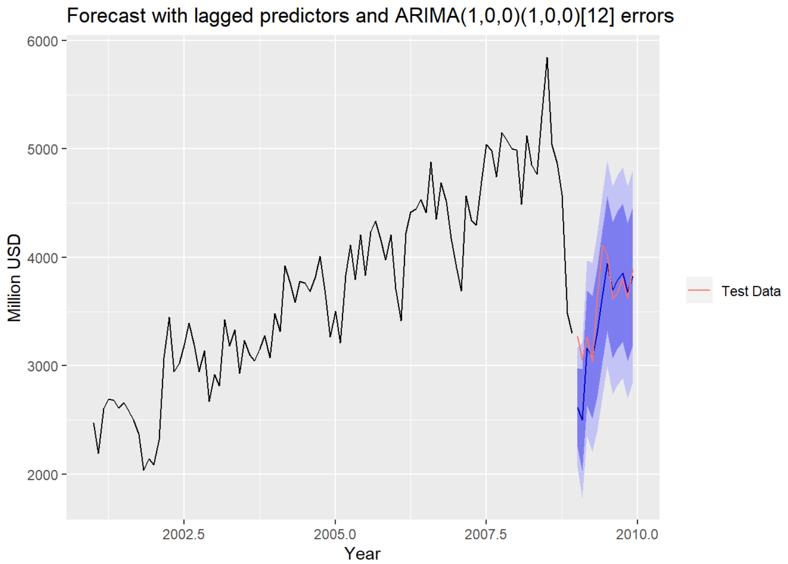
The best model (with smallest AICc) has two lagged predictors on top of the current value; that is, it includes the manufacturing production index up to the last two months.

```
## Series: lag.train.ts[, "imports"]
## Regression with ARIMA(1,0,0)(1,0,0)[12] errors
##
## Coefficients:
##          ar1    sar1  ProdLag0  ProdLag1  ProdLag2
##       0.8271  0.4191   22.9880    4.1276   -0.0970
## s.e.  0.0637  0.1296    3.8966    3.5009    4.2342
##
## sigma^2 estimated as 79532:  log likelihood=-663.98
## AIC=1339.96   AICc=1340.92   BIC=1355.22
```

This model has AR(1) non-seasonal and seasonal errors.

$$\begin{aligned} y_t &= 22.99x_t + 4.13x_{t-1} - 0.10x_{t-2} + \eta_t \\ (1-0.83B)(1-0.42B^{12})\eta_t &= \varepsilon_t \\ \varepsilon_t &\sim \text{NID}(0, 70532) \end{aligned}$$

where y_t is the total imports and x_t is the manufacturing production index in month t. From the graph of the 12-month forecast below, The model seems to closely forecast the actual values. We also gather the accuracy metrics.




```
##                      ME      RMSE      MAE        MPE     MAPE      MASE
## Training set  19.57072 277.4766 233.4812 0.06493393 6.392350 0.5149374
## Test set     158.13803 302.6782 214.9751 4.63081714 6.216255 0.4741227
```

Part VI: Summary of Forecasting Performance (10%)

We compare the test accuracy of all models below. The best performing model is the dynamic regression of lagged predictor (Manufaturing Production Index) with ARIMA$(1,0,0)(1,0,0)_{12}$ error. This model achieved the lowest RSME, MAE, MAPE and MASE. The second best is ARIMA$(1,1,0)(0,1,1)_{12}$.

For this data set, the most sophisticated model, the one that uses some lagged predictors, got the best performance. This might be attributed to the additional complexity needed to explain the abrupt drop in the total imports between 2008 and 2009, which simpler models, like the benchmark ones or predictor-independent SARIMA, find it hard to capture with.

Summary of Model Performance

	Type	ME	RMSE	MAE	MPE	MAPE	MASE
Mean	Benchmark	-20.64	341.53	275.68	-1.52	7.95	0.65
Naive	Benchmark	282.58	442.8	376.75	7.02	10.09	0.88
Drift	Benchmark	243.1	407.73	347.39	5.95	9.34	0.81
Seasonal Naive	Benchmark	-1136.92	1361.23	1259.42	-32.91	36.11	2.95
ARIMA(1,1,1)(0,1,1)[12]	SARIMA	63.57	335.16	250.84	1.23	7.06	0.59
ARIMA(1,1,0)(0,1,1)[12]	SARIMA	47.95	332.49	244.38	0.79	6.89	0.57
Linear Regression	Dynamic Regression	559.88	600.65	559.88	15.36	15.36	1.23

	Type	ME	RMSE	MAE	MPE	MAPE	MASE
Deterministic	Dynamic Regression	-898.63	948.57	898.63	-25.45	25.45	2.11
Stochastic	Dynamic Regression	396.88	614.33	463.27	10.26	12.38	1.09
Harmonic Regression	Dynamic Regression	-2.39	340.85	255.11	-0.81	7.26	0.6
Lagged Predictors	**Dynamic Regression**	**158.14**	**302.68**	**214.98**	**4.63**	**6.22**	**0.47**