An Application of ARIMA: Shampoo

Logistics and Supply Chain Analytics

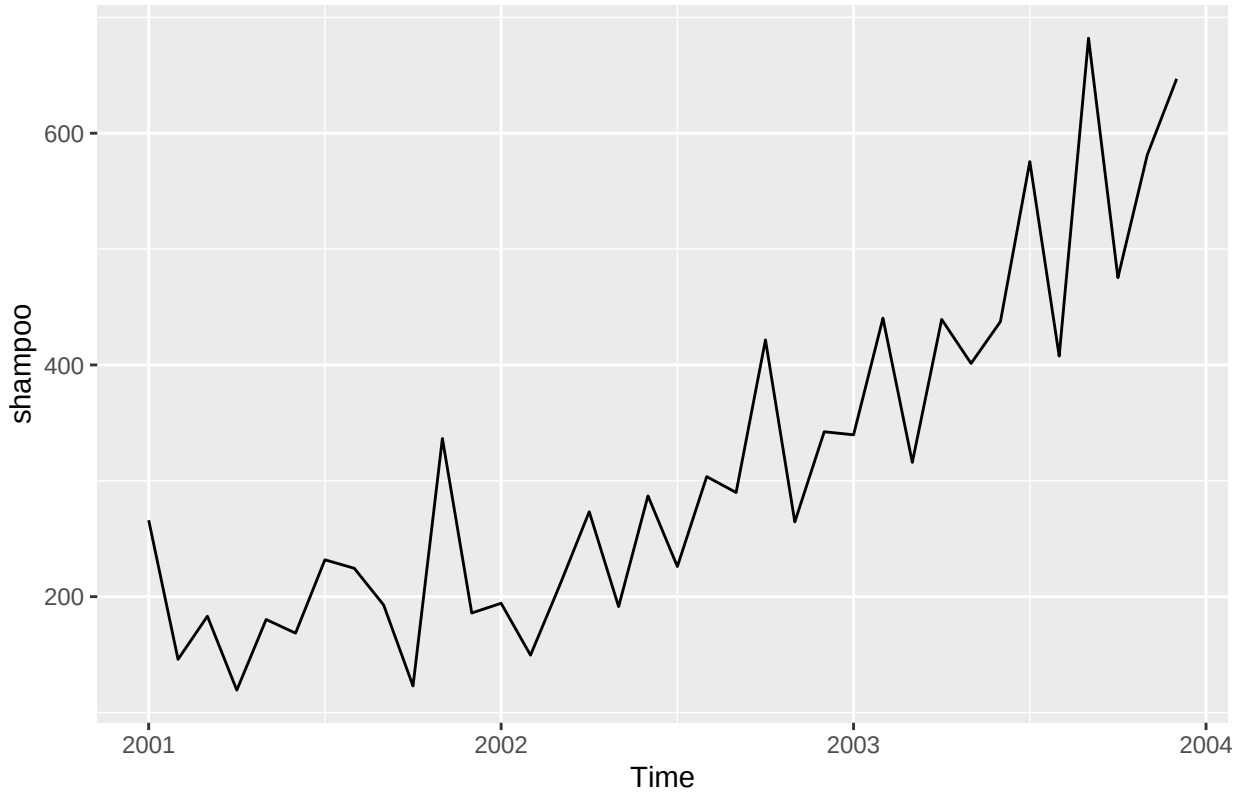
Jiahua Wu

```
data <- read.csv(file = "shampoo.csv", header = TRUE)
shampoo <- ts(data[, 2], frequency = 12, start = c(2001, 1))
```

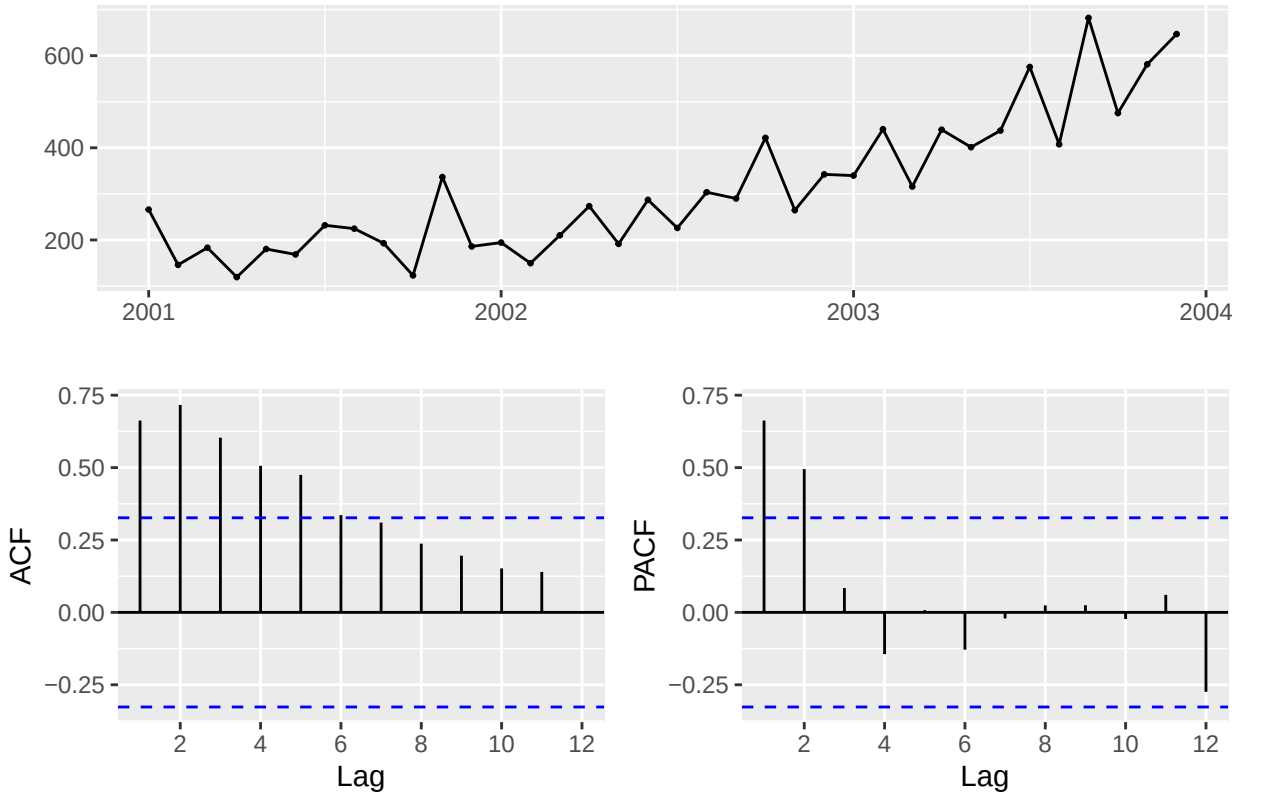
We have a dataset, which includes monthly sales for shampoo at a retailer store from January 2001 to December 2003 (shampoo.csv). Each observation includes two values: month-year pair, and sales in that particular month. We first convert sales into a time series object. *frequency* indicates the number of observations per unit of time. "The value of argument *frequency* is used when the series is sampled an integral number of times in each unit time interval. For example, one could use a value of 7 for *frequency* when the data are sampled daily, and the natural time period is a week, or 12 when the data are sampled monthly and the natural time period is a year." (source: https://stat.ethz.ch/R-manual/R-devel/library/stats/html/ts.html)

With time series analysis, the first step is always to visually inspect it. A couple of functions, see below, are readily available in R. Judging from the plots, it is clear that there is an upward trend in the sales, but there is no clear evidence of seasonality.

```
# plot.ts(shampoo)
autoplot(shampoo)
```

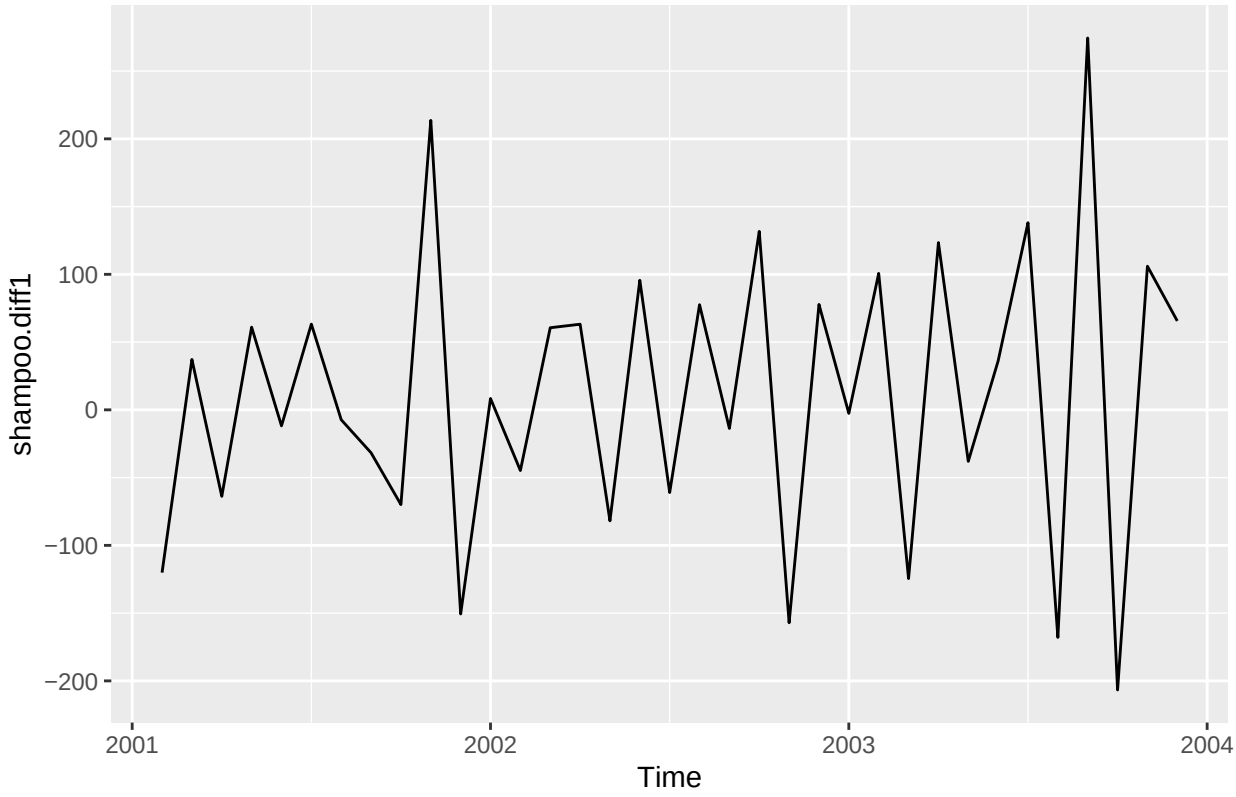


```
ggtsdisplay(shampoo)
```



Due to the trend in time series, it is non-stationary. We can get rid of trend by taking first-order difference. We plot the time series after the difference, and observe that there is no trend and appears to be stationary. We run ADF, PP and KPSS tests to formally test the stationarity of time series after the first-order difference, and all suggest that the time series is stationary.

```
### stationarize time series
# take first order difference
shampoo.diff1 <- diff(shampoo, differences = 1)
autoplot(shampoo.diff1)
```




```
# stationary test
adf.test(shampoo.diff1)
```

```
## Warning in adf.test(shampoo.diff1): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  shampoo.diff1
## Dickey-Fuller = -5.9245, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

```
pp.test(shampoo.diff1)
```

```
## Warning in pp.test(shampoo.diff1): p-value smaller than printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  shampoo.diff1
## Dickey-Fuller Z(alpha) = -54.269, Truncation lag parameter = 3, p-value
## = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(shampoo.diff1)
```

```
## Warning in kpss.test(shampoo.diff1): p-value greater than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  shampoo.diff1
```

```
## KPSS Level = 0.32152, Truncation lag parameter = 3, p-value = 0.1
```

There are also two automatic functions, $ndiffs()$ and $nsdiffs()$ that would tell us how many first-order differences, and how many seasonal differences, respectively, that we need to take to make the time series stationary. By default, $ndiffs()$ is based on KPSS test.

```
ndiffs(shampoo)
```

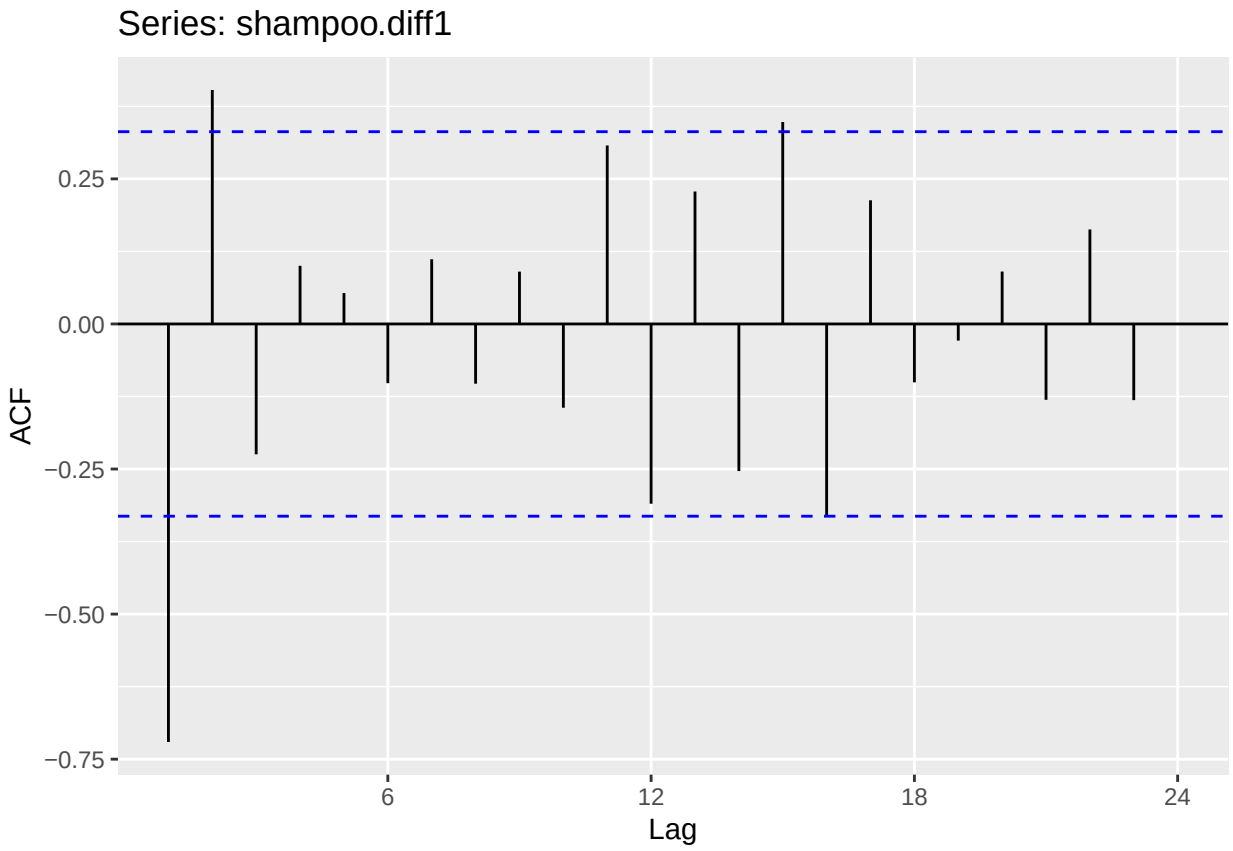
```
## [1] 1
```

```
# seasonal stationarity
nsdiffs(shampoo)
```

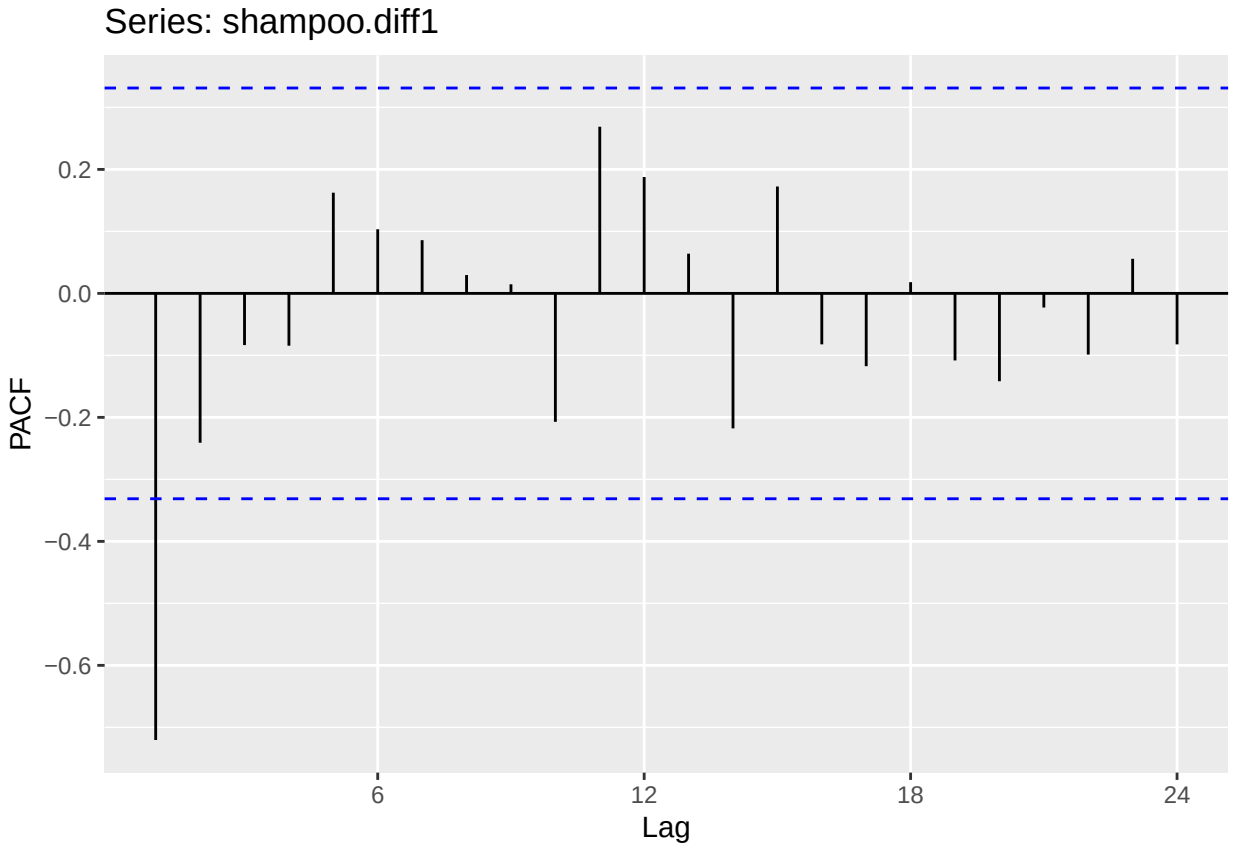
```
## [1] 0
```

Once we have a stationary time series, the next step is to determine the optimal orders of MA and AR components. We first plot the ACF and PACF of the time series. Based on the plots, we know that the order of MA is no greater than 2 (as the first two lags of ACF are significant), and the order of AR is no greater than 1 (as only the first lag of PACF is significant).

```
### determine optimal p and q
# acf plot
# acf(shampoo.diff1, lag.max = 20) # q <= 2
ggAcf(shampoo.diff1)
```




```
# pacf plot
# pacf(shampoo.diff1, lag.max = 20) # p <= 1
ggPacf(shampoo.diff1)
```



Next we use *auto.arima*() to search for the best ARIMA models. Potentially, we can use our earlier analysis as input to refine the models that will be evaluated by *auto.arima*(). But for this simple exercise, we just use the default setting from *auto.arima*(), and run a stepwise search.

```
# choose optimal p and q based on information criteria
auto.arima(shampoo, trace = TRUE, ic = 'bic')
```

```
##
##  ARIMA(2,1,2)(1,0,1)[12] with drift         : Inf
##  ARIMA(0,1,0)            with drift         : 433.9838
##  ARIMA(1,1,0)(1,0,0)[12] with drift         : 410.9315
##  ARIMA(0,1,1)(0,0,1)[12] with drift         : Inf
##  ARIMA(0,1,0)                               : 430.7841
##  ARIMA(1,1,0)            with drift         : 410.6883
##  ARIMA(1,1,0)(0,0,1)[12] with drift         : 409.1295
##  ARIMA(1,1,0)(1,0,1)[12] with drift         : Inf
##  ARIMA(0,1,0)(0,0,1)[12] with drift         : Inf
##  ARIMA(2,1,0)(0,0,1)[12] with drift         : Inf
##  ARIMA(1,1,1)(0,0,1)[12] with drift         : 407.3023
##  ARIMA(1,1,1)            with drift         : 408.688
##  ARIMA(1,1,1)(1,0,1)[12] with drift         : Inf
##  ARIMA(1,1,1)(1,0,0)[12] with drift         : 408.7365
##  ARIMA(2,1,1)(0,0,1)[12] with drift         : Inf
##  ARIMA(1,1,2)(0,0,1)[12] with drift         : 410.1852
##  ARIMA(0,1,2)(0,0,1)[12] with drift         : Inf
##  ARIMA(2,1,2)(0,0,1)[12] with drift         : Inf
##  ARIMA(1,1,1)(0,0,1)[12]                    : 413.8951
##
```

```
##  Best model: ARIMA(1,1,1)(0,0,1)[12] with drift
```

```
## Series: shampoo
## ARIMA(1,1,1)(0,0,1)[12] with drift
##
## Coefficients:
##           ar1      ma1     sma1    drift
##       -0.5479  -0.4823  -0.7441  12.8027
## s.e.   0.1736   0.1753   0.6477   2.4503
##
## sigma^2 estimated as 3435:  log likelihood=-194.76
## AIC=399.53   AICc=401.59   BIC=407.3
```

```
# Best model: ARIMA(1,1,1)(0,0,1)[12] with drift (BIC=407.3)
# Second best: ARIMA(1,1,1) with drift (BIC=408.7)
```

Based on the output of *auto.arima*(), a couple of models have similar BICs. Now suppose that we choose the two models with the lowest BICs, namely ARIMA(1,1,1)(0,0,1)[12] with drift and ARIMA(1,1,1) with drift, as the candidate models that we would like to evaluate further. The only difference between the two models is the seasonal MA part. Recall that our preliminary analysis with ACF and PACF plots does not reveal any seasonal factors, so we will have to determine whether or not to include the seasonal MA part in the model.

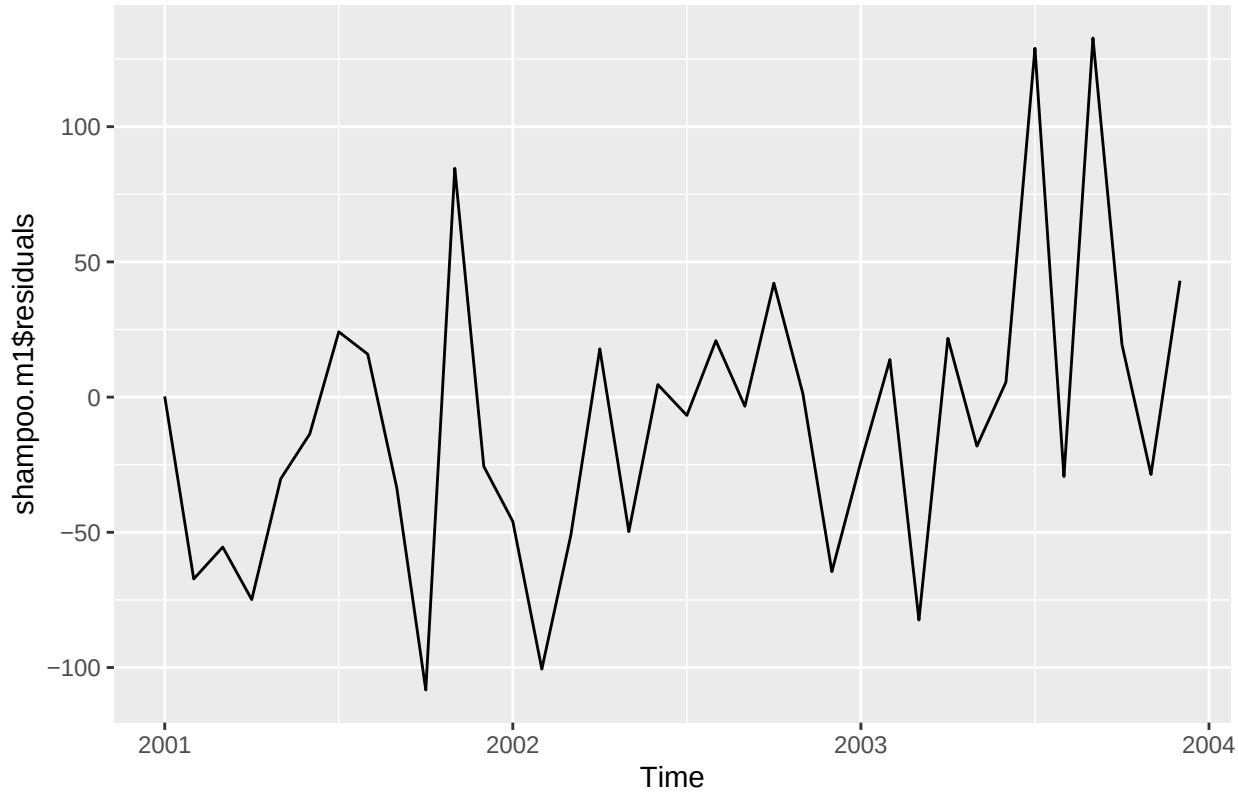
```
# two candidate models
shampoo.m1 <- Arima(shampoo, order = c(1, 1, 1),
                    seasonal = list(order = c(0, 0, 1), period = 12), include.drift = TRUE)
shampoo.m2 <- Arima(shampoo, order = c(1, 1, 1), include.drift = TRUE)
```

A couple of functions are proven to be useful to evaluate the in-sample performance/fit of the model. One is *accuracy*() function, which summarizes various measures of fitting errors. In the post-estimation analysis, we would also like to check out the residual plots, including time series plot, ACF and etc., to make sure that there is no warning signal. In particular, residuals shall have a zero mean, constant variance, and distributed symmetrically around mean zero. ACF of any lag greater 0 is expected to be statistically insignificant.

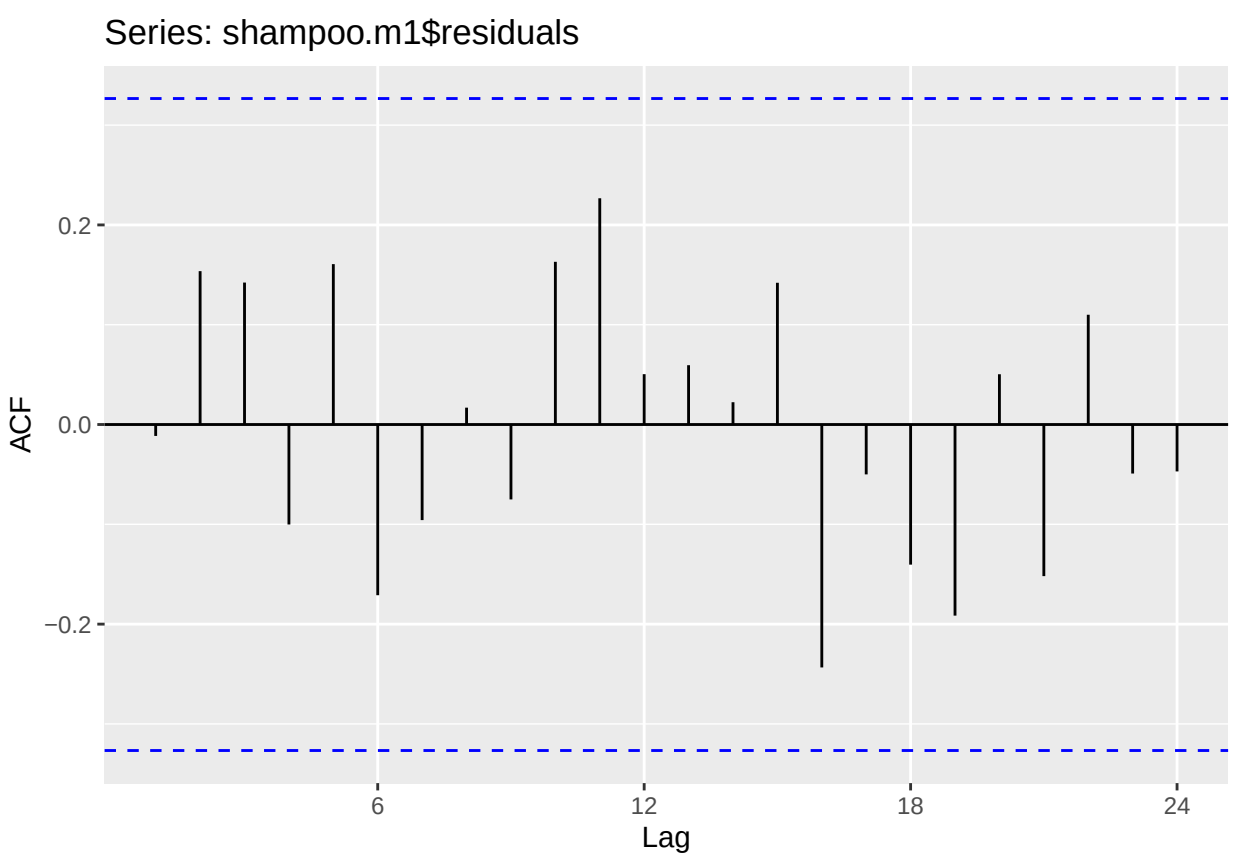
```
# in-sample one-step forecasts
accuracy(shampoo.m1)
```

```
##                      ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -9.343786 54.38305 41.40145 -10.19758 17.43117 0.269709
##                      ACF1
## Training set -0.01143662
```

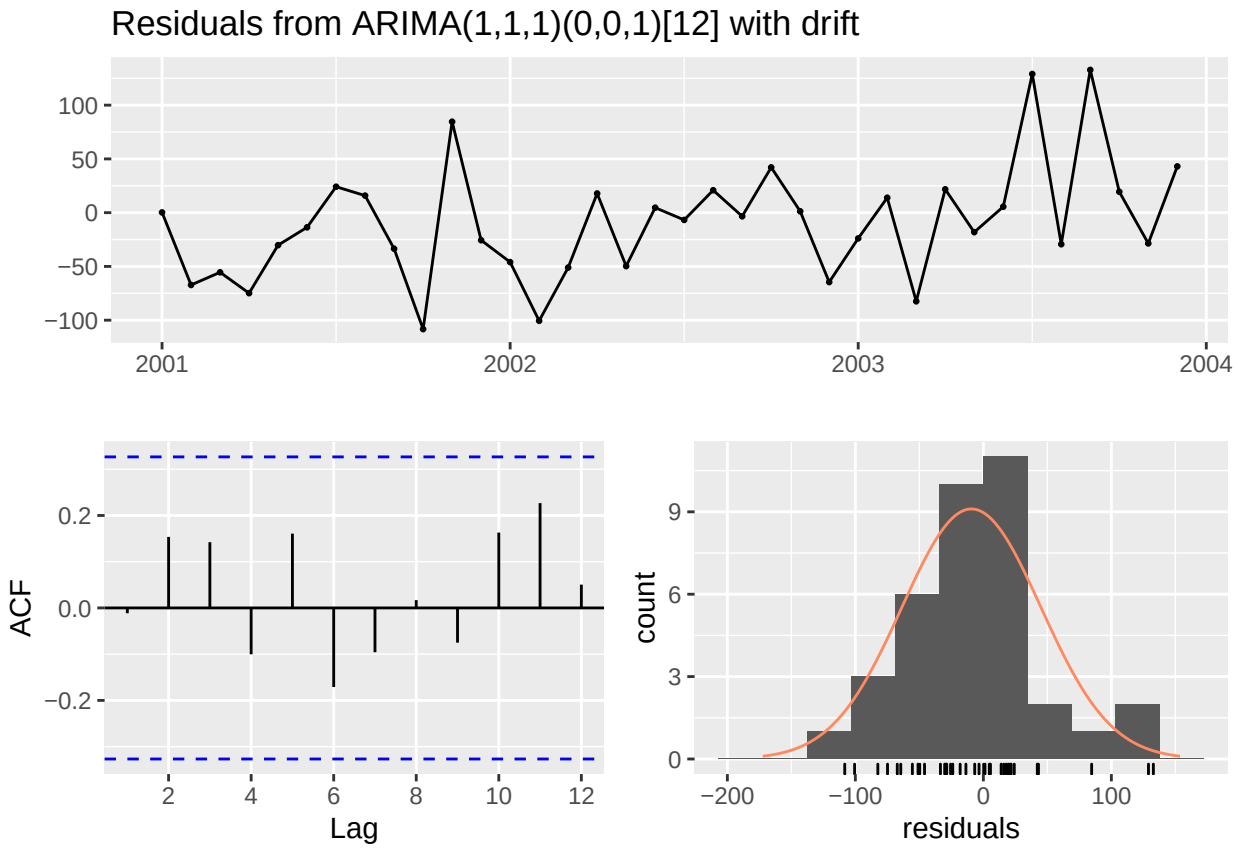
```
# residual analysis
autoplot(shampoo.m1$residuals)
```




```
ggAcf(shampoo.m1$residuals)
```



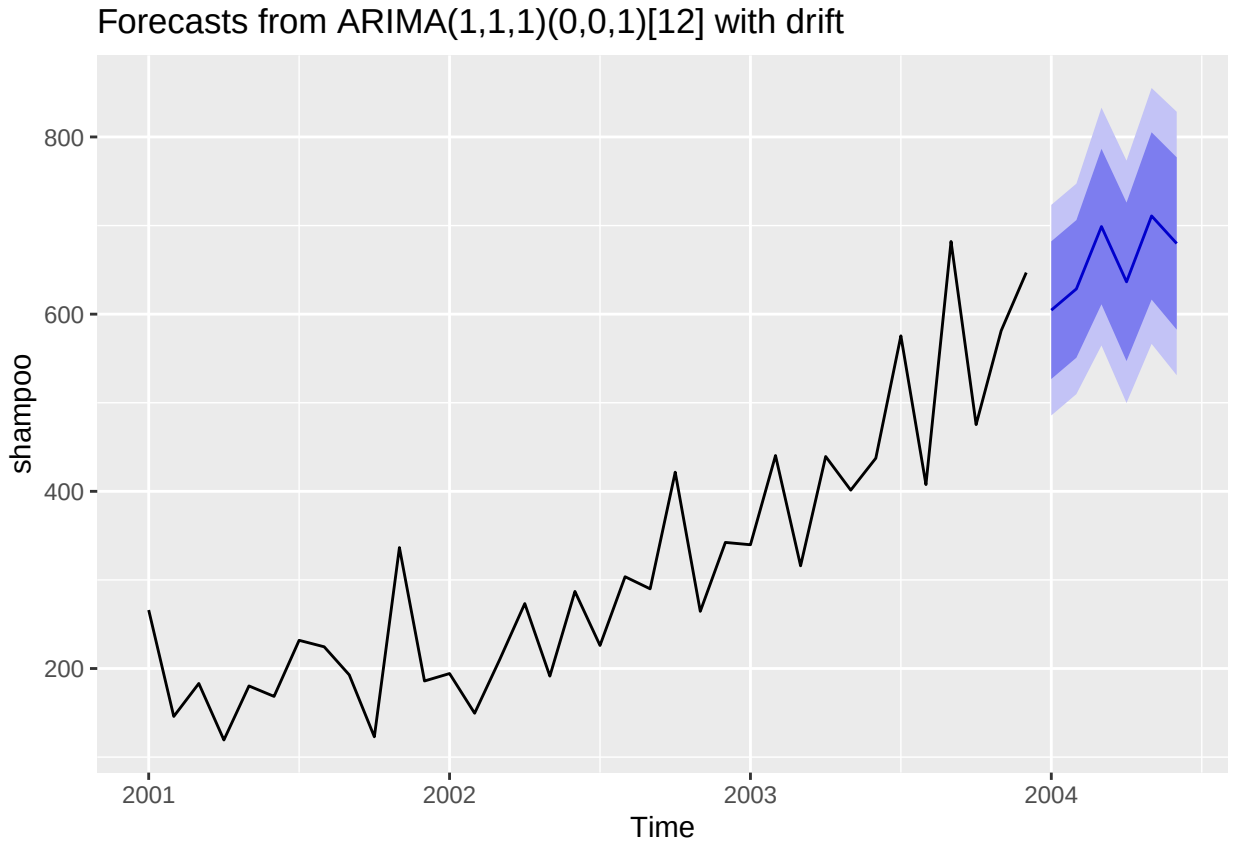
```
checkresiduals(shampoo.m1)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,1)(0,0,1)[12] with drift
## Q* = 5.1294, df = 3, p-value = 0.1626
##
## Model df: 4.   Total lags used: 7
```

Once we have selected a model, forecasting part is easy - it is implemented by *forecast*() function from the forecast package.

```
# forecast
shampoo.f <- forecast(shampoo.m1, h = 6)
autoplot(shampoo.f)
```



Lastly, let us discuss how we can evaluate the out-of-sample performance of ARIMA models. First, we need to divide the data into two sets: training set and test set. For time series analysis, we divide data based on time indices of observations. Earlier observations are used for training, and more recent observations are used for testing.

Suppose we use the first two-and-half year of data for training. Based on *auto.arima*(), we choose two candidate models with the lowest BICs (here we skipped the steps on stationarity tests, ACF and PACF plots, and etc.; but they are definitely important for the analysis).

```
### model evaluation
# Fit model with first 2.5-year of data
auto.arima(window(shampoo, end = c(2003, 6)), d = 1, trace = TRUE, ic = 'bic')
```

```
##
##  ARIMA(2,1,2)            with drift         : 337.6606
##  ARIMA(0,1,0)            with drift         : 349.1833
##  ARIMA(1,1,0)            with drift         : 333.9026
##  ARIMA(0,1,1)            with drift         : 332.9748
##  ARIMA(0,1,0)                               : 345.9445
##  ARIMA(1,1,1)            with drift         : 332.667
##  ARIMA(2,1,1)            with drift         : 335.9243
##  ARIMA(1,1,2)            with drift         : 334.3087
##  ARIMA(0,1,2)            with drift         : 332.4953
##  ARIMA(0,1,3)            with drift         : 335.848
##  ARIMA(1,1,3)            with drift         : 337.6186
##  ARIMA(0,1,2)                               : 332.9803
##
##  Best model: ARIMA(0,1,2)            with drift
```

```
## Series: window(shampoo, end = c(2003, 6))
## ARIMA(0,1,2) with drift
##
## Coefficients:
##           ma1     ma2   drift
##       -1.1532  0.4352  8.6419
## s.e.   0.1951  0.1991  3.1521
##
## sigma^2 estimated as 3722:  log likelihood=-159.51
## AIC=327.03   AICc=328.69   BIC=332.5
```

```
# two candidate models: ARIMA(0,1,2) with drift, ARIMA(1,1,1) with drift
m1 <- Arima(window(shampoo, end = c(2003, 6)), order = c(1, 1, 1), include.drift = TRUE)
m2 <- Arima(window(shampoo, end = c(2003, 6)), order = c(0, 1, 2), include.drift = TRUE)
```

There are two different ways to evaluate out-of-sample performance of models: one is based on one-step ahead forecast, and the other based on multi-step ahead forecast. With one-step ahead forecast, at time period t (t is from June 2003 to Nov 2003), we generate a forecast of sales in time period $t+1$, and compare it against real sales data. *accuracy*() function summarizes various measures of one-step ahead forecasting errors.

```
# Apply fitted model to later data
m1.f <- Arima(window(shampoo, start = c(2003, 7)), model = m1)
m2.f <- Arima(window(shampoo, start = c(2003, 7)), model = m2)

# out-of-sample one-step ahead forecasts
accuracy(m1.f)
```

```
##                     ME     RMSE     MAE        MPE     MAPE MASE       ACF1
## Training set 11.71466 79.46943 61.1882 -0.3220726 11.25665  NaN -0.4886267
```

```
accuracy(m2.f)
```

```
##                     ME     RMSE      MAE        MPE     MAPE MASE       ACF1
## Training set 10.23239 77.45575 60.33499 -0.6108842 11.22599  NaN -0.5231363
```

Alternatively, in June 2003, we generate forecasts of monthly sales for each month from July 2003 to Dec 2003. Then again, we use *accuracy*() to report various measures of forecasting errors across this 6-month period.

```
# out-of-sample multi-step ahead forecasts
accuracy(forecast(m1, h = 6), window(shampoo, start = c(2003, 7)))
```

```
##                      ME      RMSE       MAE       MPE     MAPE      MASE
## Training set  -5.309904  57.14505  45.87963 -9.950254 21.76062 0.3817294
## Test set     114.664852 147.28821 124.98553 18.004335 20.53640 1.0399092
##                    ACF1 Theil's U
## Training set  0.08956709        NA
## Test set     -0.70731992 0.8573852
```

```
accuracy(forecast(m2, h = 6), window(shampoo, start = c(2003, 7)))
```

```
##                      ME      RMSE       MAE       MPE     MAPE      MASE
## Training set  -4.677904  56.79416  45.75745 -9.574285 21.79676 0.3807128
## Test set     108.864315 142.05917 120.18644 16.984726 19.76248 0.9999796
##                    ACF1 Theil's U
## Training set  0.09609745        NA
## Test set     -0.71161499  0.824693
```

Whether you want to evaluate models using one-step or multi-step ahead forecasts depend on the application. Under the situation that your firm has a very responsive supply chain, and is able to source products quickly

(say less than a month), then one-step ahead forecast would be sufficient. However, in many cases, firms have only one chance to place order, and need to determine order quantities for the entire selling season (say the whole winter from Oct to Feb) well in advance. In this case, firms prefer to minimize forecasting errors for the entire selling season, at the time when the order is placed. So, a model with lowest multi-period ahead forecasting errors is preferred.