In the file nodx_2017.csv, we have data on Singapore monthly, non- seasonally adjusted domestic exports from Jan 1976 to Dec 2017. In this project, our group will try to forecast the period over Jan 2018 to Dec 2018, using the data up to Dec 2017. The first step is using ARIMA + trend + seasonality model to fit the given time series. Then, we will evaluate our model by forecasting 1-step ahead over the period Jan 2013 to Dec 2017. Finally, we will produce the forecasts for the period Jan 2018 to Dec 2018 and compare to the realized outcomes.

# 1. Model fitting

In this section, we will discuss how to fit an appropriate model for the given time series. Firstly, we import some useful packages and define correl function in order to plot correlogram latter.

```
library(tidyverse)

library(forecast)

library(ggfortify)

library(gridExtra)

library(grid)

library(dynlm)

library(urca)

# Correllogram function

correl <- function(y){
  p1 <- ggAcf(y,30) + scale_y_continuous(limits=c(-1,1)) +
    theme(axis.title.x = element_blank(), plot.title = element_blank())
  p2 <- ggPacf(y, 30) + scale_y_continuous(limits=c(-1,1)) +
    theme(axis.title.x = element_blank(), plot.title = element_blank())
  grid.arrange(p1,p2,ncol=1)
}
```

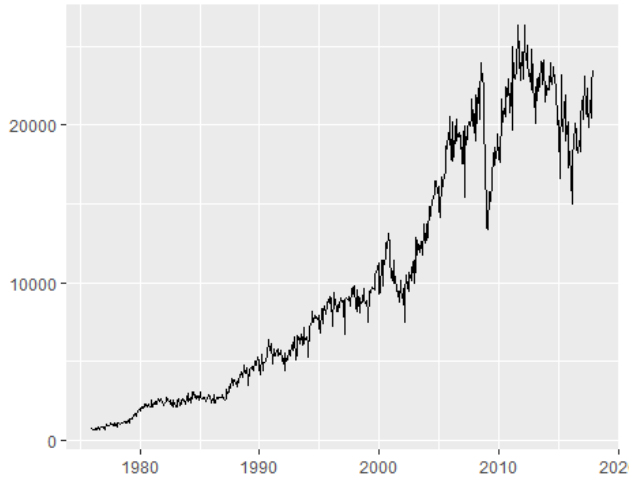
We set up our data by importing the csv file and define the corresponding time series.

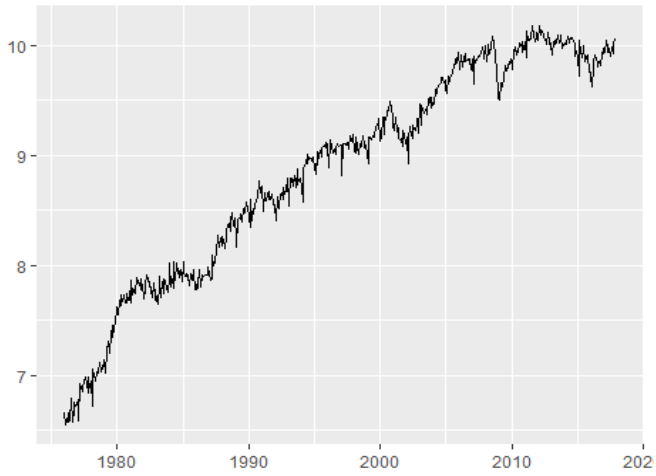
```
#read data
nodx <- read.csv("nodx_2017.csv")
glimpse(nodx)

## Observations: 504
## Variables: 2
## $ Date   <fct> Jan 76, Feb 76, Mar 76, Apr 76, May 76, Jun 76, Jul 76,...
## $ Volume <dbl> 783.226, 700.088, 715.228, 777.443, 720.721, 725.852, 8...

nodx.ts <- ts(nodx$Volume,
              start=c(1976,1), end=c(2017,12), frequency=12) #since it is monthly d
```

```
ata
y <- nodx.ts
```

As the first step, we plot and look at the series to roughly check whether the series is stationary, that is, to see whether the series has a constant mean and variance. We may also apply a log-transformation, since it will reduce the magnitude of the data, and display a pure "growth rate". After we plot the series, we can see that there is a trend in the series.

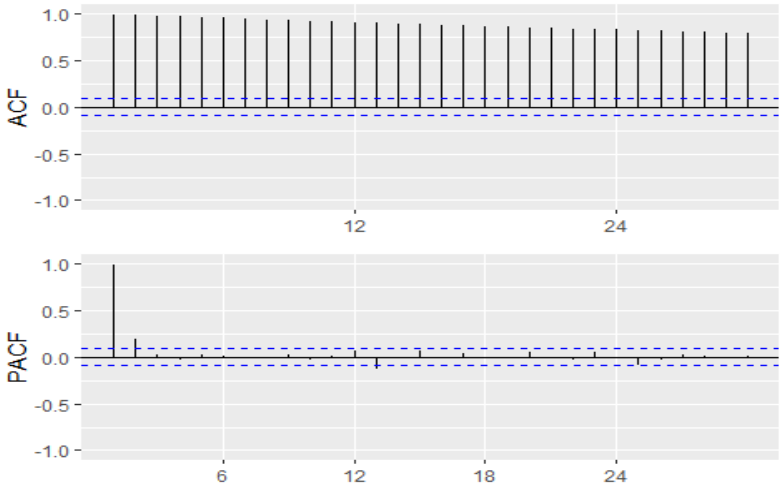
```
p1 <- autoplot(y)
#check the plots
p1

p2 <- autoplot(log(y))
p2
```




Furthermore, we can check the correlogram of the series.

```
correl(log(y))
```



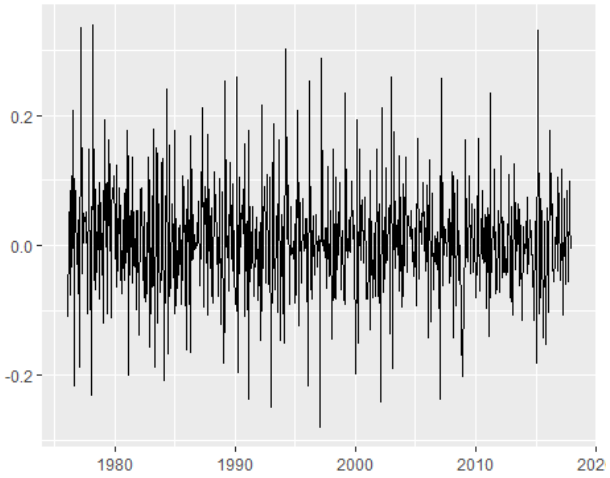
The evidence suggests that the given time series is not stationary. We can confirm our guess by performing KPSS test.

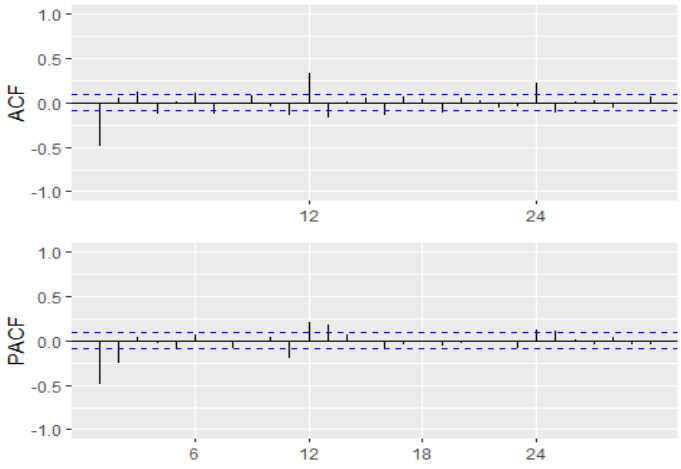
```
#KPSS test
uy <- ur.kpss(log(y), type="tau", lags="short") #since the series has a trend
summary(uy)
```

```
##
## ########################
## # KPSS Unit Root Test #
## ########################
##
## Test is of type: tau with 5 lags.
##
## Value of test-statistic is: 1.2152
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.119 0.146  0.176 0.216
```

The test result shows that the series is not stationary, and we need to do a first order differencing.

```
#check numeber of unit root
autoplot(diff(log(y)))

correl(diff(log(y)))
```




After doing a first order differencing, the series behaves quite like a stationary process. We conduct KPSS test again to see whether we need to apply a second order differencing for the series.

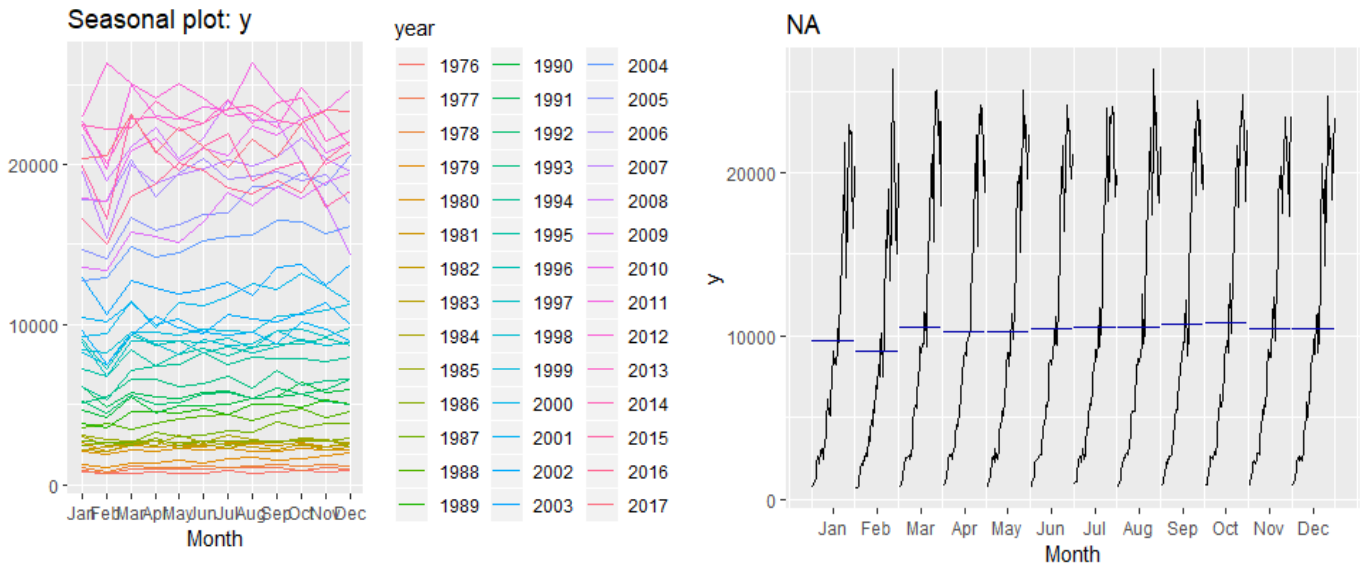
```
#KPSS
ky <- ur.kpss(diff(log(y)), type="mu", lags="short") #after differencing no trend a
nymore
summary(ky)
```

```
##
## ########################
## # KPSS Unit Root Test #
## ########################
##
## Test is of type: mu with 5 lags.
##
## Value of test-statistic is: 0.2257
##
## Critical value for a significance level of:
##                  10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

The test statistic this time becomes insignificant and we conclude that the first order differencing of the series is stationary. By looking at the correlogram of the first-differenced series, the ACF decreases exponentially and the PACF becomes insignificant after the second lag, which indicates the series could be an AR(2) process. Moreover, from the correlogram, we can easily observe a strong seasonality since the correlation repeats every 12 lags.

The seasonality is shown in the following plots.

```
ggseasonplot(y)

ggsubseriesplot(y)
```



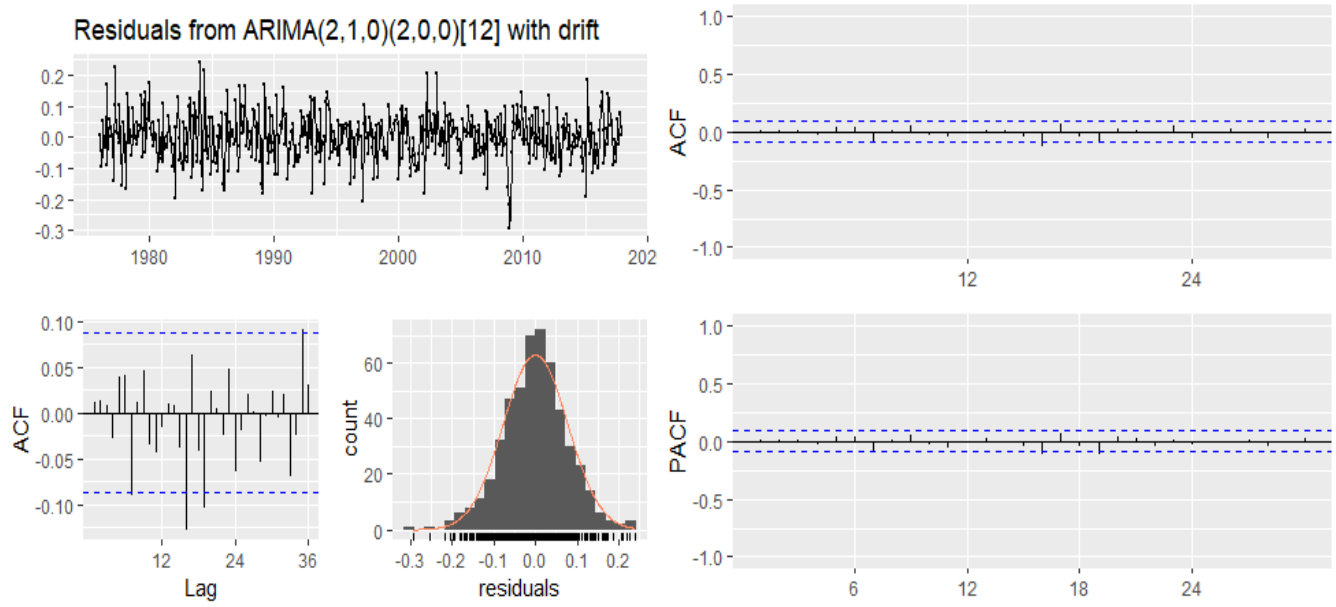

Next, we use auto.arima to help us determine the appropriate lag for ARMA(p,q) as well as SARMA(p,q).

```
fit1 <- auto.arima(y,lambda = 0, biasadj = TRUE, seasonal = T) #apply log-transform
ation and capture seasonality

summary(fit1)
```

```
## Series: y
## ARIMA(2,1,0)(2,0,0)[12] with drift
## Box Cox transformation: lambda= 0
##
## Coefficients:
##           ar1      ar2    sar1    sar2   drift
##       -0.6336  -0.2474  0.2818  0.1647  0.0070
## s.e.   0.0434   0.0433  0.0440  0.0453  0.0032
##
## sigma^2 estimated as 0.006035:  log likelihood=572.75
## AIC=-1133.49   AICc=-1133.32   BIC=-1108.17
##
## Training set error measures:
##                      ME     RMSE      MAE         MPE     MAPE      MASE
## Training set -50.66619 917.4266 582.7413 -0.6212474 5.989983 0.4388017
##                    ACF1
## Training set 0.1386727
```

```
checkresiduals(fit1)
```

```
correl(residuals(fit1))
```




The residual of the model is fairly clean, and therefore an ARIMA(2,1,0)(2,0,0)[12] seems to be a good model for the given series. To confirm our choice and to see whether this model is underfit or overfit, we fit the series using function Arima by changing (p,q) in both ARMA and SARMA.

```
# Define trend and seasonals
trendvar <- seq_along(nodx.ts)
seasonalvars <- seasonaldummy(nodx.ts)
trendseasvars = cbind(trendvar, seasonalvars)
```

```r
#Fitting a model with ARIMA(2,1,0)(2,0,0)[12]
fit2 <- Arima(y, order=c(2,1,0),
              seasonal=c(2,0,0),
              xreg=trendseasvars,
              #include.constant = T,
              lambda=0) # box-cox transformation, lambda=0 is log-transformation
summary(fit2)

sse <- sum(fit2$residuals^2)
print(paste0("SSE is ", as.character(round(sse,2))))

checkresiduals(fit2)

#Fitting a model with ARIMA(3,1,0)(2,0,0)[12]
fit3 <- Arima(y, order=c(3,1,0),
              seasonal=c(2,0,0),
              xreg=trendseasvars,
              #include.constant = T,
              lambda=0) # box-cox transformation, lambda=0 is log-transformation
summary(fit3)

sse <- sum(fit3$residuals^2)
print(paste0("SSE is ", as.character(round(sse,2))))

checkresiduals(fit3)

#Fitting a model with ARIMA(1,1,0)(2,0,0)[12]
fit4 <- Arima(y, order=c(1,1,0),
              seasonal=c(2,0,0),
              xreg=trendseasvars,
              #include.constant = T,
              lambda=0) # box-cox transformation, lambda=0 is log-transformation
summary(fit4)



sse <- sum(fit4$residuals^2)
print(paste0("SSE is ", as.character(round(sse,2))))

checkresiduals(fit4)
```

By comparing the models discussed above (details of output please refer to the appendix), we can see that ARIMA(1,1,0)(2,0,0)[12] doesn't describe the series well. Meanwhile, both ARIMA(2,1,0)(2,0,0)[12] and ARIMA(3,1,0)(2,0,0)[12] can fit the series very well. However, the residual of ARIMA(3,1,0)(2,0,0)[12] is "too clean" and the overall fit of this model doesn't improve much. Hence, by choosing ARIMA(3,1,0)(2,0,0)[12] model, we may face the problem of overfitting. By similar process, we can show that ARIMA(2,1,0)(2,0,0)[12] is a better choice comparing to ARIMA(2,1,0)(1,0,0)[12] and ARIMA(2,1,0)(3,0,0)[12]. Therefore, we conclude that ARIMA(2,1,0)(2,0,0)[12] is the best appropriate model for us to fit the given series.

## 2. Model evaluation

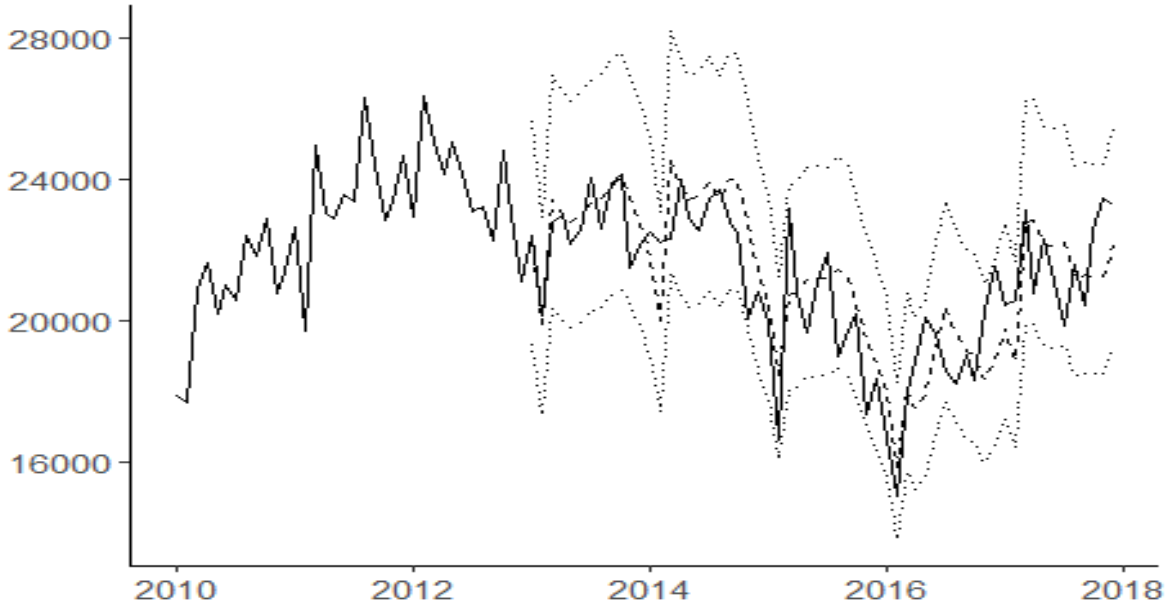
In this section, we will discuss how to evaluate our fitted model. We firstly get the in-sample $R^2$ by following code.

```
sse <- sum(fit2$residuals^2)
sst <- sum((fit2$x - mean(fit2$x))^2)
Rsqr <- 1 - sse/sst
print(paste0("R-sqr is ", as.character(round(Rsqr,2))))
```

```
## [1] "R-sqr is 1"
```

The result shows the in-sample $R^2$ is 1, due to the huge SST. This also indicates that in-sample $R^2$ is not a good criterion to evaluate our model. To evaluate the forecasting ability of our model, we will suppose we are at the end of 2012, and generate a one-step ahead forecast over Jan 2013 to Dec 2017 using our fitted model without any information about the realization from 2013 onwards.

```
# Theme settings for figures:
ts_thm <- theme(text = element_text(size=14),
                axis.title = element_text(size=11),
                axis.line = element_line(linetype = 'solid'),
                panel.background = element_blank())

# Jan 1976 to Dec 2012 is observation 1:444
# Jan 2013 to Dec 2017 is observation 445 to 504 (60 observations)
fcst1step<-ts(matrix(rep(NA,60*3),ncol=3), start=c(2013,1), frequency=12) # to stor
e forecasts
colnames(fcst1step) <- c("mean", "lower", "upper")
for (i in 1:60){
  fit <- Arima(y[1:(444+i-1)],
               order=c(2,1,0),
               seasonal=c(2,0,0),
               xreg = trendseasvars[1:(444+i-1),],
               #include.constant = T,
               lambda=0)
  temp <- forecast(fit, h=1, xreg=matrix(trendseasvars[(444+i):(444+i),], nrow=1))
  fcst1step[i,]<-cbind(temp$mean, temp$lower[,"95%"], temp$upper[,"95%"])
  }

ts.plot <- ts.union(Actual=window(y,start=c(2010,1)),fcst1step)
autoplot(ts.plot) +
  scale_color_manual(values=rep("black", 4)) +
  ylab("") + xlab("") +
  aes(linetype=series) +
  scale_linetype_manual(values=c("solid", "dashed", rep("dotted",2))) +
  ts_thm + theme(legend.position="none")
```




```
f2err <- ts.union(Actual=window(y,start=c(2013,1)),Fcst=fcst1step[,1])
sse <- sum((f2err[,"Actual"]-f2err[,"Fcst"])^2)
sst <- sum((f2err[,"Actual"]-mean(f2err[,"Actual"]))^2)
OOSR2 <- 1-sse/sst
print(paste0("Out-of-sample RMSE is ",as.character(round(sqrt(sse/60),2))))
```

```
## [1] "Out-of-sample RMSE is 1321.02"
```

```
print(paste0("Out-of-sample R-sqr is ",as.character(round(OOSR2,2))))
```

```
## [1] "Out-of-sample R-sqr is 0.6"
```

As the result shows, the Out-of-sample RMSE is 1321.02 and the Out-of-sample R-sqr is 0.6, which means that we managed to forecast 60% of the variation in the target variable.

Moreover, we can apply "Mincer-Zarnowitz" equation to check for optimality. Since in the optimal forecast, the forecast error should be just "White noise", following "Mincer-Zarnowitz" equation $\epsilon_{T+i+1|T+i} = \beta_0 + \beta_1 Y_{T+i+1|T+i} + u_{T+i+1} \Rightarrow Y_{T+i+1} = \beta_0 + \gamma_1 Y_{T+i+1|T+i} + u_{T+i+1}$, we have $H_0: \beta_0 = 0$ and $\gamma_1 = 1$.

```
fit5 <- dynlm(f2err[,"Actual"] ~ f2err[,"Fcst"])
summary(fit5)
```

```
##
## Time series regression with "ts" data:
## Start = 2013(1), End = 2017(12)
##
## Call:
## dynlm(formula = f2err[, "Actual"] ~ f2err[, "Fcst"])
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2241.52  -818.42    80.16   836.56  2599.06
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.570e+03  1.725e+03   2.069    0.043 *
## f2err[, "Fcst"] 8.208e-01  8.073e-02  10.167 1.67e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1268 on 58 degrees of freedom
## Multiple R-squared:  0.6406, Adjusted R-squared:  0.6344
## F-statistic: 103.4 on 1 and 58 DF,  p-value: 1.674e-14
```

As the test summary shows, $\beta_0$ is 0.00357 close to 0, and $\gamma_1$ is 0.8 close to 1. Both are significant. Therefore, such a result may suggest that our model forecasted well over the period Jan 2013 to Dec 2017.

## 3. Forecast

Since we only use data up to Dec 2017, we produce the forecast using 1 to 12 steps ahead forecasting.

```
#read data
dx <- read.csv("dx_2018.csv")
dx.ts <- ts(dx$Volume,
              start=c(1976,1), end=c(2018,12), frequency=12) #since it is monthly d
ata
y1 <- dx.ts

# Define trend and seasonals
trendvar1 <- seq_along(dx.ts)
seasonalvars1 <- seasonaldummy(dx.ts)
trendseasvars1 = cbind(trendvar1, seasonalvars1)

# forecast 1 to 12 steps (1 years x 12 months) ahead
fcst <- forecast(fit2, h=12,
                 xreg=trendseasvars1[505:516,],
                 biasadj=T)

fcst.plot <- ts.union(fcst$mean, fcst$lower[,"80%"], fcst$upper[,"80%"])
ts.plot <- ts.union(Actual=window(y1,start=c(2010,1)),fcst.plot)
autoplot(ts.plot) +
  scale_color_manual(values=rep("black", 6)) +
  ylab("") + xlab("") +
  aes(linetype=series) +
  scale_linetype_manual(values=c("solid", "dashed", rep("dotted",4))) +
  ts_thm + theme(legend.position="none")
```


```
f2err <- ts.union(Actual=window(y1,start=c(2018,1)),Fcst=fcst$mean)
sse <- sum((f2err[,"Actual"]-f2err[,"Fcst"])^2)
sst <- sum((f2err[,"Actual"]-mean(f2err[,"Actual"]))^2)
OOSR2 <- 1-sse/sst
print(paste0("Out-of-sample RMSE is ",as.character(round(sqrt(sse/12),2))))
```

```
## [1] "Out-of-sample RMSE is 1723.33"
```

```
print(paste0("Out-of-sample R-sqr is ",as.character(round(OOSR2,2))))
```

```
## [1] "Out-of-sample R-sqr is -0.07"
```

In the 1 to 12 steps ahead forecasting, the performance of forecasts is not as good as 1 step ahead forecasts, since we do not have newest information to update our data set.

# Appendix

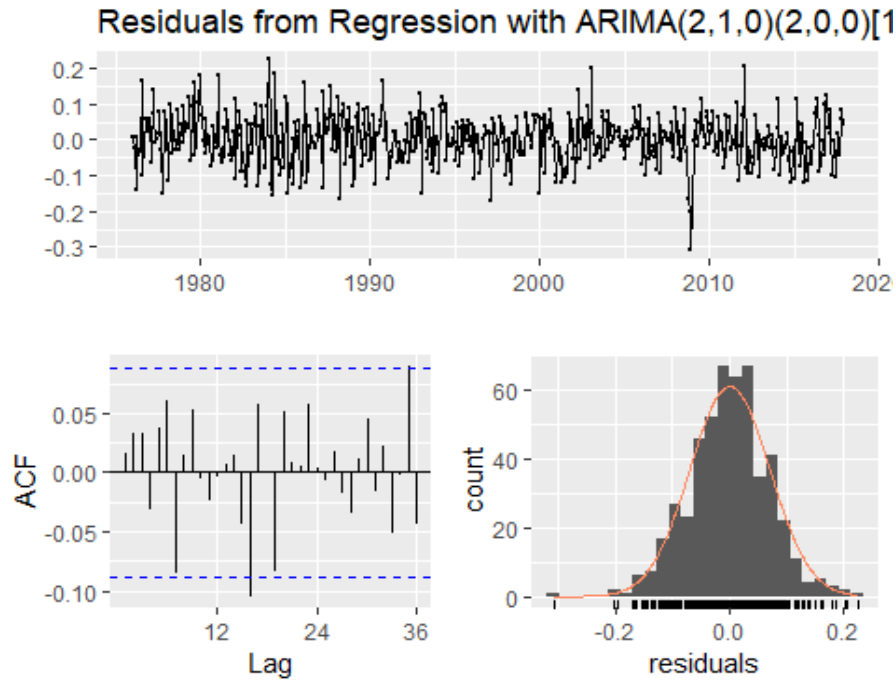
```r
# Define trend and seasonals
trendvar <- seq_along(nodx.ts)
seasonalvars <- seasonaldummy(nodx.ts)
trendseasvars = cbind(trendvar, seasonalvars)

#Fitting a model with ARIMA(2,1,0)(2,0,0)[12]
fit2 <- Arima(y, order=c(2,1,0),
              seasonal=c(2,0,0),
              xreg=trendseasvars,
              #include.constant = T,
              lambda=0) # box-cox transformation, lambda=0 is log-transformation
summary(fit2)
```

```
## Series: y
## Regression with ARIMA(2,1,0)(2,0,0)[12] errors
## Box Cox transformation: lambda= 0
##
## Coefficients:
##           ar1      ar2    sar1     sar2  trendvar      Jan      Feb
##       -0.6292  -0.2295  0.0614  -0.0521    0.0067  -0.0247  -0.1186
## s.e.   0.0436   0.0436  0.0452   0.0460    0.0017   0.0130   0.0125
##          Mar     Apr      May     Jun     Jul     Aug     Sep     Oct
##       0.0301  0.0077  -0.0002  0.0092  0.0241  0.0143  0.0277  0.0348
## s.e.  0.0130  0.0140   0.0141  0.0142  0.0140  0.0139  0.0129  0.0124
##          Nov
##       0.0051
## s.e.  0.0129
##
## sigma^2 estimated as 0.00503:  log likelihood=625.16
## AIC=-1216.31   AICc=-1215.05   BIC=-1144.56
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -37.7831 840.5233 518.1944 -0.2498418 5.418236 0.3901982
##                   ACF1
## Training set 0.1648382
```

```r
sse <- sum(fit2$residuals^2)
print(paste0("SSE is ", as.character(round(sse,2))))
```

```
## [1] "SSE is 2.45"
```

```r
checkresiduals(fit2)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,1,0)(2,0,0)[12] errors
## Q* = 25.286, df = 8, p-value = 0.00139
##
## Model df: 16.   Total lags used: 24
```

```
#Fitting a model with ARIMA(3,1,0)(2,0,0)[12]
fit3 <- Arima(y, order=c(3,1,0),
              seasonal=c(2,0,0),
              xreg=trendseasvars,
              #include.constant = T,
              lambda=0) # box-cox transformation, lambda=0 is log-transformation
summary(fit3)
```

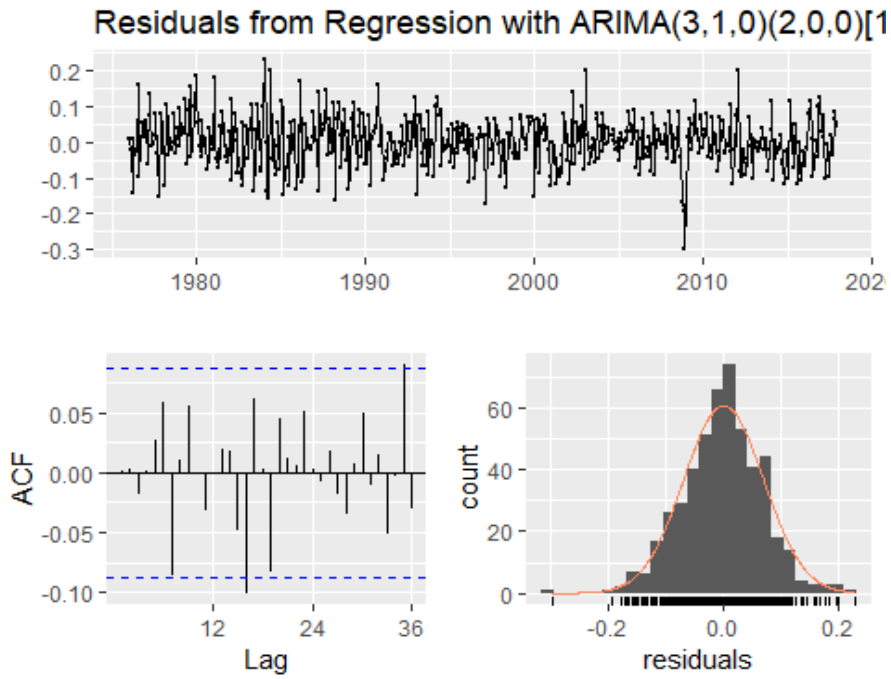
```
## Series: y
## Regression with ARIMA(3,1,0)(2,0,0)[12] errors
## Box Cox transformation: lambda= 0
##
## Coefficients:
##           ar1      ar2     ar3    sar1     sar2  trendvar      Jan
##       -0.6136  -0.1859  0.0693  0.0570  -0.0507    0.0067  -0.0245
## s.e.   0.0446   0.0518  0.0448  0.0452   0.0459    0.0018   0.0129
##           Feb     Mar     Apr     May     Jun     Jul     Aug     Sep
##       -0.1184  0.0302  0.0078  0.0000  0.0094  0.0242  0.0145  0.0278
## s.e.   0.0124  0.0128  0.0142  0.0141  0.0143  0.0141  0.0142  0.0128
##          Oct     Nov
##       0.0349  0.0054
```

```
## s.e.  0.0123  0.0128
##
## sigma^2 estimated as 0.005017:  log likelihood=626.35
## AIC=-1216.7   AICc=-1215.29   BIC=-1140.73
##
## Training set error measures:
##                     ME     RMSE      MAE         MPE     MAPE      MASE
## Training set -35.3284 833.1542 514.8088 -0.2469592 5.399756 0.3876489
##                    ACF1
## Training set 0.1490682
```

```
sse <- sum(fit3$residuals^2)
print(paste0("SSE is ", as.character(round(sse,2))))
```

```
## [1] "SSE is 2.44"
```

```
checkresiduals(fit3)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(3,1,0)(2,0,0)[12] errors
## Q* = 23.384, df = 7, p-value = 0.001461
##
## Model df: 17.   Total lags used: 24
```

```
#Fitting a model with ARIMA(1,1,0)(2,0,0)[12]
fit4 <- Arima(y, order=c(1,1,0),
              seasonal=c(2,0,0),
              xreg=trendseasvars,
```

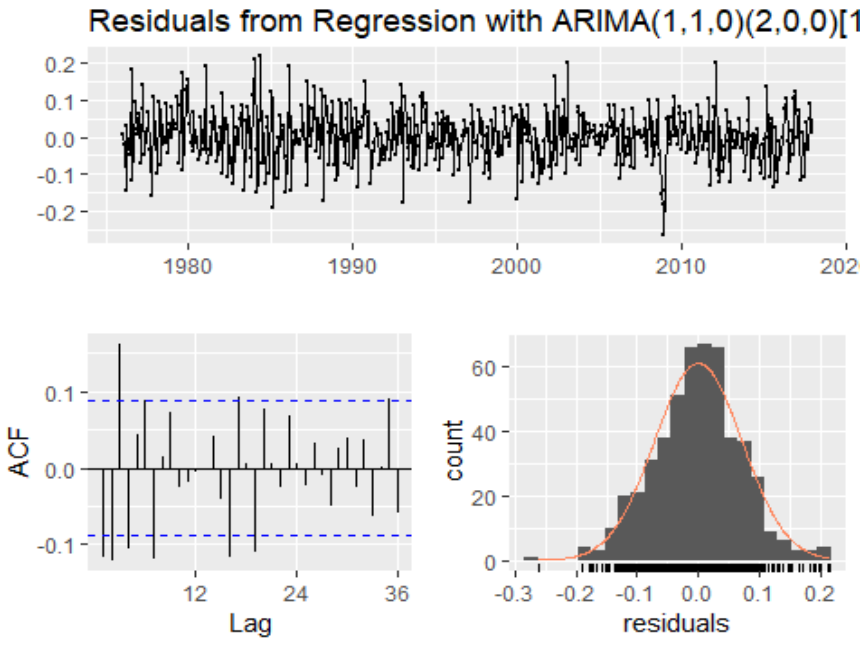
```
              #include.constant = T,
              lambda=0) # box-cox transformation, lambda=0 is log-transformation
summary(fit4)
```

```
## Series: y
## Regression with ARIMA(1,1,0)(2,0,0)[12] errors
## Box Cox transformation: lambda= 0
##
## Coefficients:
##           ar1    sar1     sar2  trendvar      Jan      Feb     Mar     Apr
##       -0.5107  0.0692  -0.0657    0.0067  -0.0252  -0.1192  0.0295  0.0071
## s.e.   0.0383  0.0453   0.0458    0.0021   0.0128   0.0121  0.0144  0.0147
##           May     Jun     Jul     Aug     Sep     Oct     Nov
##       -0.0008  0.0087  0.0236  0.0139  0.0274  0.0344  0.0047
## s.e.   0.0154  0.0153  0.0154  0.0146  0.0144  0.0121  0.0128
##
## sigma^2 estimated as 0.005297:  log likelihood=611.68
## AIC=-1191.36   AICc=-1190.24   BIC=-1123.83
##
## Training set error measures:
##                       ME    RMSE      MAE         MPE     MAPE      MASE
## Training set -31.07747 854.267 532.3994 -0.2561961 5.584296 0.4008945
##                       ACF1
## Training set -0.004393644
```

```
sse <- sum(fit4$residuals^2)
print(paste0("SSE is ", as.character(round(sse,2))))
```

```
## [1] "SSE is 2.58"
```

```
checkresiduals(fit4)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(1,1,0)(2,0,0)[12] errors
## Q* = 74.246, df = 9, p-value = 2.226e-12
##
## Model df: 15.   Total lags used: 24
```