ECON 430 Project 2

Gefei Zhao

12/4/2020

Contents

1. Introduction

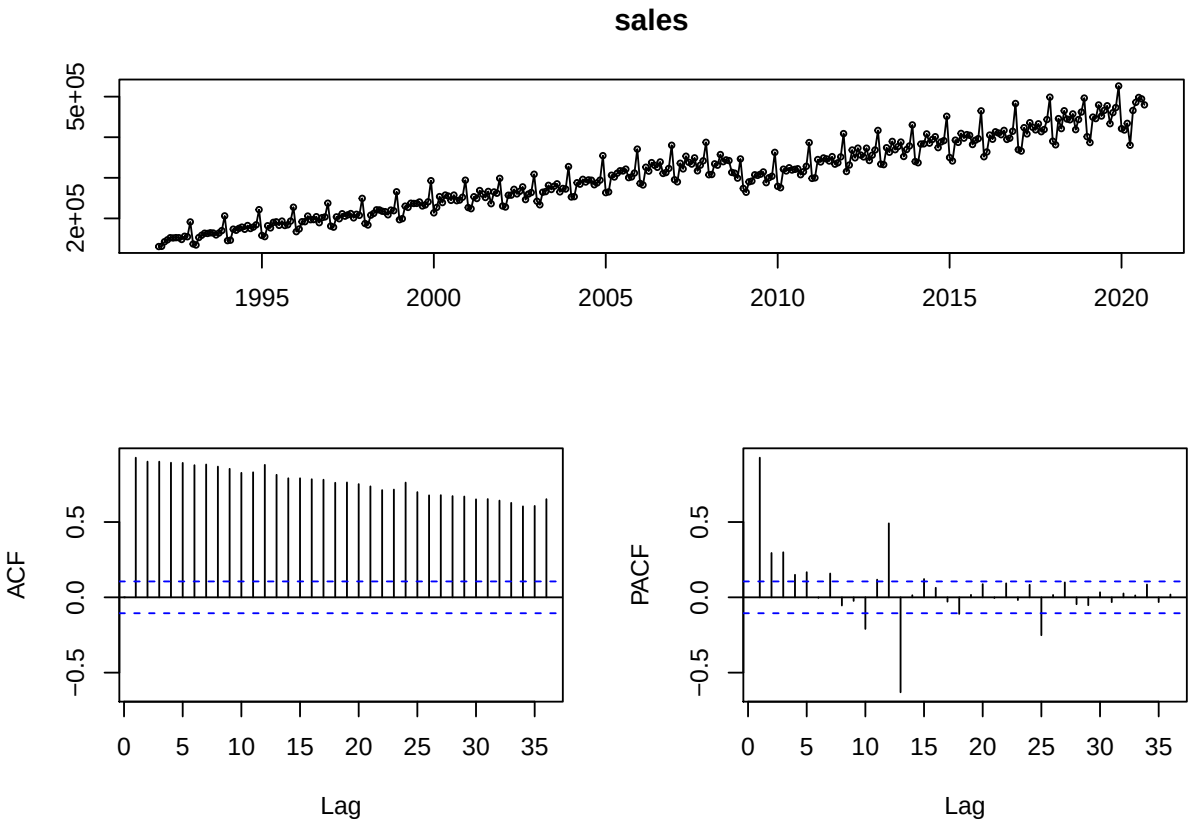
I select data of retail sales and retail inventories of the United State from January 1992 to September 2020 from FRED.

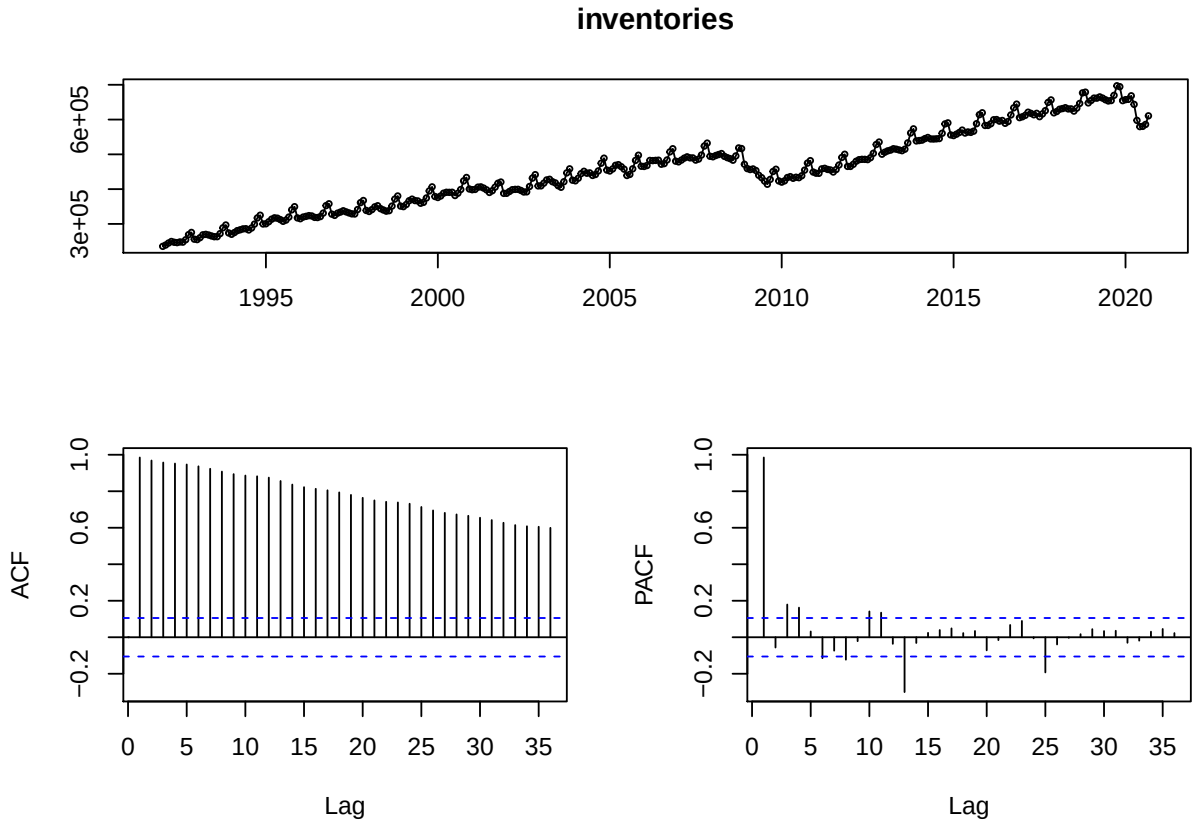
The retail sales and inventories data are both time-series with trend, seasonality and cycles and I suspect the retail sales would have a influence on the inventories in next period. Therefore, we can build respective ARIMA model and VAR model to forecast the sales and inventories of retailing in the United State.

2. Results

2.1 Time-series Plots

First of all, we can look at the time-series plots of the data.





The two time-series data have obvious trend, seasonality and cycles. Specifically, both sales and retail have a sharp decline in recession in 2009 and 2020. Besides, the patterns of sales and inventories looks identical but invent is lagging to some degree.

As for the ACF plots, both time series shows high persistence that the ACF decay slowly. In PACF plots, there are few spikes in low-level lags so that we can suspect they are more likely to be estimated by Autoregressive models.

2.2 Baseline ARIMA Model

As a baseline model, we can use the auto.arima function to fit models for sales and inventories. From the times-series plots in part 1, we can observe a increasing variance in each data. So I take log to both data to stabilize the variance then build the models.

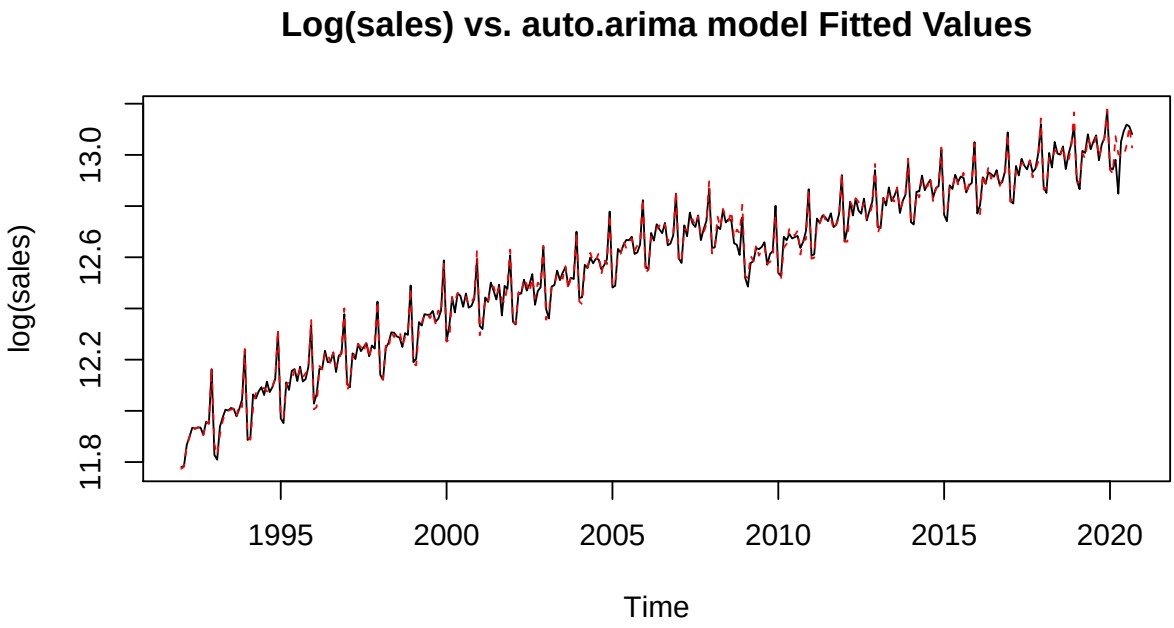
```
## Series: log(sales)
## ARIMA(2,1,2)(0,1,2)[12]
##
## Coefficients:
##           ar1      ar2     ma1      ma2     sma1     sma2
##       -0.5737  -0.1676  0.1470  -0.2376  -0.5385  -0.2260
## s.e.   0.1817   0.1421  0.1808   0.1643   0.0651   0.0624
##
## sigma^2 estimated as 0.000575:  log likelihood=767.36
## AIC=-1520.71   AICc=-1520.36   BIC=-1494.07
##
```

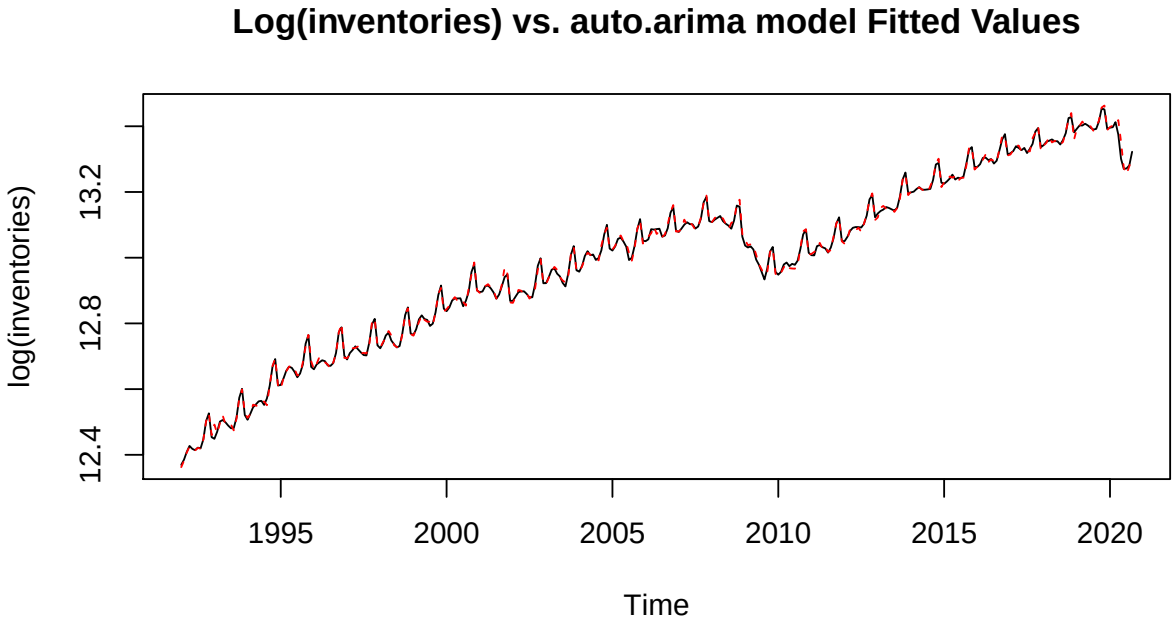
```
## Training set error measures:
##                         ME       RMSE        MAE          MPE     MAPE     MASE
## Training set -0.000391932 0.02330966 0.01641816 -0.003264256 0.130366 0.323554
##                      ACF1
## Training set 0.007960705
```

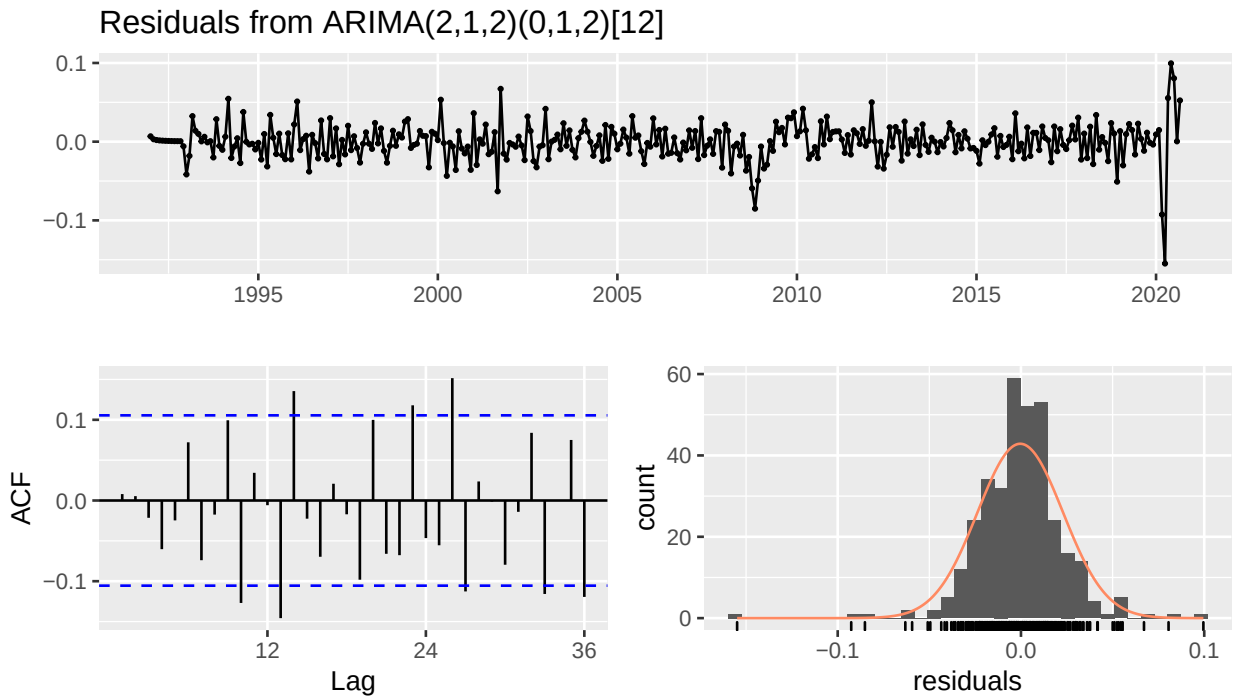
```
## Series: log(inventories)
## ARIMA(2,1,1)(1,1,2)[12]
##
## Coefficients:
##           ar1     ar2     ma1    sar1     sma1     sma2
##       -0.5765  0.3187  0.9660  0.0168  -0.4736  -0.2989
## s.e.   0.0703  0.0609  0.0406  0.1518   0.1400   0.1029
##
## sigma^2 estimated as 7.81e-05:  log likelihood=1110.16
## AIC=-2206.32   AICc=-2205.97   BIC=-2179.68
##
## Training set error measures:
##                          ME        RMSE         MAE          MPE       MAPE
## Training set -0.0006048229 0.008590796 0.006144264 -0.004653762 0.04725313
##                    MASE        ACF1
## Training set 0.1262384 -0.01388041
```

The function suggests a ARIMA(2,1,2) with a monthly seasonal ARIMA(0,1,2) for log(sales) and a ARIMA(2,1,1) with a monthly seasonal ARIMA(1,1,2) for log(inventories). The sigma squares, log likelihood and the error metrics look good for both models.

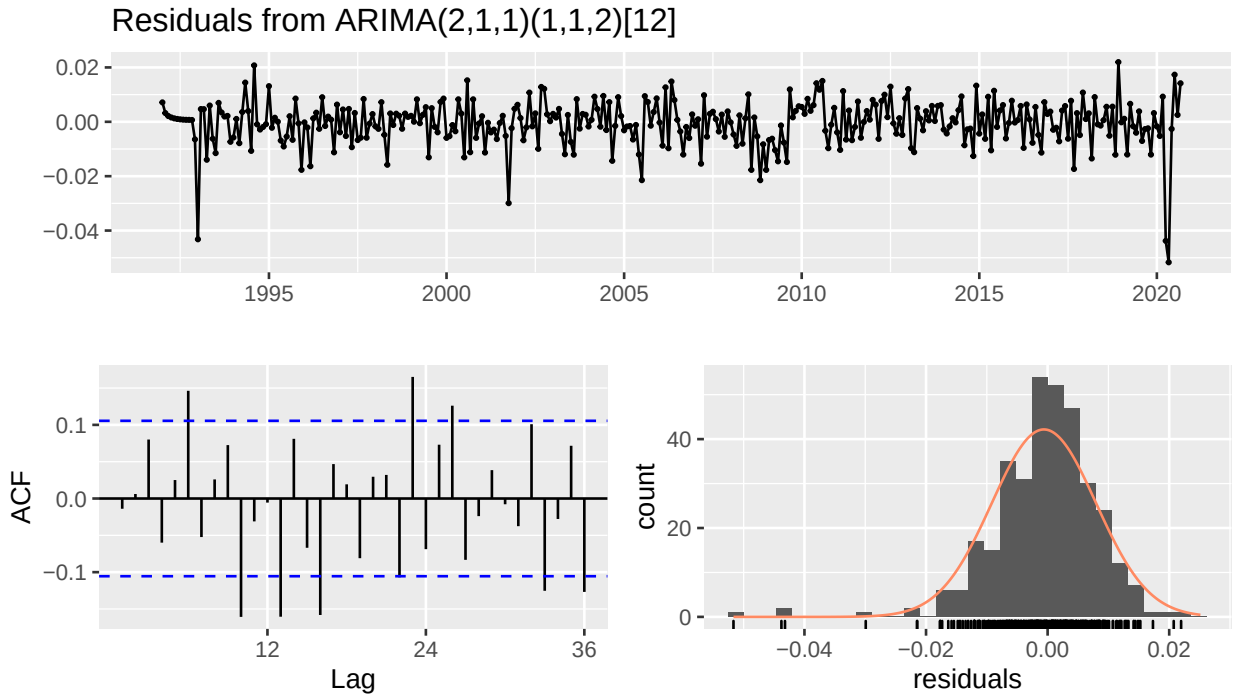
Next, we can further look at the fitted values and residuals for respective model to thoroughly comment on the performances of models.


Log(sales) vs. auto.arima model Fitted Values
log(sales)
11.8
12.2
12.6
13.0
1995
2000
2005
2010
2015
2020
Time


Log(inventories) vs. auto.arima model Fitted Values
log(inventories)
12.4
12.8
13.2
1995
2000
2005
2010
2015
2020
Time




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,2)(0,1,2)[12]
## Q* = 48.304, df = 18, p-value = 0.0001357
##
## Model df: 6.   Total lags used: 24
```



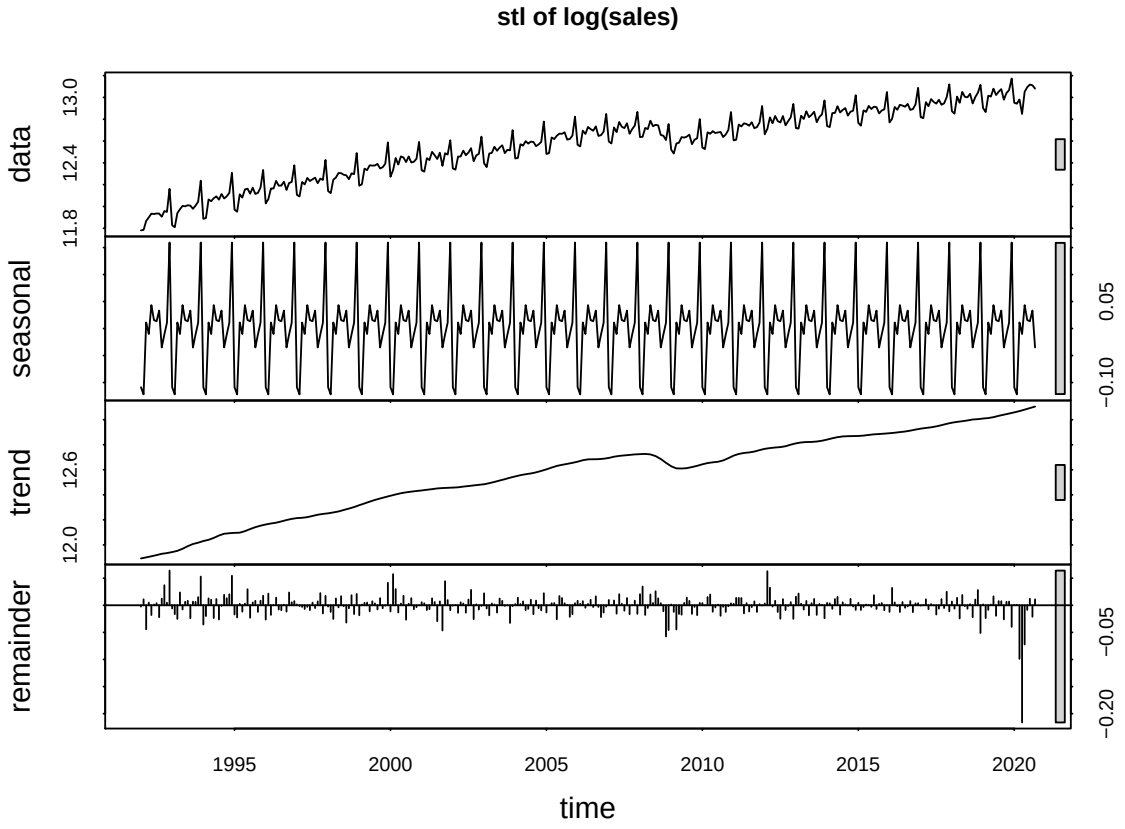
```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)(1,1,2)[12]
## Q* = 66.607, df = 18, p-value = 1.68e-07
##
## Model df: 6.   Total lags used: 24
```

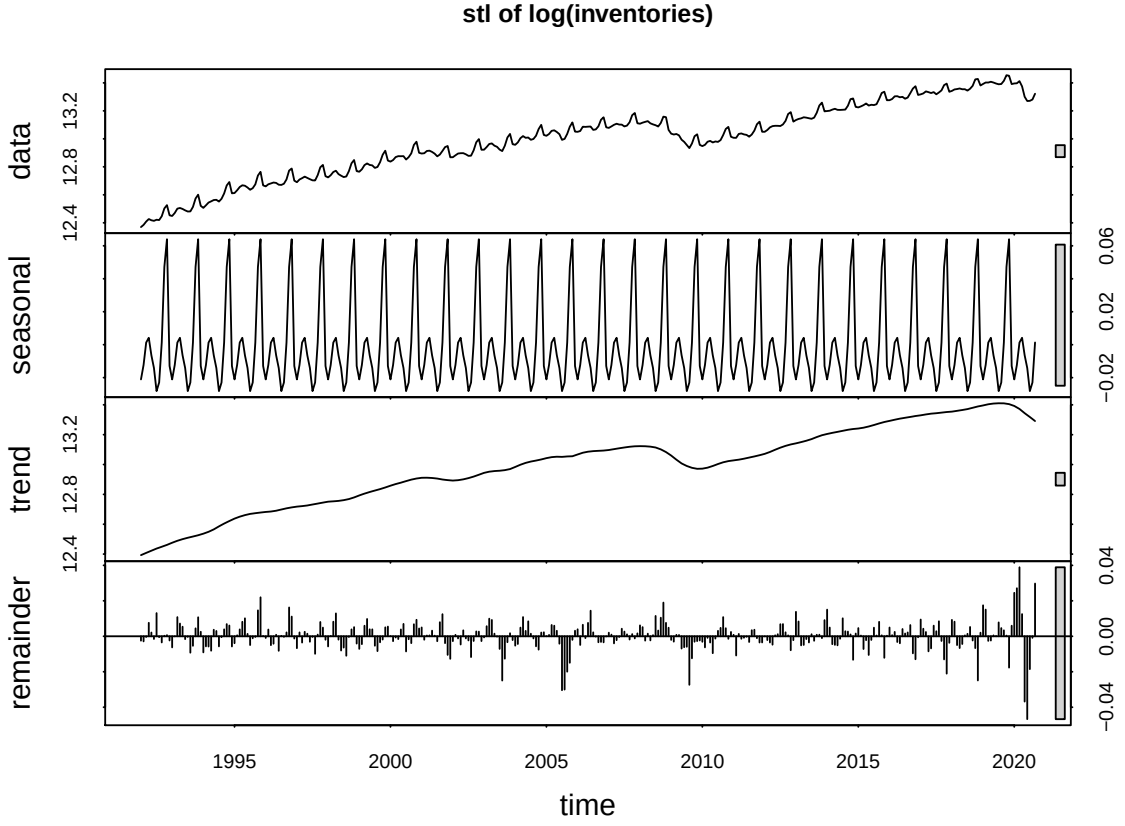
The curves seem to fit good according to the plots of real data vs. fitted values. However, when we look at the residuals, there are some spikes in ACF plots and the distribution of residuals are not normal, along with the Ljung-Box test reject null, indicating there are still some dynamic left and not capture by the arima models.

2.3 Model Including Trend, Seasonality and Cycle

To build our ARIMA model, we can firstly look at the decomposition of the time-series and get a brief overview of the components of model.

Log(sales) and Log(inventories) both have a linear or quadratic trend and a downturn in 2009. As for seasonality, the data before log have increasing seasonal fluctuations with time which is multiplicative seasonality. Since we take log for the data and stablize the variance, $log(y_t) = log(S_t * T_t * R_t) = log(S_t) + log(T_t) + log(R_t)$. We can remove seasonality by simply substract the fitted value of seasonality.




stl of log(inventories)
data
seasonal
trend
remainder
12.4
12.8
13.2
−0.02
0.02
0.06
−0.04
0.00
0.04
1995
2000
2005
2010
2015
2020
time

2.3.1 Trend Fitting

Table 1:

	Dependent variable:	
	lsales	
	(1)	(2)
t	0.037***	3.035***
	(0.001)	(0.291)
t2		−0.001***
		(0.0001)
Constant	−62.159***	−3,069.245***
	(1.238)	(291.937)
Observations	345	345
R^2	0.914	0.934
Adjusted R^2	0.914	0.934
Residual Std. Error	0.095 (df = 343)	0.083 (df = 342)
F Statistic	3,642.553*** (df = 1; 343)	2,432.396*** (df = 2; 342)
Note:		*p<0.1; **p<0.05; ***p<0.01

Table 2:

	Dependent variable:	
	linvent	
	(1)	(2)
t	0.030***	1.979***
	(0.0005)	(0.231)
t2		−0.0005***
		(0.0001)
Constant	−47.579***	−2,002.389***
	(0.945)	(231.968)
Observations	345	345
R^2	0.923	0.936
Adjusted R^2	0.923	0.936
Residual Std. Error	0.073 (df = 343)	0.066 (df = 342)
F Statistic	4,112.388*** (df = 1; 343)	2,511.434*** (df = 2; 342)
Note:		*p<0.1; **p<0.05; ***p<0.01

The estimate of coefficients are all statistically significant and the R^2 are higher in log-quadratic model for sales and inventories.

```
##     logLinear   logQuad
## AIC -638.1781 -729.3997
```

```
## BIC -626.6474 -714.0255
```

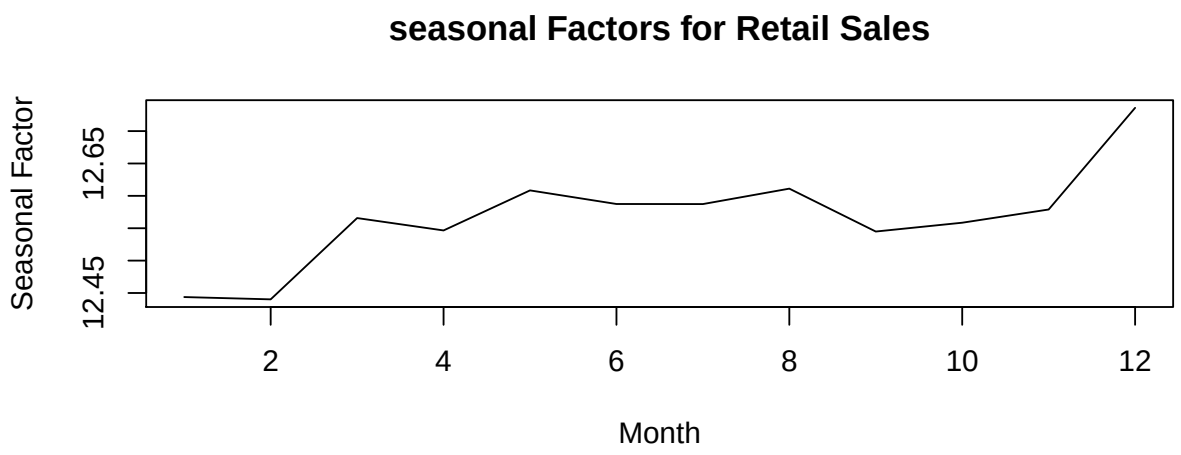
```
##     logLinear   logQuad
## AIC -824.9652 -888.0589
## BIC -813.4346 -872.6847
```

AIC and BIC agrees wit choosing the Log-Quadratic model for both sales and inventories.

2.3.2 Seasonality Fitting

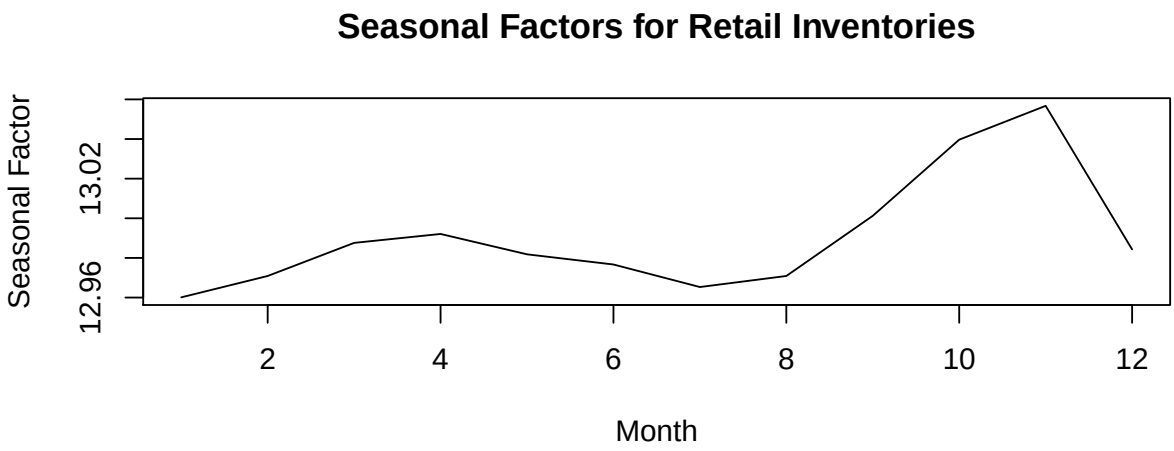
We can test for the seasonality using tslm function.

```
##
## Call:
## tslm(formula = lsales ~ season + 0)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.69869 -0.23065  0.06552  0.26472  0.53614
##
## Coefficients:
##          Estimate Std. Error t value Pr(>|t|)
## season1  12.44372    0.05969   208.5   <2e-16 ***
## season2  12.44019    0.05969   208.4   <2e-16 ***
## season3  12.56571    0.05969   210.5   <2e-16 ***
## season4  12.54659    0.05969   210.2   <2e-16 ***
## season5  12.60842    0.05969   211.2   <2e-16 ***
## season6  12.58742    0.05969   210.9   <2e-16 ***
## season7  12.58732    0.05969   210.9   <2e-16 ***
## season8  12.61114    0.05969   211.3   <2e-16 ***
## season9  12.54490    0.05969   210.2   <2e-16 ***
## season10 12.55854    0.06075   206.7   <2e-16 ***
## season11 12.57894    0.06075   207.1   <2e-16 ***
## season12 12.73590    0.06075   209.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3214 on 333 degrees of freedom
## Multiple R-squared:  0.9994, Adjusted R-squared:  0.9993
## F-statistic: 4.394e+04 on 12 and 333 DF,  p-value: < 2.2e-16
```



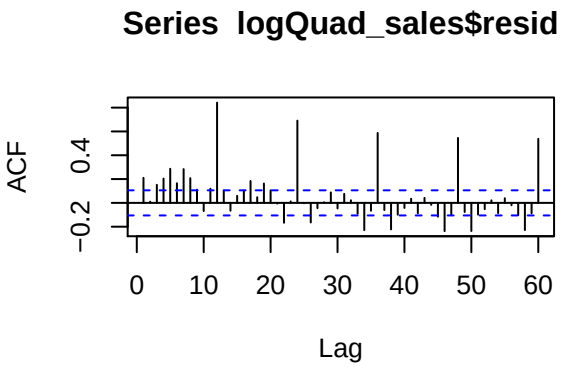
There are seasonality in the time series of sales because estimates of parameters are statistically significant. The coefficients suggest that retail sales peak at December since people love to consume in the holidays. And the retail stores sell worst in January and Feburary because people prefer to consume less and save money after shopping in December.

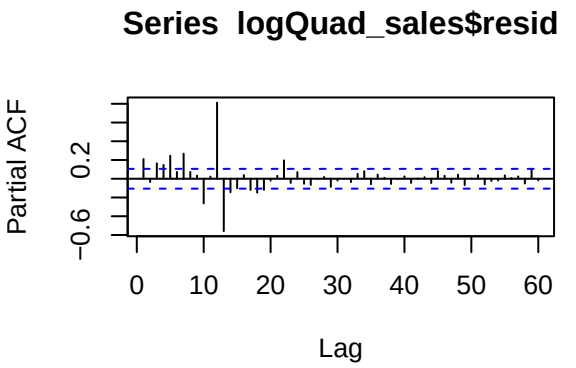
```
##
## Call:
## tslm(formula = linvent ~ season + 0)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59060 -0.17522  0.03027  0.17797  0.43570
##
## Coefficients:
##          Estimate Std. Error t value Pr(>|t|)
## season1  12.96011    0.04915   263.7   <2e-16 ***
## season2  12.97088    0.04915   263.9   <2e-16 ***
## season3  12.98759    0.04915   264.2   <2e-16 ***
## season4  12.99209    0.04915   264.3   <2e-16 ***
## season5  12.98182    0.04915   264.1   <2e-16 ***
## season6  12.97668    0.04915   264.0   <2e-16 ***
## season7  12.96532    0.04915   263.8   <2e-16 ***
## season8  12.97088    0.04915   263.9   <2e-16 ***
## season9  13.00124    0.04915   264.5   <2e-16 ***
## season10 13.03967    0.05002   260.7   <2e-16 ***
## season11 13.05673    0.05002   261.0   <2e-16 ***
## season12 12.98430    0.05002   259.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2647 on 333 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 6.925e+04 on 12 and 333 DF,  p-value: < 2.2e-16
```

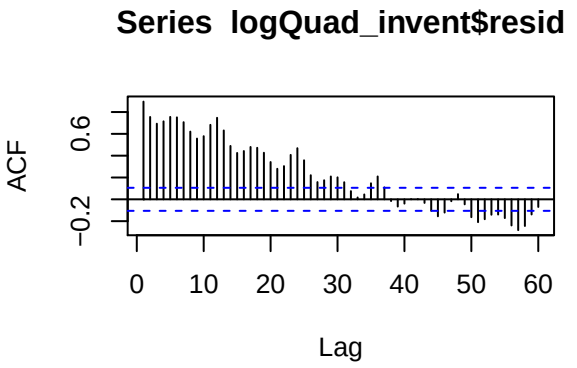


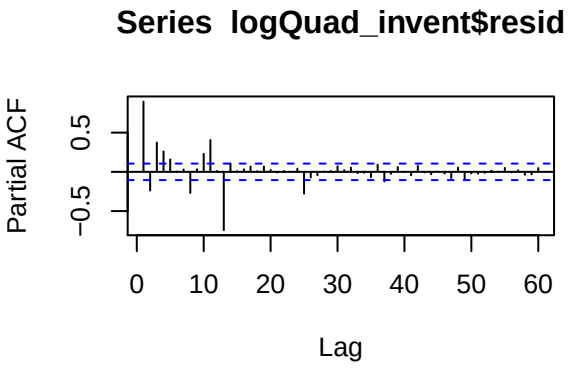
The estimates of parameters are statistically significant as well so there do exist seasonality for retail inventories. Compared to the seasonal factor of sales, the inventories peak at November because they need to prepare as much as possible to satisfy the larger consumption in the coming December.

To find the fitted ARMA model for seasonality of the time series, we can look at the ACF and PACF plots of residuals of the trend model.





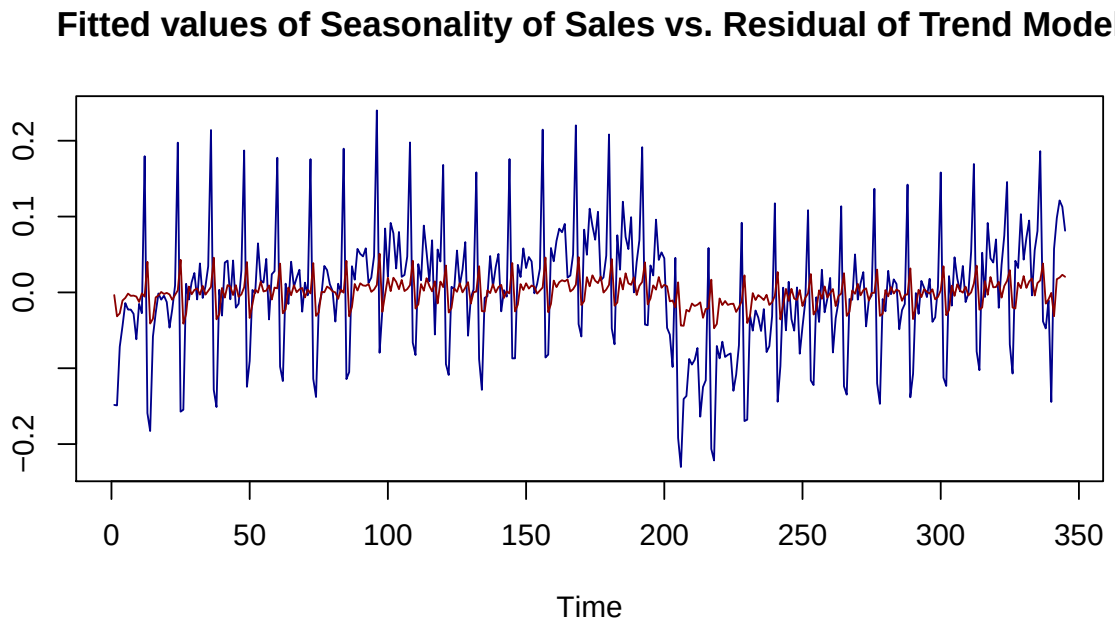




Log(sales) is typical S-AR(2) model, because spikes at 12, 24, 36,... and decay to zero in ACF plot and there are 2 spike at 12 and cut-off at 24.

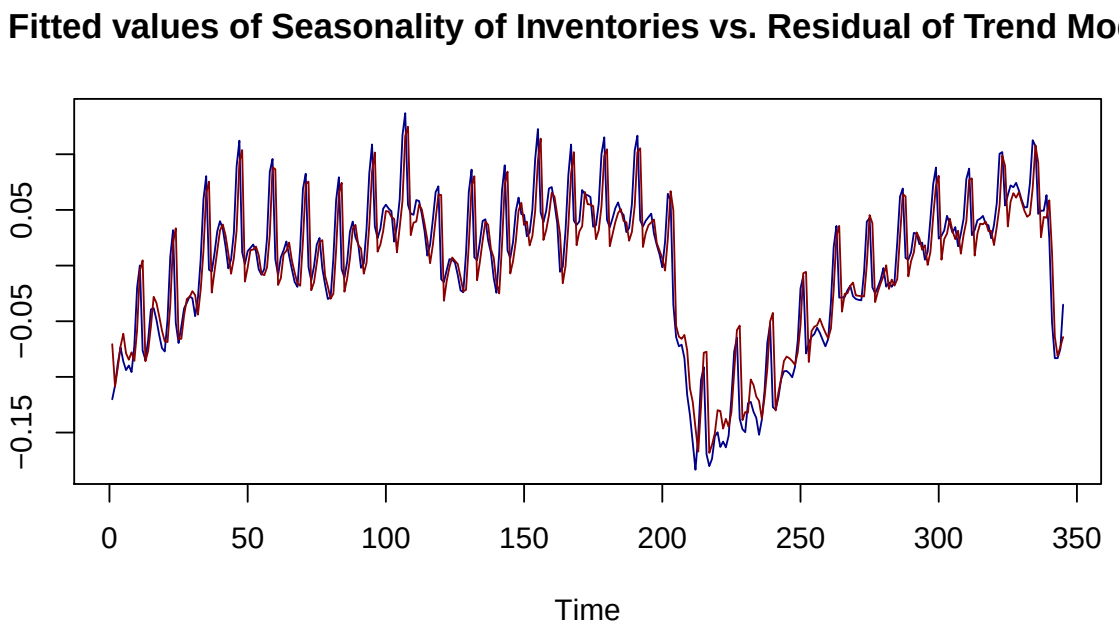
However there are other spikes at non-seasonl lag, log(Inventories) looks like kind of S-AR(2), for the decay spikes at 12, 24,... and spikes ar 12, 24.

```
##
## Call:
## arima(x = logQuad_sales$resid, seasonal = list(order = c(2, 0, 0)))
##
## Coefficients:
##         sar1     sar2  intercept
##       0.2193  -0.0339     0.0000
## s.e.  0.0539   0.0543     0.0054
##
## sigma^2 estimated as 0.006589:  log likelihood = 376.79,  aic = -745.58
##
## Training set error measures:
##                       ME       RMSE        MAE      MPE     MAPE     MASE
## Training set 7.32144e-05 0.08117473 0.06097285 669.8118 695.5575 0.868536
##                      ACF1
## Training set 0.004668999
```

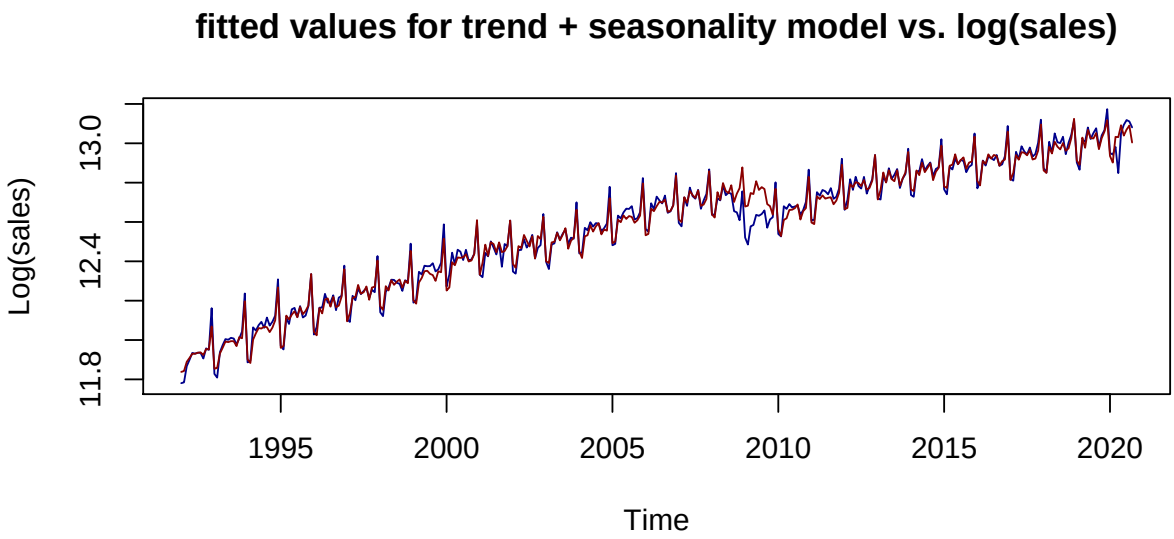


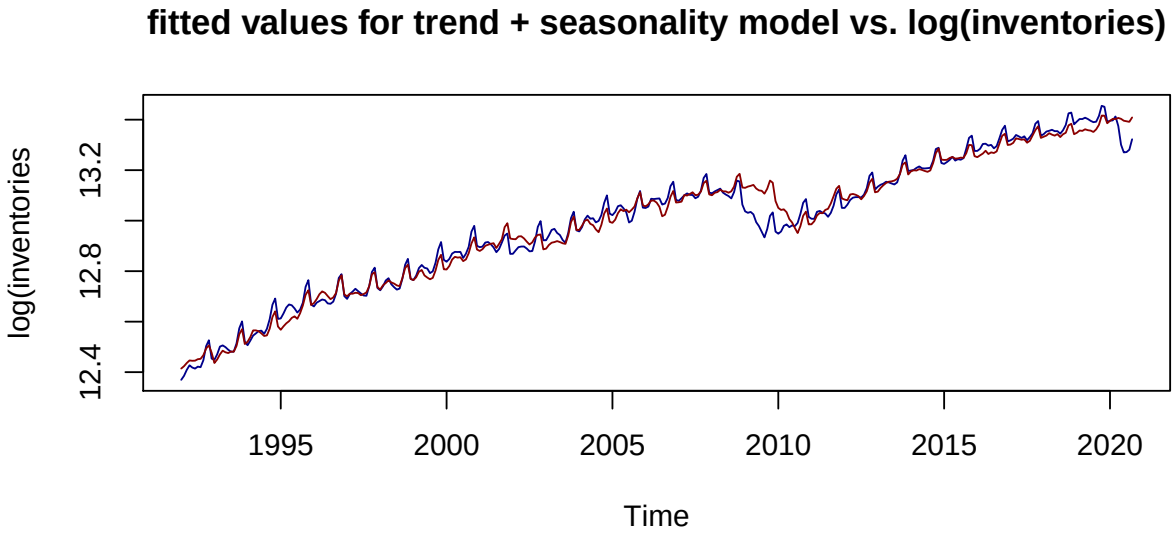

```
##
## Call:
## arima(x = logQuad_invent$resid, seasonal = list(order = c(2, 0, 0)))
##
## Coefficients:
##         sar1     sar2  intercept
##       1.1234  -0.2465    -0.0022
## s.e.  0.0522   0.0525     0.0120
##
## sigma^2 estimated as 0.0007772:  log likelihood = 744.64,  aic = -1481.28
##
## Training set error measures:
```

```
##                         ME       RMSE        MAE      MPE     MAPE      MASE
## Training set 0.0003639497 0.02787784 0.01988982 59.66659 139.4749 0.9769435
##                     ACF1
## Training set 0.09181077
```



We can see the models fit well from the plots of fitted values and residual of trend model. Then we can combine the trend and seasonality model together.





The distinction between the real data and the fitted value is the cycles of the time series. We can further use the residual of trend + seasonality model to explore the fitting of cycles.

2.3.3 Cycles Fitting

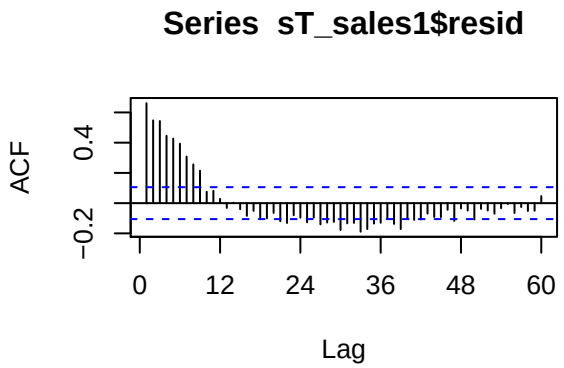
On the very begining of the fitting of cycles, we should ensure the data are stationary.

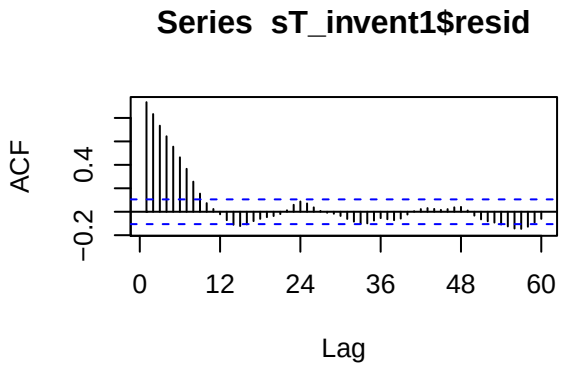
```
##
##  Augmented Dickey-Fuller Test
##
## data:  sT_sales1$resid
## Dickey-Fuller = -2.6128, Lag order = 36, p-value = 0.3182
## alternative hypothesis: stationary
##
```

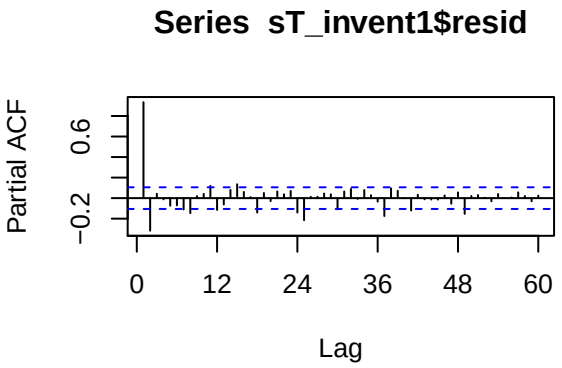
```
##  Augmented Dickey-Fuller Test
##
## data:  sT_invent1$resid
## Dickey-Fuller = -3.0273, Lag order = 36, p-value = 0.1433
## alternative hypothesis: stationary
```

The p-values are large so that we fail to reject the null hypothesis of stationary.

Next, we can look at the ACF and PACF plots of residuals of the seasonality and trend model to determine the order of model of cycles.







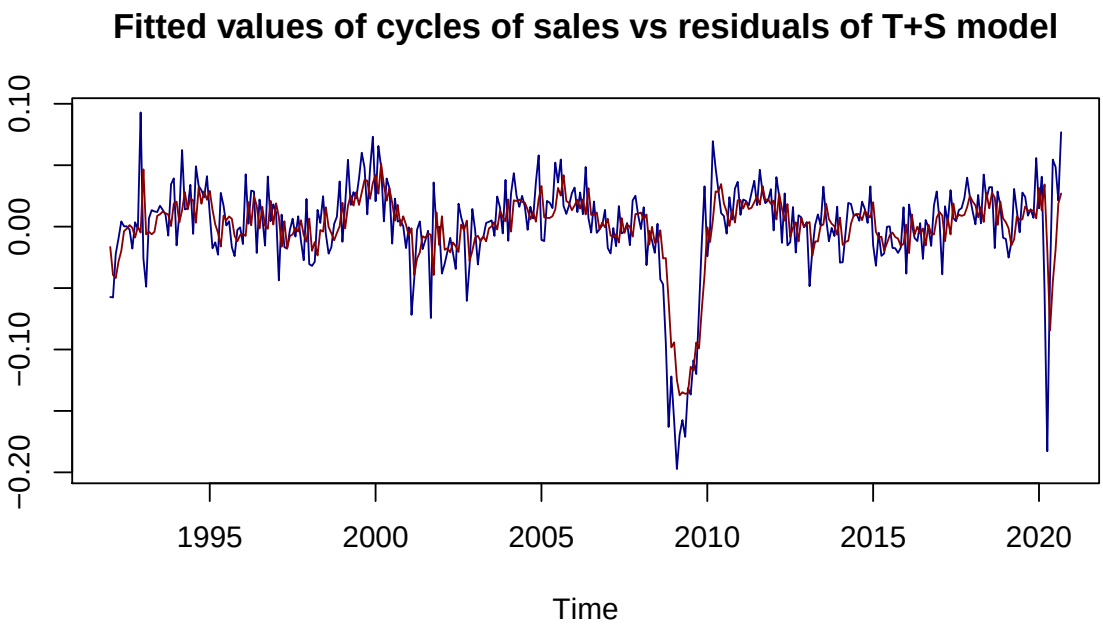
The ACF plots are typical pattern of AR process. Then we can look at the PACF plots to do order determination. Sales is an AR(3) model and inventories are an AR(2) model.

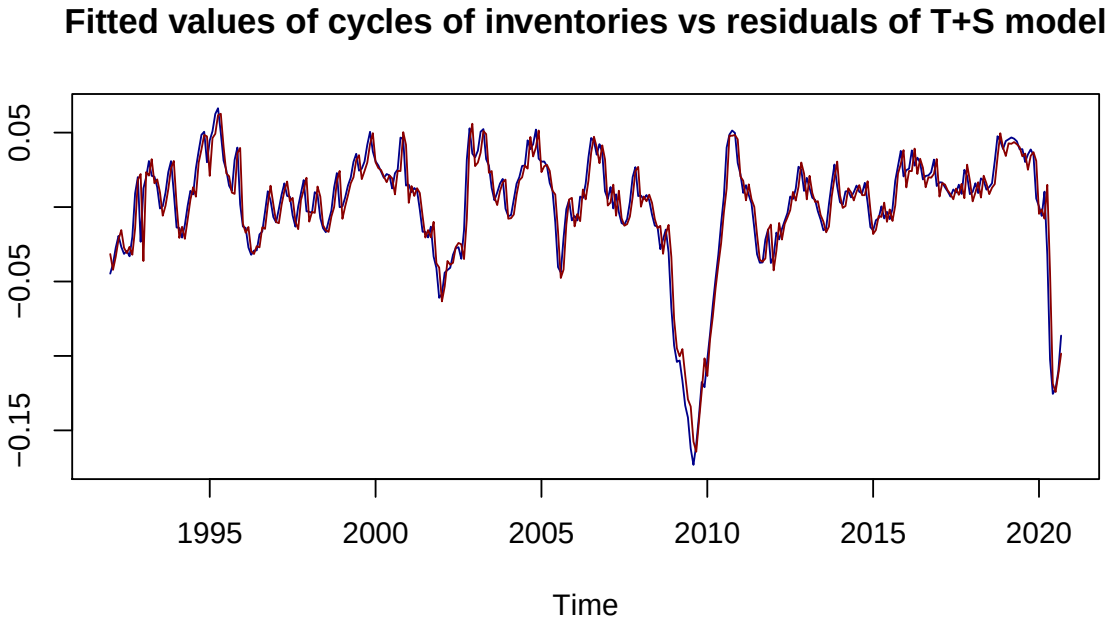
```
##
## Call:
## arima(x = sT_sales1$resid, order = c(3, 0, 0))
##
## Coefficients:
##          ar1     ar2     ar3  intercept
##       0.4932  0.0734  0.2341     0.0007
## s.e.  0.0525  0.0590  0.0527     0.0073
##
## sigma^2 estimated as 0.0007661:  log likelihood = 747.59,  aic = -1485.17
##
## Training set error measures:
##                       ME       RMSE        MAE       MPE     MAPE      MASE
## Training set 0.0003133043 0.02767858 0.01985077 -32.49791 261.6359 0.8388087
##                     ACF1
```

```
## Training set 0.002632785
```

```
##
## Call:
## arima(x = sT_invent1$resid, order = c(2, 0, 0))
##
## Coefficients:
##          ar1      ar2  intercept
##       1.2616  -0.3344    -0.0018
## s.e.  0.0509   0.0516     0.0087
##
## sigma^2 estimated as 0.0001442:  log likelihood = 1034.85,  aic = -2061.7
##
## Training set error measures:
##                       ME       RMSE         MAE      MPE     MAPE      MASE
## Training set 0.00012928 0.01200891 0.008966103 119.2692 188.6817 0.9308051
##                     ACF1
## Training set 0.02232866
```

The sigma squares and error metrics are quite small, indicating the models fits good with the real data. We can also see this from the fitted value plots.



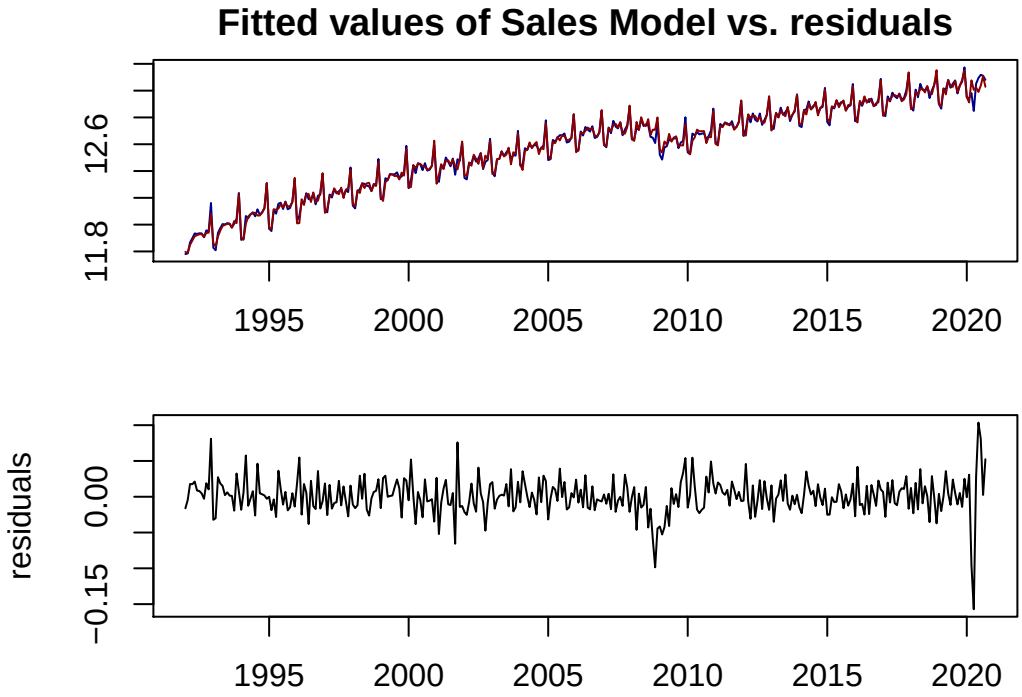


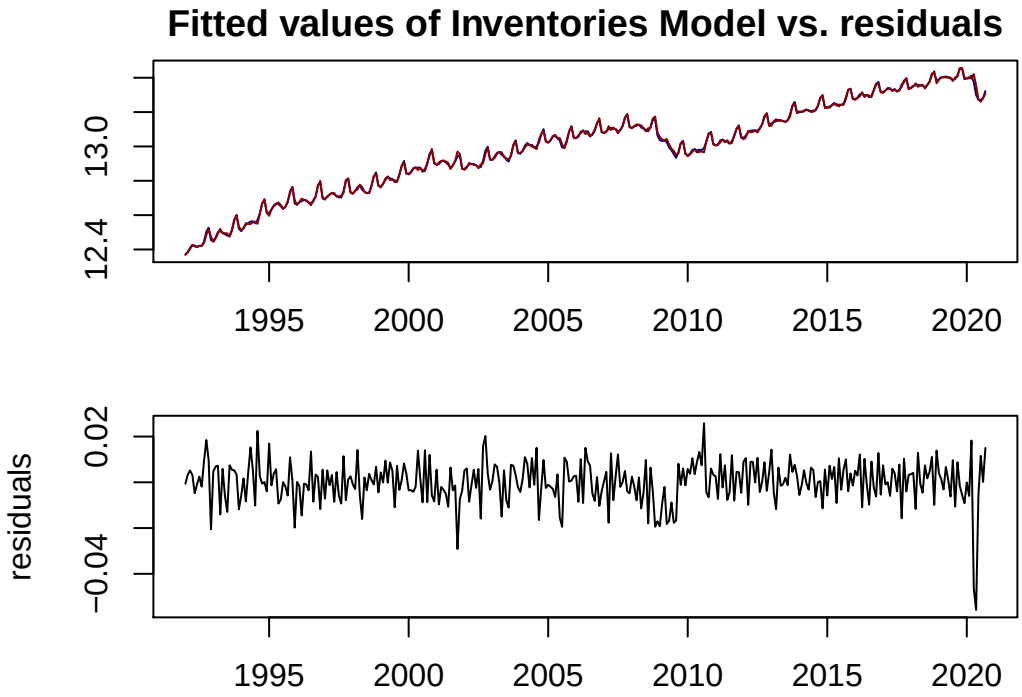
2.3.4 Full Model

At last, we combine the above three model into the full model.

```
# TREND + SEASONALITY + CYCLE
tsc_sales <- Arima(lsales, order = c(3, 0, 0), xreg = cbind(t, t2), seasonal = list(order = c(2, 0, 0)))
tsc_invent <- Arima(linvent, order = c(2, 0, 0), xreg = cbind(t, t2), seasonal = list(order = c(2, 0, 0
```

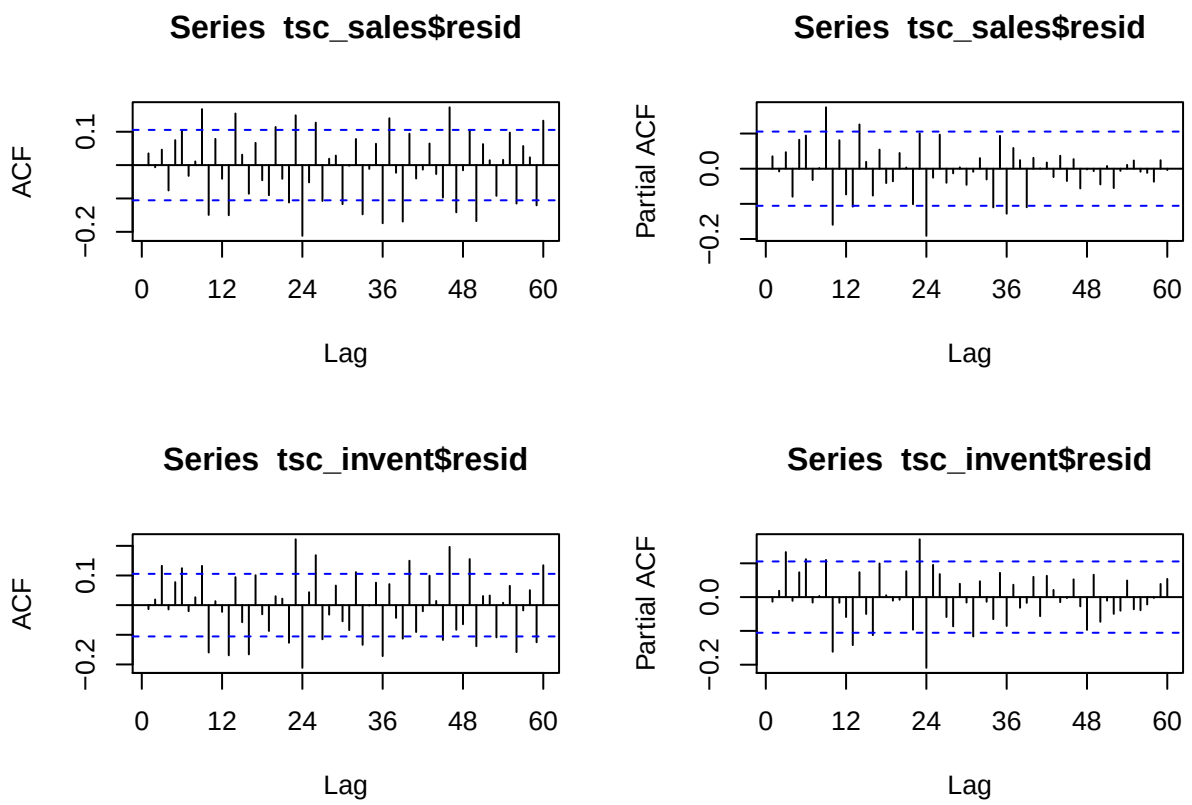
2.4 Residuals vs. Fitted values





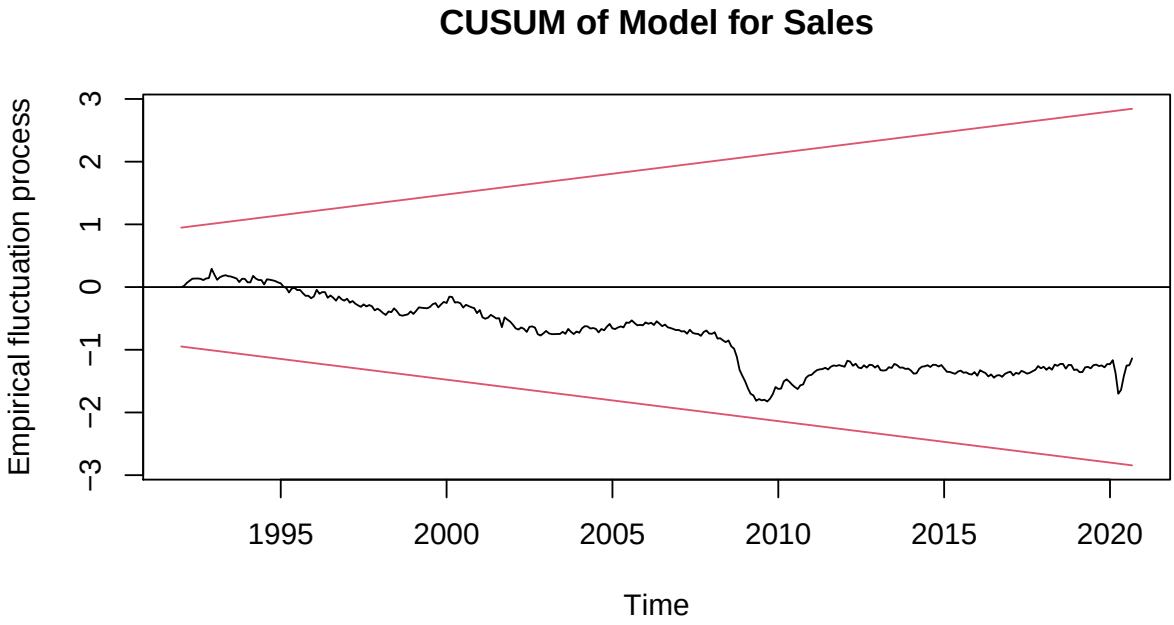
The residuals are around zero and has serveral shocks in the recession in 2009 and 2020. However, the residuals are quite small. So the error of our model is small.

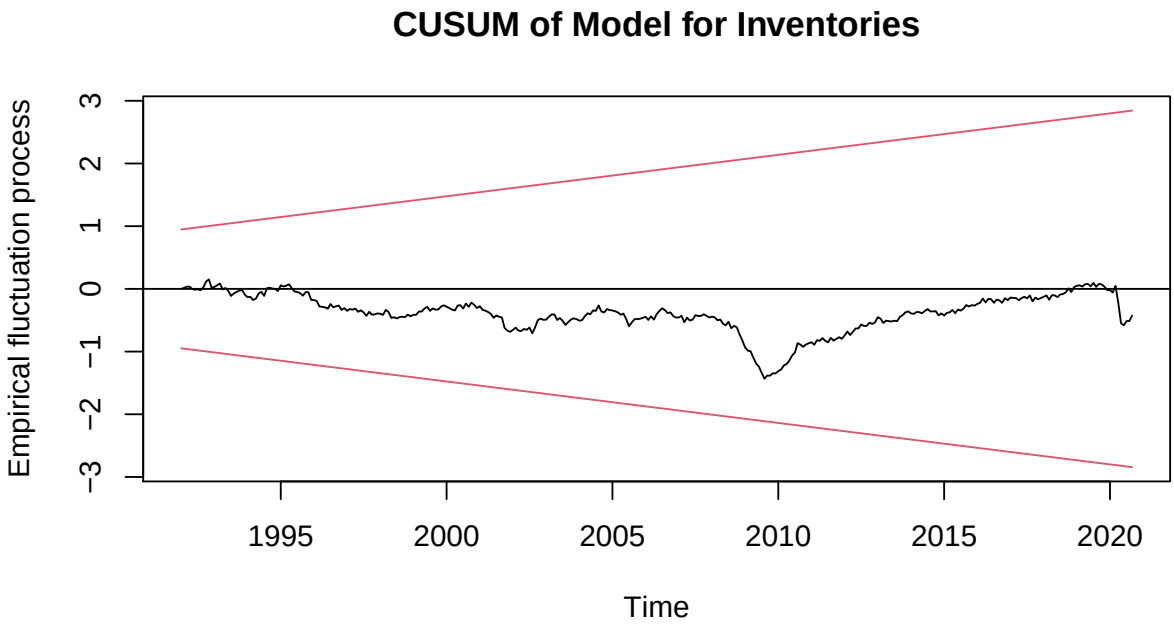
2.5 ACF and PACF of residuals



However, the ACF and PACF looks bad, there still some autocorrelations and partial correlations in the residual. I tried to take difference on data but R shows something goes infinite so the model cannot be estimated. Compared with the ARIMA model estimated by auto.arima function, their residuals also have some significant non-zero spikes. I infer that there are some dynamics in the variance term and we can talk about it later.

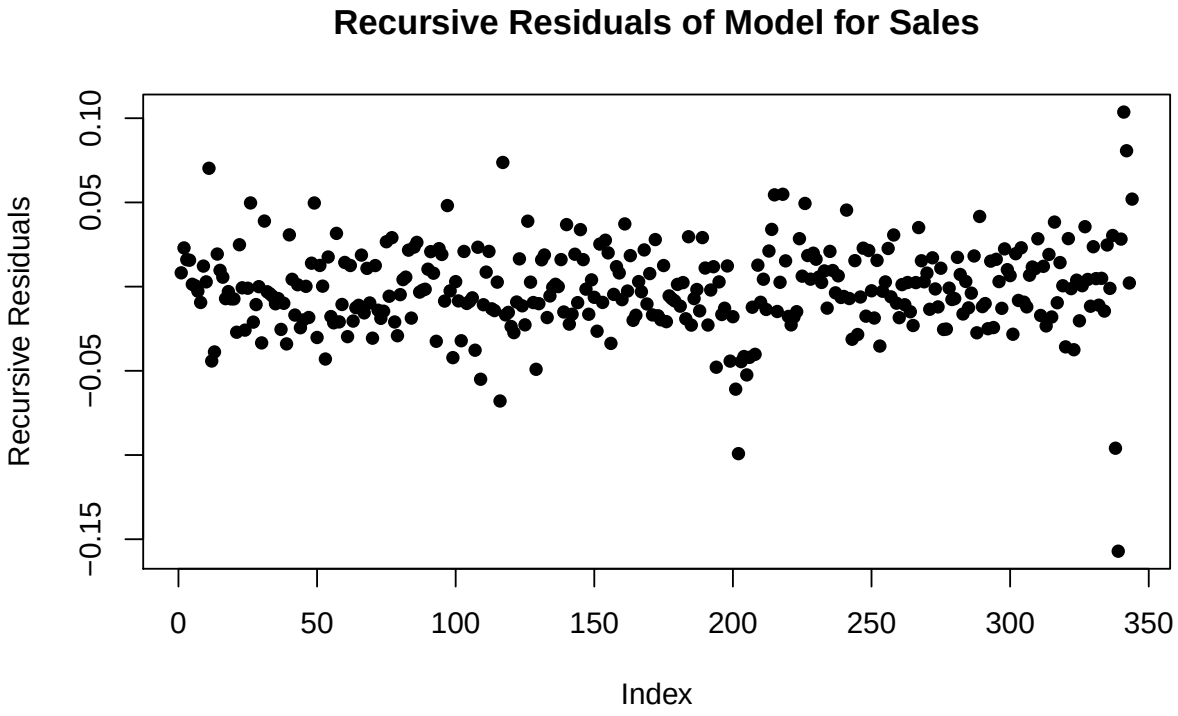
2.6 CUSUM

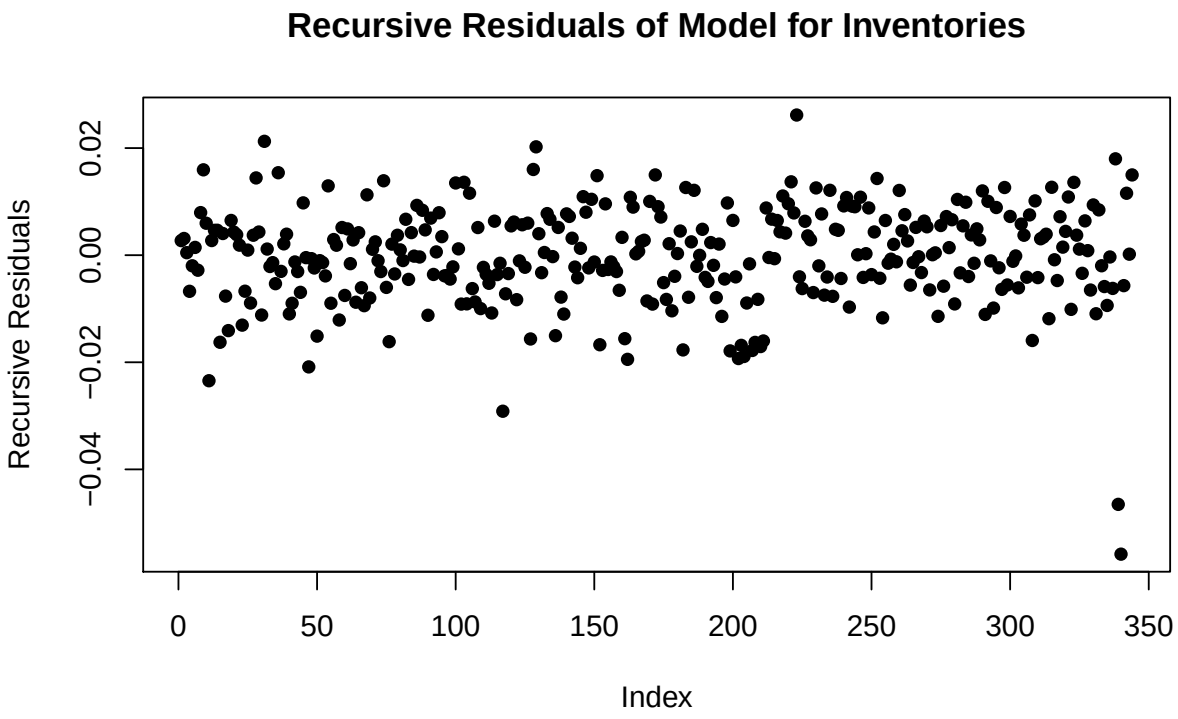




The CUSUM of each time-series do not cross the interval of red line, suggesting there is no structural break in each model.

2.7 Recursive Residuals





The recursive residuals are scattered around zero which are good.

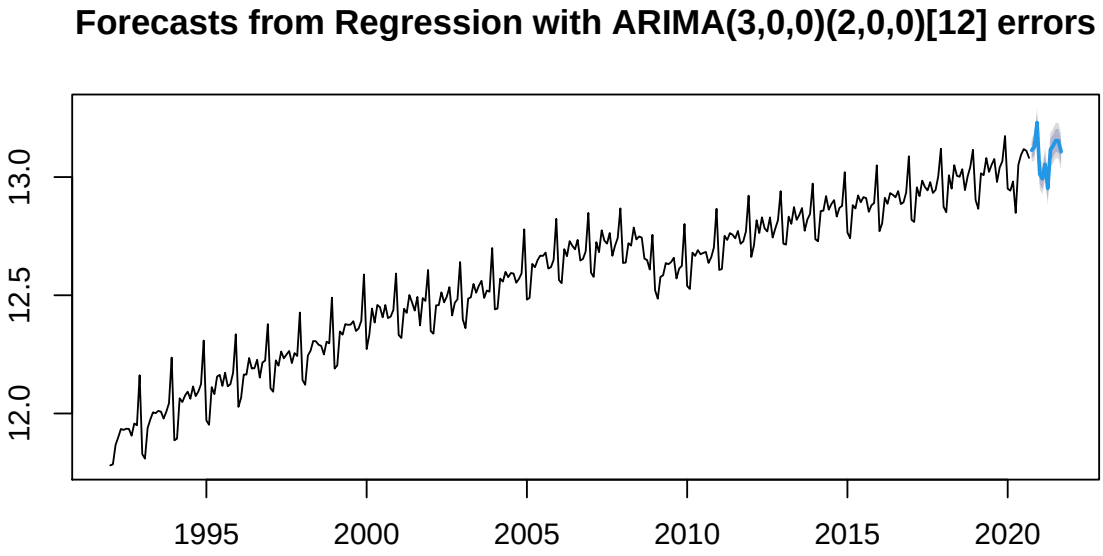
2.8 Diagnostic Statistics

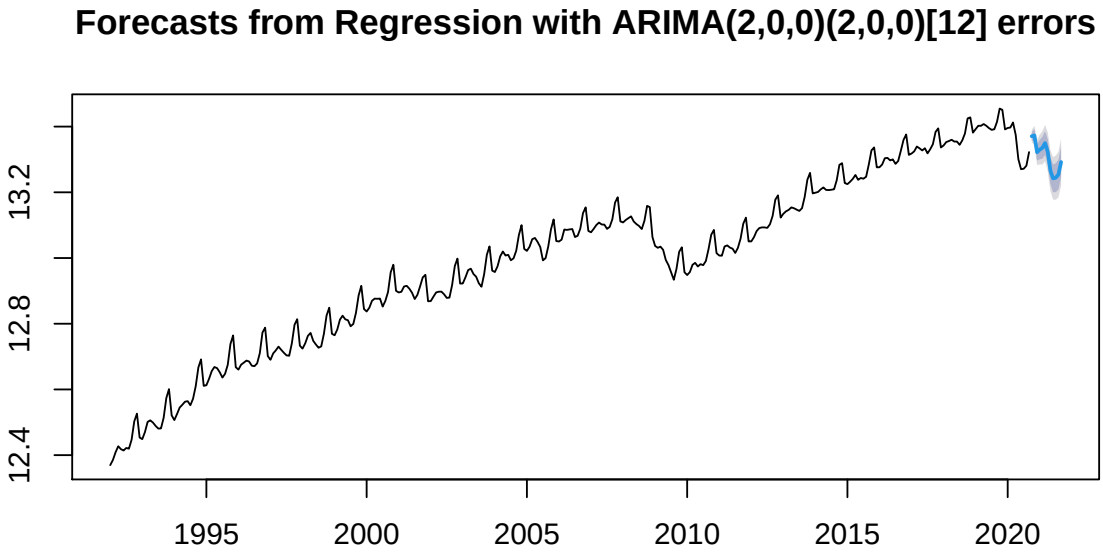
```
## Series: lsales
## Regression with ARIMA(3,0,0)(2,0,0)[12] errors
##
## Coefficients:
##          ar1     ar2     ar3    sar1    sar2   intercept       t      t2
##       0.4806  0.0940  0.2827  0.7385  0.2276  -2251.9753  2.2163  -5e-04
## s.e.  0.0524  0.0583  0.0531  0.0632  0.0645    224.2801  0.2200   1e-04
##
## sigma^2 estimated as 0.0006658:  log likelihood=759.96
## AIC=-1501.92   AICc=-1501.38   BIC=-1467.33
##
## Training set error measures:
##                        ME       RMSE        MAE        MPE      MAPE      MASE
## Training set 0.0004945828 0.02550279 0.01845321 0.00385917 0.1467625 0.3636589
##                    ACF1
## Training set 0.03513445
```

```
## Series: linvent
## Regression with ARIMA(2,0,0)(2,0,0)[12] errors
##
## Coefficients:
##          ar1      ar2    sar1    sar2   intercept       t       t2
##       1.2840  -0.3182  0.7927  0.1624  -4989.7472  4.9565  -0.0012
## s.e.  0.0536   0.0547  0.0629  0.0641    613.1935  0.6101   0.0002
##
## sigma^2 estimated as 8.731e-05:  log likelihood=1110.09
## AIC=-2204.18   AICc=-2203.75   BIC=-2173.43
##
## Training set error measures:
##                        ME        RMSE         MAE          MPE       MAPE
## Training set 4.222049e-05 0.009248734 0.006972003 0.0003352194 0.05364772
##                    MASE        ACF1
## Training set 0.1432449 -0.01346241
```

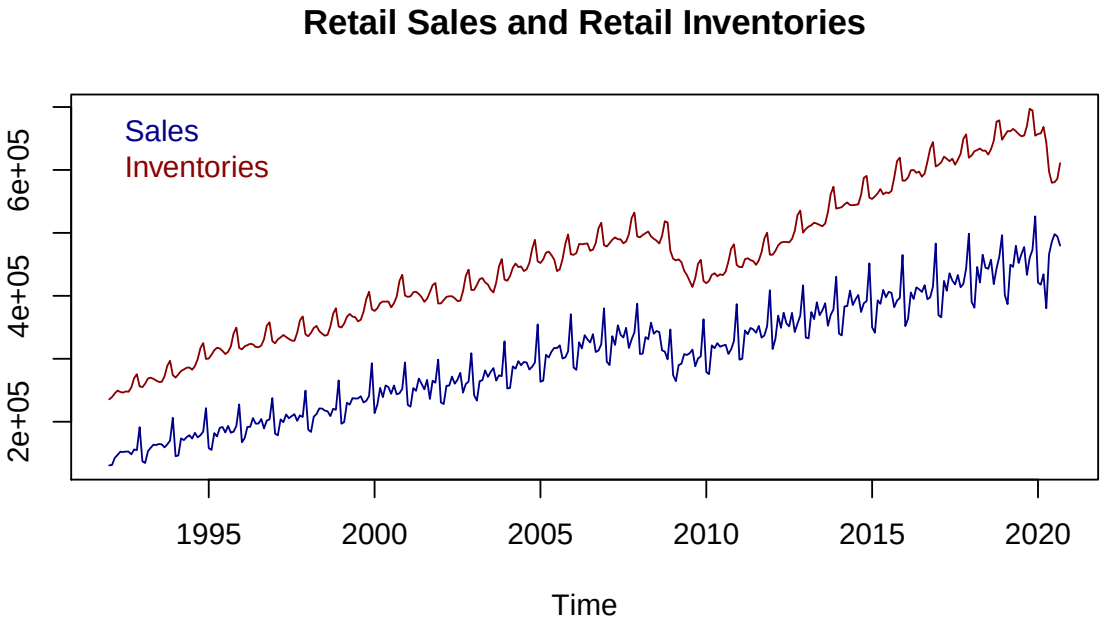
For both model, the standard errors are small so that our estimates of coefficients are significant. The error metrics are quite small as well, indicating our models fits good.

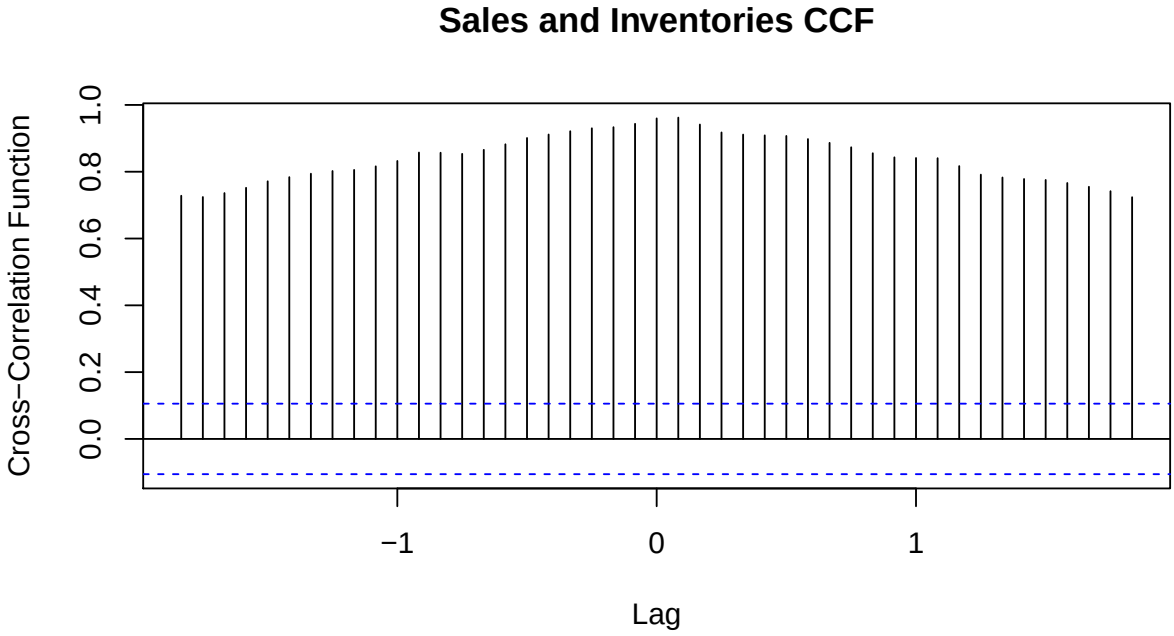
2.9 Forecast


Forecasts from Regression with ARIMA(3,0,0)(2,0,0)[12] errors
12.0
12.5
13.0
1995
2000
2005
2010
2015
2020


Forecasts from Regression with ARIMA(2,0,0)(2,0,0)[12] errors
12.4
12.8
13.2
1995
2000
2005
2010
2015
2020

2.10 VAR model





Combining the time-series plot of both time series and the cross-correlation function, they suggest that retail sales influence retail inventories by one month.

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     16     16     16     16
##
```

```
## $criteria
##                  1             2             3             4             5
## AIC(n) -1.232386e+01 -1.282265e+01 -1.343815e+01 -1.379886e+01 -1.402376e+01
## HQ(n)  -1.229598e+01 -1.277619e+01 -1.337309e+01 -1.371522e+01 -1.392153e+01
## SC(n)  -1.225400e+01 -1.270623e+01 -1.327515e+01 -1.358929e+01 -1.376762e+01
## FPE(n)  4.444436e-06  2.698948e-06  1.458455e-06  1.016820e-06  8.120497e-07
##                  6             7             8             9            10
## AIC(n) -1.410199e+01 -1.411066e+01 -1.457886e+01 -1.490388e+01 -1.526685e+01
## HQ(n)  -1.398118e+01 -1.397127e+01 -1.442088e+01 -1.472731e+01 -1.507170e+01
## SC(n)  -1.379929e+01 -1.376139e+01 -1.418302e+01 -1.446146e+01 -1.477787e+01
## FPE(n)  7.509634e-07  7.445145e-07  4.661904e-07  3.368547e-07  2.343398e-07
##                 11            12            13            14            15
## AIC(n) -1.567061e+01 -1.666888e+01 -1.705766e+01 -1.707331e+01 -1.707595e+01
## HQ(n)  -1.545687e+01 -1.643655e+01 -1.680675e+01 -1.680381e+01 -1.678786e+01
## SC(n)  -1.513505e+01 -1.608675e+01 -1.642897e+01 -1.639805e+01 -1.635411e+01
## FPE(n)  1.565116e-07  5.768512e-08  3.910964e-08  3.850948e-08  3.841634e-08
##                 16            17            18            19            20
## AIC(n) -1.724058e+01 -1.722161e+01 -1.721108e+01 -1.719452e+01 -1.718564e+01
## HQ(n)  -1.693391e+01 -1.689636e+01 -1.686723e+01 -1.683209e+01 -1.680462e+01
## SC(n)  -1.647217e+01 -1.640664e+01 -1.634953e+01 -1.628640e+01 -1.623095e+01
## FPE(n)  3.259270e-08  3.322586e-08  3.358805e-08  3.416044e-08  3.447828e-08
```

The order determined by the four information criteria is at 16.

The results of VAR model are shown below.

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: lsales, linvent
## Deterministic variables: const
## Sample size: 329
## Log Likelihood: 1970.605
## Roots of the characteristic polynomial:
## 0.9995 0.9995 0.998 0.998 0.9978 0.9978 0.9975 0.9975 0.9974 0.9974 0.9959 0.9896 0.9741 0.9741 0.97
## Call:
## vars::VAR(y = lsi, p = 16)
##
##
## Estimation results for equation lsales:
## =======================================
## lsales = lsales.l1 + linvent.l1 + lsales.l2 + linvent.l2 + lsales.l3 + linvent.l3 + lsales.l4 + linv
##
##              Estimate Std. Error t value Pr(>|t|)
## lsales.l1    0.592247   0.054282  10.911  < 2e-16 ***
## linvent.l1  -0.344967   0.191908  -1.798  0.07327 .
## lsales.l2    0.115142   0.065363   1.762  0.07917 .
## linvent.l2   0.350003   0.318502   1.099  0.27270
## lsales.l3    0.201565   0.066212   3.044  0.00254 **
## linvent.l3   0.253418   0.331258   0.765  0.44487
## lsales.l4   -0.015201   0.064299  -0.236  0.81327
## linvent.l4  -0.685847   0.311718  -2.200  0.02856 *
## lsales.l5    0.009242   0.053488   0.173  0.86294
## linvent.l5   0.492424   0.288378   1.708  0.08877 .
## lsales.l6    0.089539   0.057762   1.550  0.12218
```

```
## linvent.l6  -0.092199   0.257717  -0.358  0.72078
## lsales.l7   -0.003089   0.056082  -0.055  0.95612
## linvent.l7  -0.077921   0.266296  -0.293  0.77003
## lsales.l8    0.026519   0.056351   0.471  0.63827
## linvent.l8   0.077590   0.263050   0.295  0.76823
## lsales.l9    0.033385   0.057056   0.585  0.55891
## linvent.l9   0.186820   0.264006   0.708  0.47973
## lsales.l10  -0.149628   0.056617  -2.643  0.00866 **
## linvent.l10 -0.342572   0.265717  -1.289  0.19832
## lsales.l11  -0.065073   0.057295  -1.136  0.25698
## linvent.l11  0.226194   0.271844   0.832  0.40604
## lsales.l12   0.792813   0.059115  13.411  < 2e-16 ***
## linvent.l12 -0.104800   0.271841  -0.386  0.70013
## lsales.l13  -0.520994   0.077841  -6.693 1.09e-10 ***
## linvent.l13  0.340678   0.266409   1.279  0.20198
## lsales.l14   0.200462   0.082124   2.441  0.01523 *
## linvent.l14  0.237014   0.280687   0.844  0.39912
## lsales.l15  -0.256979   0.080354  -3.198  0.00153 **
## linvent.l15 -1.670532   0.277680  -6.016 5.27e-09 ***
## lsales.l16  -0.054165   0.072323  -0.749  0.45450
## linvent.l16  1.150808   0.170056   6.767 7.03e-11 ***
## const        0.109989   0.133871   0.822  0.41197
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.02338 on 296 degrees of freedom
## Multiple R-Squared: 0.9944,  Adjusted R-squared: 0.9938
## F-statistic:  1647 on 32 and 296 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation linvent:
## ========================================
## linvent = lsales.l1 + linvent.l1 + lsales.l2 + linvent.l2 + lsales.l3 + linvent.l3 + lsales.l4 + lin
##
##               Estimate Std. Error t value Pr(>|t|)
## lsales.l1    0.1043019  0.0162050   6.436 4.92e-10 ***
## linvent.l1   1.3004200  0.0572910  22.698  < 2e-16 ***
## lsales.l2    0.0251881  0.0195130   1.291 0.197767
## linvent.l2  -0.3970866  0.0950835  -4.176 3.90e-05 ***
## lsales.l3    0.0323377  0.0197666   1.636 0.102907
## linvent.l3   0.0876531  0.0988919   0.886 0.376147
## lsales.l4   -0.0538717  0.0191955  -2.806 0.005341 **
## linvent.l4  -0.0844202  0.0930583  -0.907 0.365052
## lsales.l5    0.0248304  0.0159679   1.555 0.121010
## linvent.l5   0.1700505  0.0860905   1.975 0.049168 *
## lsales.l6   -0.0326600  0.0172439  -1.894 0.059200 .
## linvent.l6  -0.1494635  0.0769374  -1.943 0.053005 .
## lsales.l7   -0.0416570  0.0167423  -2.488 0.013392 *
## linvent.l7   0.0609838  0.0794984   0.767 0.443630
## lsales.l8   -0.0313104  0.0168227  -1.861 0.063707 .
## linvent.l8  -0.1088234  0.0785293  -1.386 0.166861
## lsales.l9    0.0003706  0.0170331   0.022 0.982658
## linvent.l9   0.1573090  0.0788148   1.996 0.046857 *
```

```
## lsales.l10   0.0196150  0.0169021   1.161 0.246778
## linvent.l10 -0.2665958  0.0793255  -3.361 0.000879 ***
## lsales.l11   0.0964022  0.0171044   5.636 4.05e-08 ***
## linvent.l11  0.3286314  0.0811548   4.049 6.56e-05 ***
## lsales.l12  -0.1209086  0.0176480  -6.851 4.25e-11 ***
## linvent.l12  0.2630658  0.0811536   3.242 0.001324 **
## lsales.l13   0.0025761  0.0232383   0.111 0.911807
## linvent.l13 -0.5252771  0.0795321  -6.605 1.85e-10 ***
## lsales.l14  -0.0120132  0.0245168  -0.490 0.624496
## linvent.l14  0.1526026  0.0837946   1.821 0.069593 .
## lsales.l15   0.0026830  0.0239885   0.112 0.911022
## linvent.l15 -0.1700042  0.0828968  -2.051 0.041167 *
## lsales.l16   0.0055173  0.0215910   0.256 0.798485
## linvent.l16  0.1544866  0.0507675   3.043 0.002552 **
## const        0.0740794  0.0399651   1.854 0.064791 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.006978 on 296 degrees of freedom
## Multiple R-Squared: 0.9992,  Adjusted R-squared: 0.9991
## F-statistic: 1.191e+04 on 32 and 296 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##             lsales    linvent
## lsales   5.464e-04 -6.769e-06
## linvent -6.769e-06  4.870e-05
##
## Correlation matrix of residuals:
##          lsales linvent
## lsales   1.0000 -0.0415
## linvent -0.0415  1.0000
```

Diagram of fit and residuals for lsales

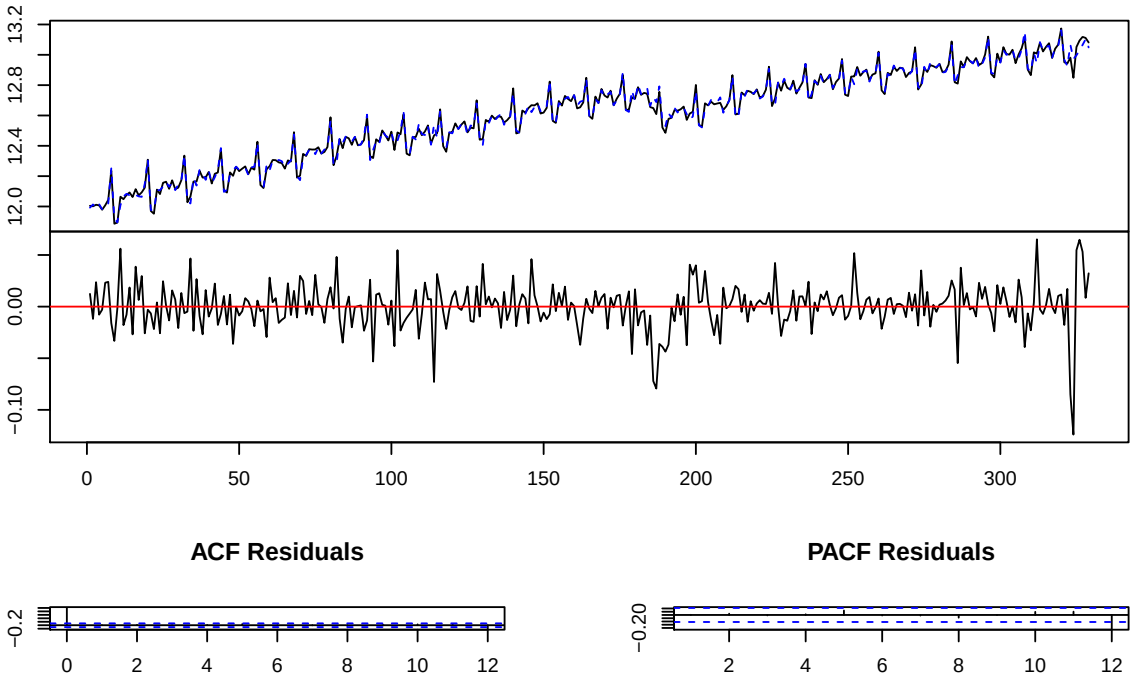
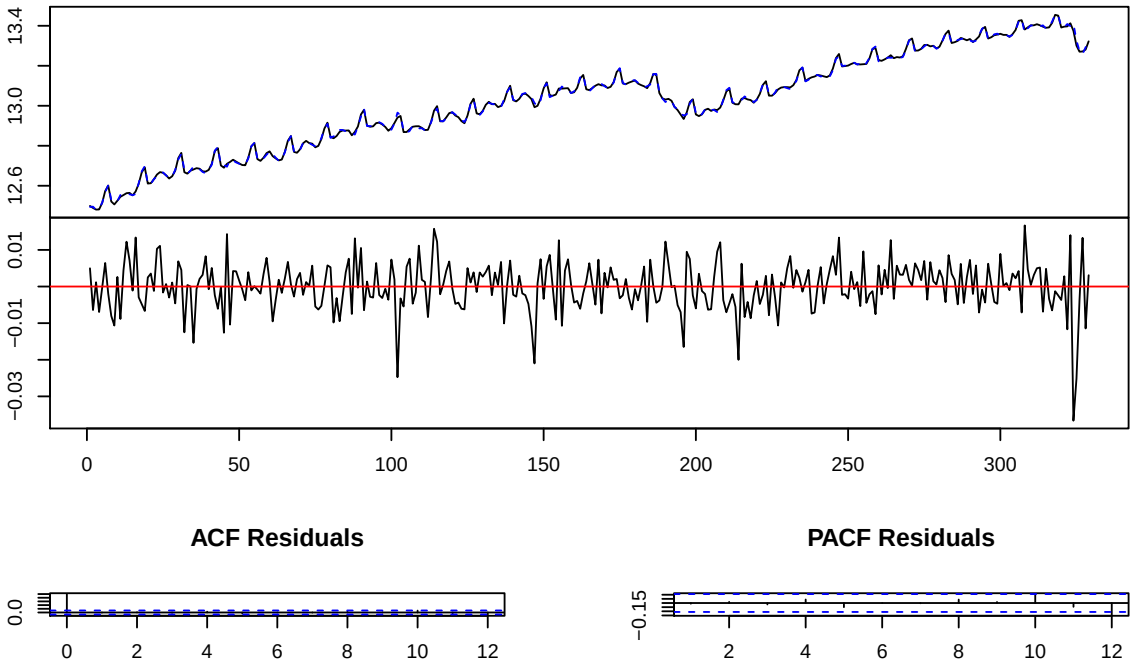
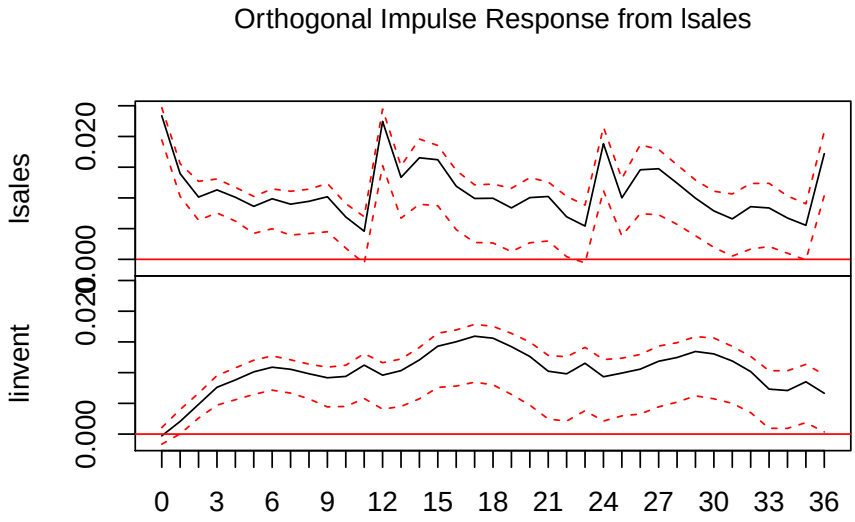


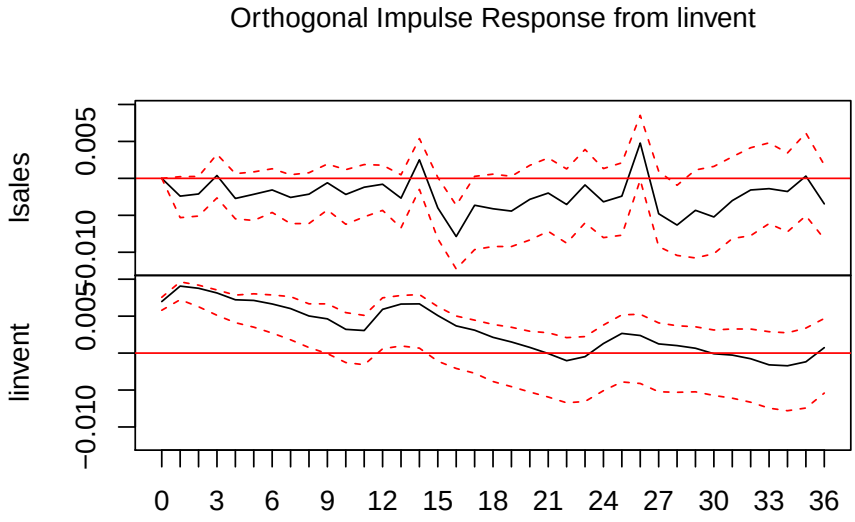
Diagram of fit and residuals for linvent



From the plots, we can find that the fits are quite good. Compared with ARIMA models, there are few spikes in ACF and PACF plots, suggesting that we can do prediction more convincing.

2.11 IRF





For the sales' shock on subsequent sales, there are seasonality shocks on the subsequent years which are

decay slowly, and in each year the shock decay till next year's shock.

For the sales' shock on subsequent inventories, there is no effect at first, then builts up peaking around 17-18 month.

For the inventories' shock on subsequent sales, there is no effect at first but there are small seasonal shcok in the following years.

For the inventories' shock on subsequent on subsequent inventories, there is a large effct at first then decay to zero after 2 years.

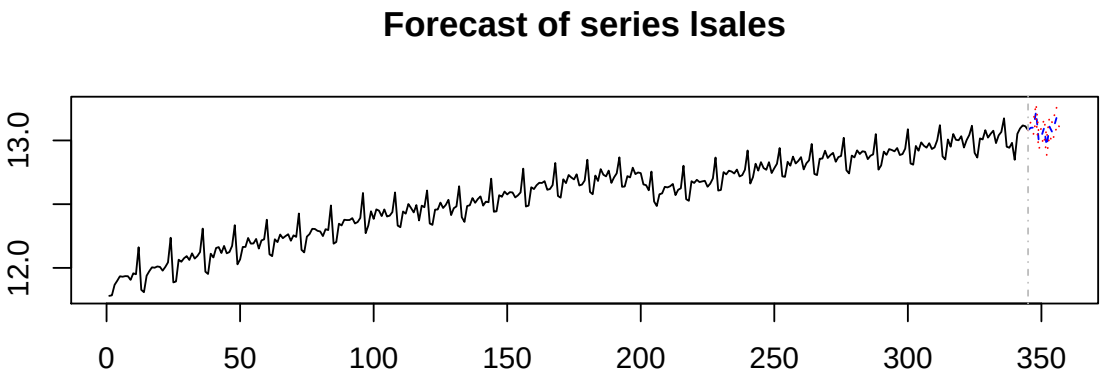
2.12 Granger-Causality Test

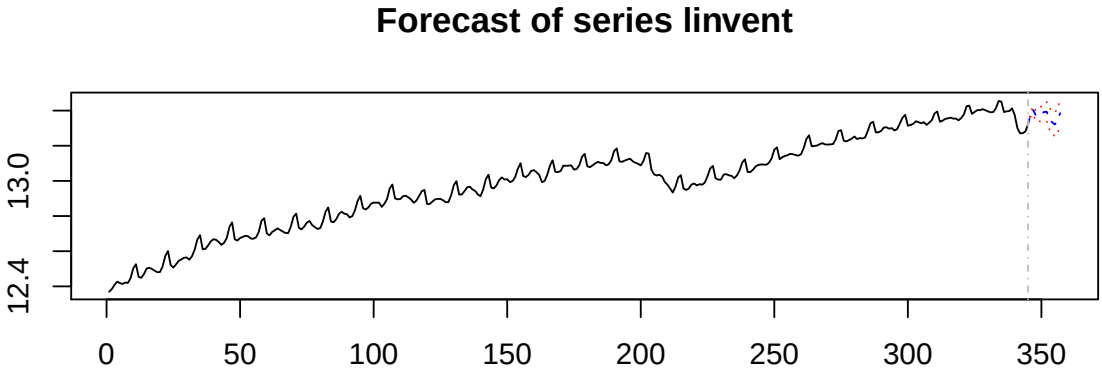
```
## Granger causality test
##
## Model 1: sales ~ Lags(sales, 1:16) + Lags(inventories, 1:16)
## Model 2: sales ~ Lags(sales, 1:16)
##   Res.Df  Df      F    Pr(>F)
## 1    296
## 2    312 -16 7.6465 4.448e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Granger causality test
##
## Model 1: inventories ~ Lags(inventories, 1:16) + Lags(sales, 1:16)
## Model 2: inventories ~ Lags(inventories, 1:16)
##   Res.Df  Df      F    Pr(>F)
## 1    296
## 2    312 -16 19.155 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results of Granger-Causality test suggest that not only retail sales influences inventories but also retail inventories influences sales. The reason why the results are not as my suspection might be the sales in each year have strong seasonality so that they would adjust the inventories ahead of the sales according to the regularity to make more profits.

2.13 Forecast Using VAR





Compared with the forecast of ARIMA model, the predictted values of sales and inventories are more correlated with each other. In ARIMA model, the predicted values are persistent with themselves. We can find that the predicted values of sales using ARIMA are higher than values using VAR and predicted values of inventories using ARIMA are lower than values using VAR. By using VAR, the two time-series looks converging to each other to some degree.

2.14 Backtest ARIMA

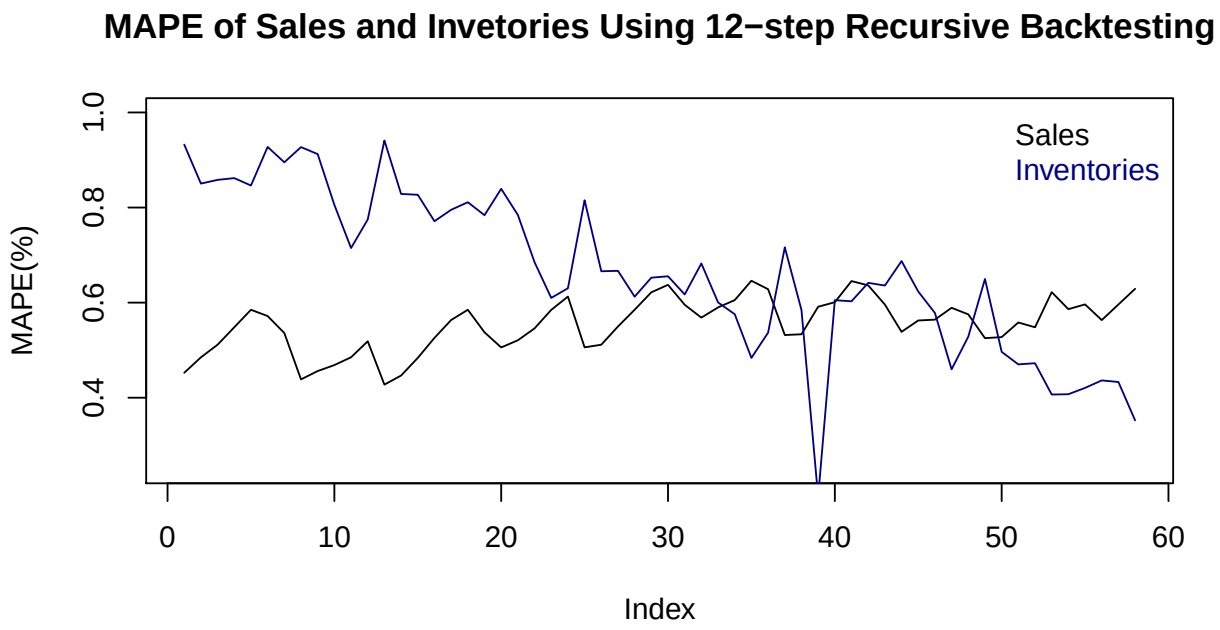
(a) Recursive Backtesting Scheme of 12-steps Ahead Forecast

To do the backtest in recursive scheme or rolling scheme, I write a function to do this and compute respective MAPE.

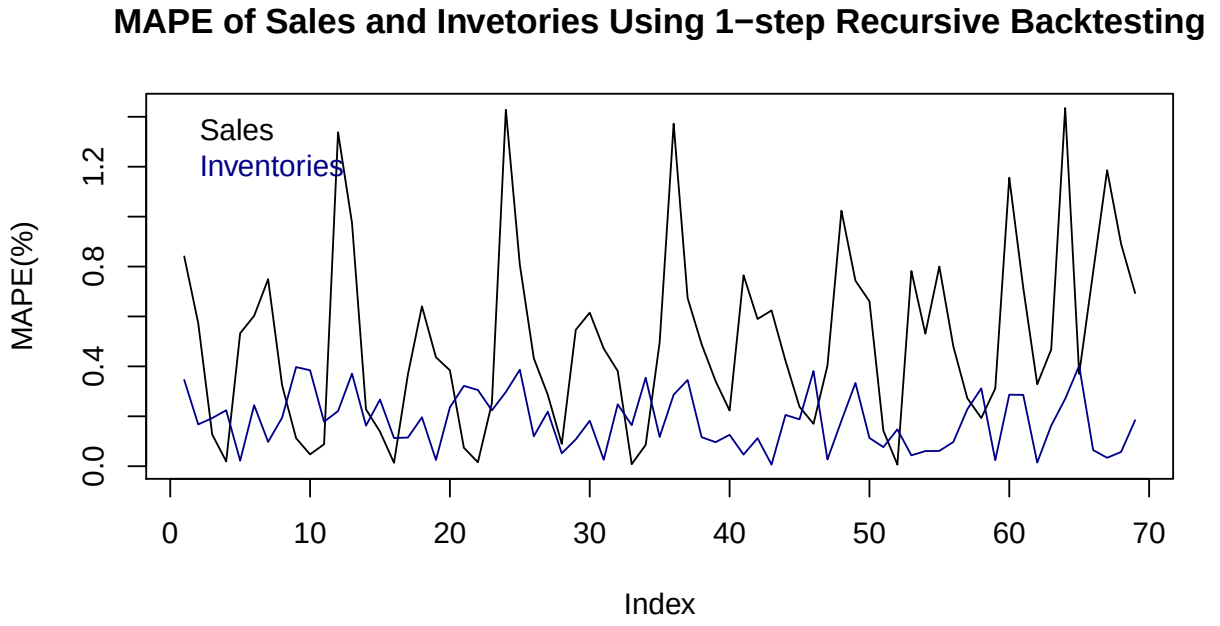
```
compute_mape <- function(data, model_order, t, h, seasonal_order, train_length, isFixed) {
  ##############################
  # input:
  #
  # data: the data we need to model
```

```
  # model_order: chosen arima model order
  # t: date corresponding to data
  # h: forecast horizon
  # seasonal_order: chosen seasonal model order
  # train_length: length of data to train model
  # isFixed: boolean. True: forecast according to adding new data. False: forecast according to moving
  ################################
  # output:
  #
  # MAPE data
  ######################

  length = length(data)
  forecast_length = length - train_length - h + 1
  t2 <- t ^ 2
  result <- rep(0, length = forecast_length)
  xreg_t = cbind(t = t, t2 = t2)

  if (isFixed == TRUE) {
    for (i in 1: forecast_length) {
      begin_i = 1
      end_i = train_length + i - 1
      model <- Arima(data[begin_i: end_i],
                     order = model_order,
                     xreg = xreg_t[begin_i: end_i, ],
                     seasonal = list(order = seasonal_order))
      f_begin_i = train_length + i
      f_end_i = train_length + i + h - 1
      prediction <- forecast(model, h = h, xreg = matrix(xreg_t[f_begin_i: f_end_i, ], ncol = 2))
      actual <- data[f_begin_i: f_end_i]
      result[i] <- 100 * mean(abs((actual - prediction$mean) / actual))
    }
  } else {
    for (i in 1: forecast_length) {
      begin_i = i
      end_i = train_length + i - 1
      model <- Arima(data[begin_i: end_i],
                     order = model_order,
                     xreg = xreg_t[begin_i: end_i, ],
                     seasonal = list(order = seasonal_order))
      f_begin_i = train_length + i
      f_end_i = train_length + i + h - 1
      prediction <- forecast(model, h = h, xreg = matrix(xreg_t[f_begin_i: f_end_i, ], ncol = 2))
      actual <- data[f_begin_i: f_end_i]
      result[i] <- 100 * mean(abs((actual - prediction$mean) / actual))
    }
  }
  return(result)
}
```

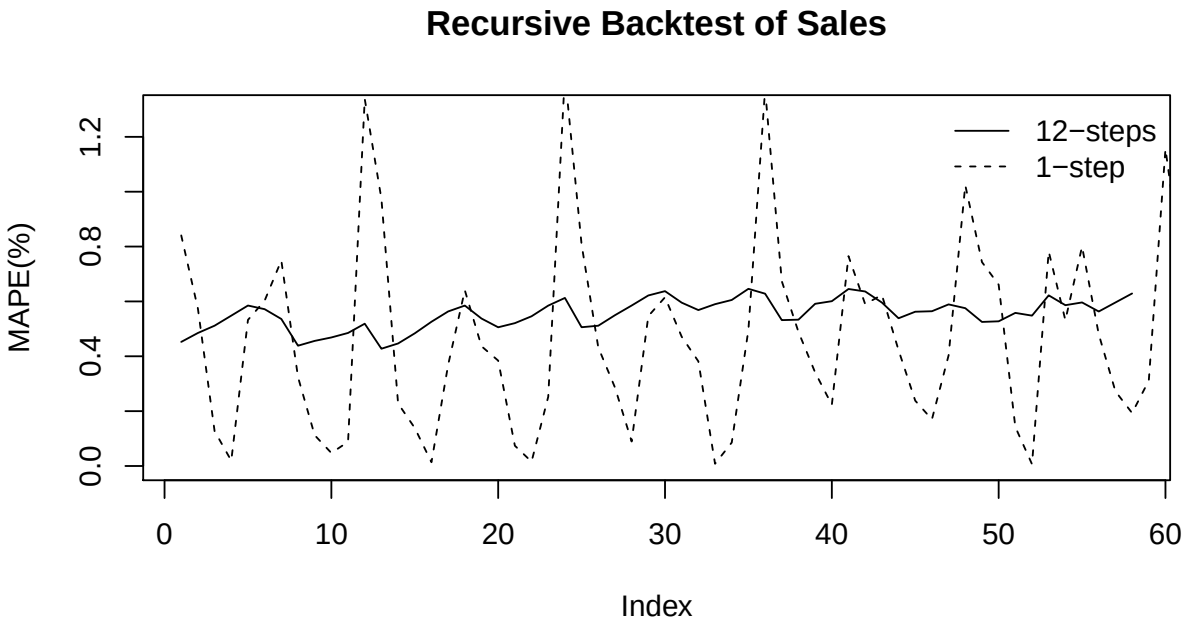
So now we can use the function to do recursive backtest to forecast 12-steps each steps.

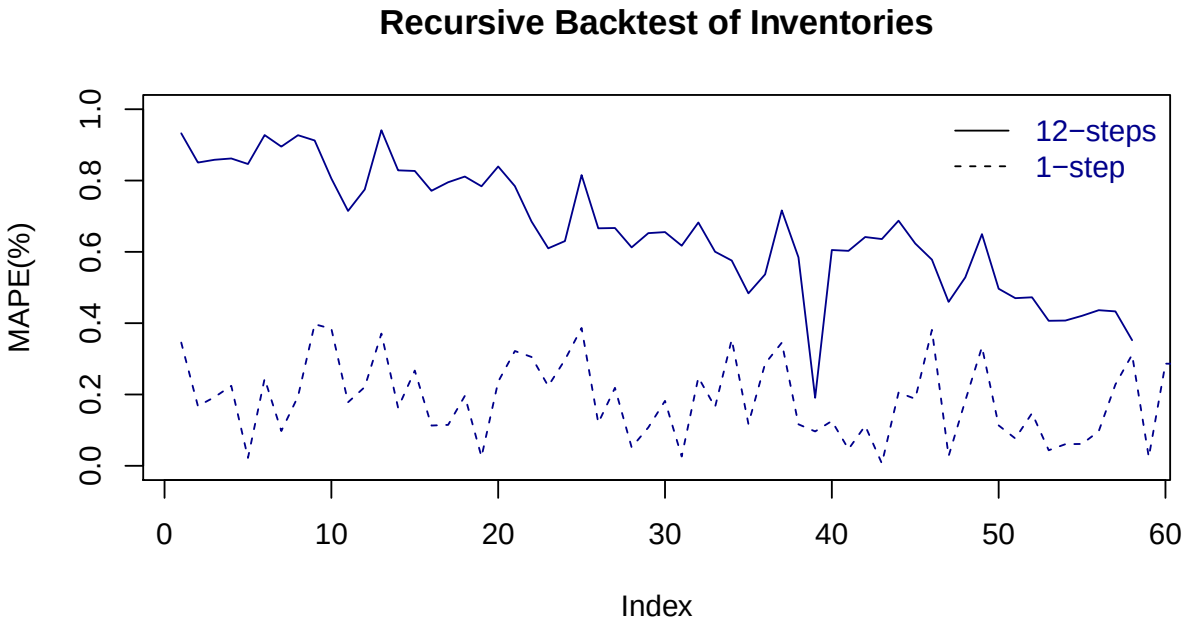


(b) Recursive Backtesting Scheme of 1-step Ahead Forecast



(c) Comparison between Short and Long Horizon



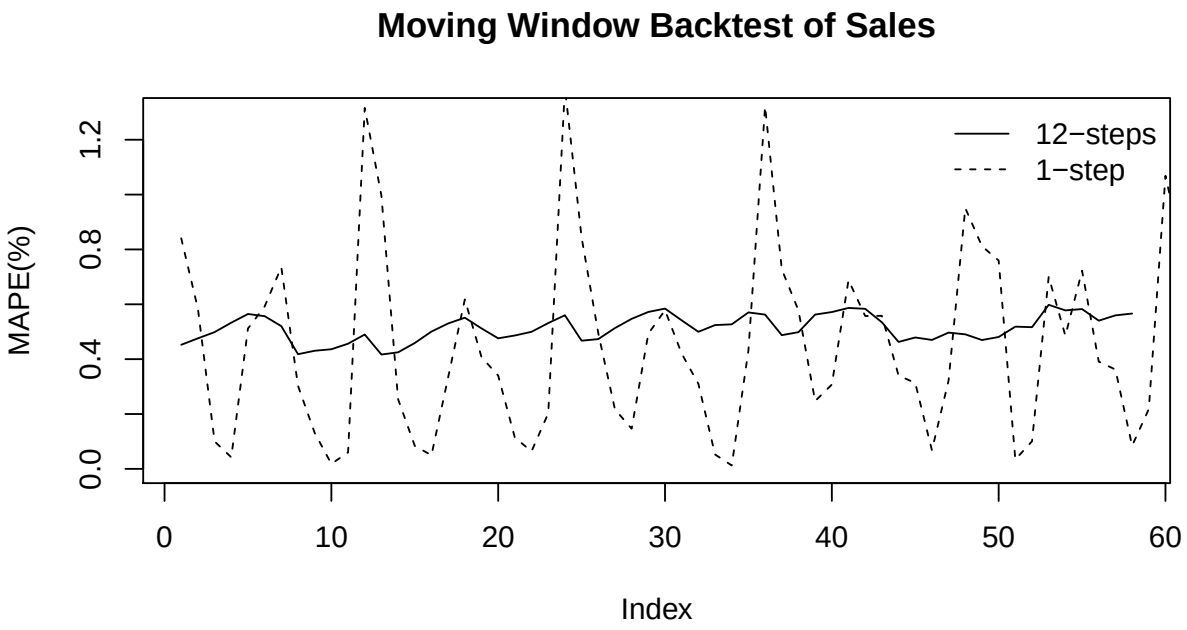


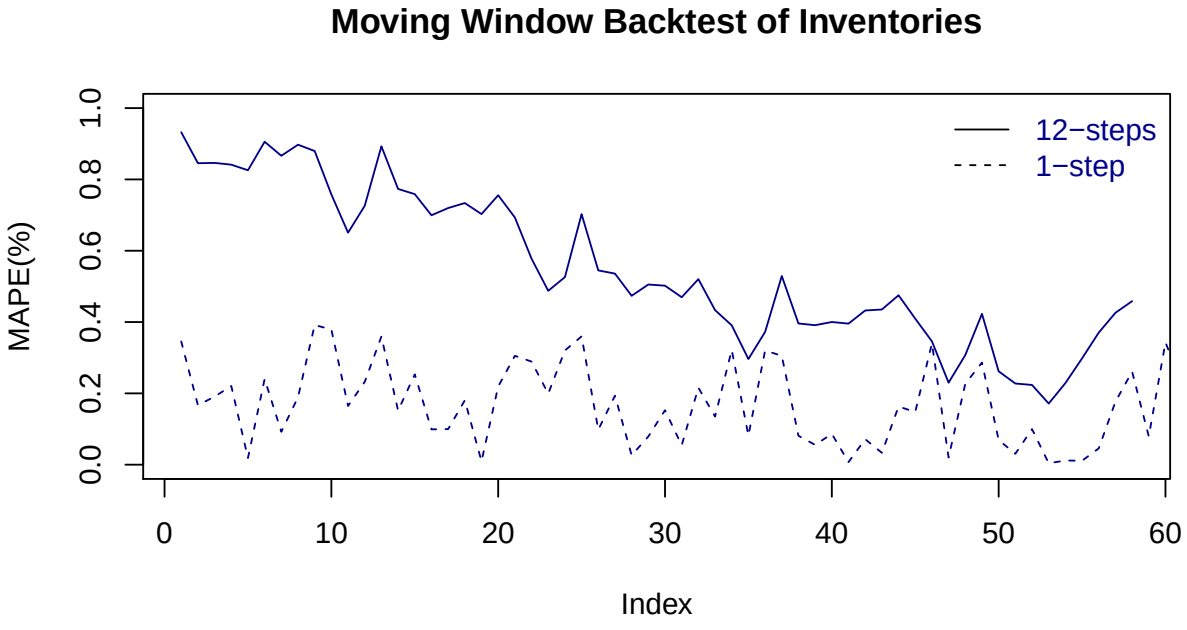
The MAPE of ARIMA of sales using recursive backtesting are small and not greater than 1.3%. For short horizon, MAPE has a quite large variance which have smallest value to near zero but largest to 1.3%. For longer horizon, the MAPE is stable at about 0.5%. Therefore, model of sales is good at forecasting in longer horizon.

The MAPE of inventories are also small and not greater than 1%, indicating our model has great accurary in both long and short horizons. The MAPE of long term horizon have a decreasing trend meaning that larger

sample sizes would lead to more precise prediction. The MAPE of short horizon is stable and small around 0.2%. Therefore, model of inventories is good ar forecasting in shorter horizon.

(d) Moving Window Backtesting

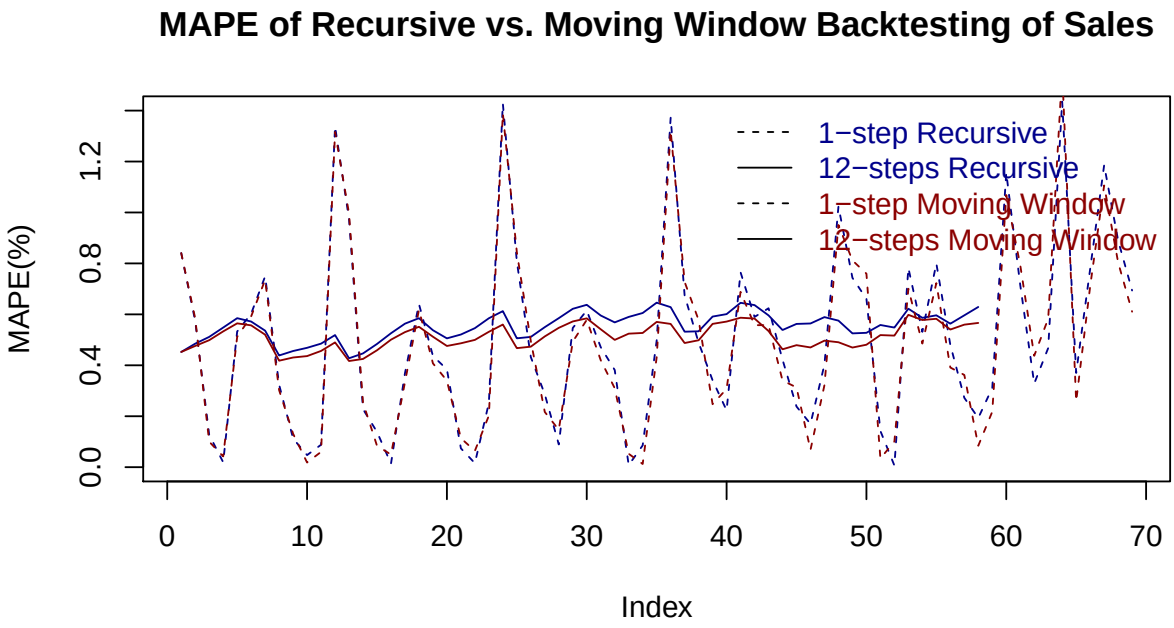


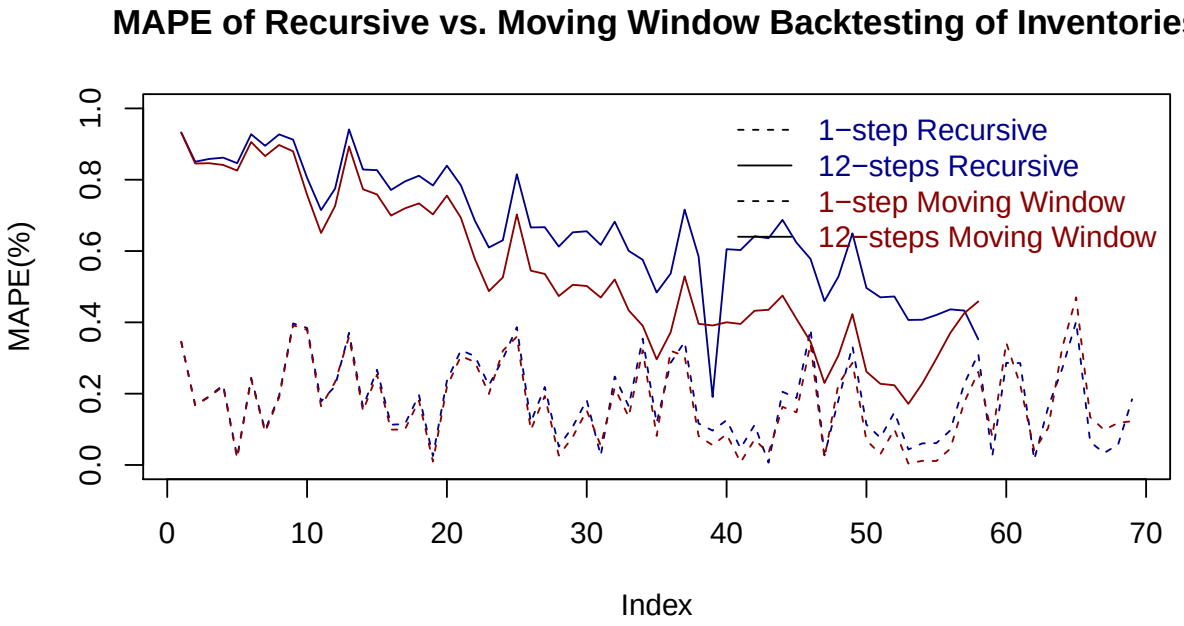


The results are kind of same with the recursive scheme backtesting. The model of sales performs better in long horizon due to the stabilization and the model of inventories performs better in short terms because of small error all the time.

(e) Comparison between Recursive and Moving Window Backtesting Scheme

- Sales





For both sales and inventories data, although the distinction is really small, the moving window backtesting has smaller errors. Our models are more precise on rolling sample means that the data points long time ago would value less for our models.

3. Conclusions and Future Work

To measure and predict the retail sales and inventories data, we can build a time-series model including quadratic trend, seasonality of s-AR(2) model and cycles of AR models. And the two series influence each other so that we can also build a VAR(16) model to estimate and predict both variables simultaneously. As for the model performance, the VAR model do better than ARIMA models because there are still some dynamic left in the residuals of ARIMA models. To further improve the ARIMA models, we can try to fit a variance model such as ARCH and GARCH models to mitigate the dynamics in residuals.

As for the prediction performance, ARIMA model of sales performs better in long-horizon moving window forecasting. ARIMA model of inventories performs better in short-horizon moving window forecasting. This means our model prefer rolling samples that our models are autoregressive but relative short-memory models.

4. Reference

- Professor Rojas's Class Notes and Codes
- Data of retail sales: https://fred.stlouisfed.org/series/RETAILSMNSA
- retail inventories: https://fred.stlouisfed.org/series/RETAILIMNSA
- Debugging of Forecast Function: https://github.com/robjhyndman/forecast/issues/682

5. R Source Code

```r
# clear all variables and prior sessions
rm(list=ls(all=TRUE))

# load libraries
library(forecast)
library(timeSeries)
library(ggplot2)
library(tseries)
library(vars)
library(MTS)

# import data
setwd("C:/Users/Gefei Zhao/Desktop/UCLA/430/Project/2")
df_sales <- read.csv("RETAILSMNSA.csv")
df_inventories <- read.csv("RETAILIMNSA.csv")

# rename the variables in datasets
names(df_sales) <- c("date", "sales")
names(df_inventories) <- c("date", "inventories")

############ [1] TIME-SERIES PLOTS ############
# convert data into time-series format
sales <- ts(df_sales$sales, start = 1992, frequency = 12)
inventories <- ts(df_inventories$inventories, start = 1992, frequency = 12)

# plot time-series plots, ACF and PACF plots
tsdisplay(sales)
tsdisplay(inventories)
```

```r
############ [2] BASILINE MODEL: AUTO ARIMA ############

# use the function to determine order automatically
fit1 <- auto.arima(log(sales))
fit2 <- auto.arima(log(inventories))

summary(fit1)
summary(fit2)

# fitted values vs. real data
autoplot(log(sales)) + autolayer(fitted(fit1)) +
  ggtitle("Log(sales) vs. auto.arima model Fitted Values")

autoplot(log(inventories)) + autolayer(fitted(fit2)) +
  ggtitle("Log(inventories) vs. auto.arima model Fitted Values")

# check the residuals of fits
checkresiduals(fit1)
checkresiduals(fit2)

############ [3] MODEL FITTING ############
# take log to stablize variance
lsales <- log(sales)
linvent <- log(inventories)

# use stl to decompose
lsales %>%
  stl(t.window=13, s.window="periodic", robust=TRUE) %>%
  autoplot() + ggtitle('stl of log(sales)')

linvent %>%
  stl(t.window=13, s.window="periodic", robust=TRUE) %>%
  autoplot() + ggtitle('stl of log(inventories)')

# --------- (a) TREND ---------
t <- seq(1992, 2020.75, length = length(sales))
t2 <- t^2

# sales
logLin_sales <- lm(lsales ~ t)
logQuad_sales <- lm(lsales ~ t + t2)

models_sales <- list(logLinear = logLin_sales, logQuad = logQuad_sales)
summ_sales <- data.frame(sapply(models_sales, function(x) c(AIC(x), BIC(x))))
row.names(summ_sales) <- c("AIC", "BIC")

# inventories
logLin_invent <- lm(linvent ~ t)
logQuad_invent <- lm(linvent ~ t + t2)

models_invent <- list(logLinear = logLin_invent, logQuad = logQuad_invent)
summ_invent <- data.frame(sapply(models_invent, function(x) c(AIC(x), BIC(x))))
row.names(summ_invent) <- c("AIC", "BIC")
```

```r
# stargazer::stargazer(logLin_sales, logQuad_sales)
# stargazer::stargazer(logLin_invent, logQuad_invent)

summ_sales
summ_invent

# --------- (b) SEASONALITY ---------

## fit seasonal lm model for sales
season_sales <- tslm(lsales ~ season + 0)
summary(season_sales)
plot(season_sales$coefficients, type = "l", main = "seasonal Factors for Retail Sales",
     xlab = "Month", ylab = "Seasonal Factor")

## fit seasonal lm model for inventories
season_invent <- tslm(linvent ~ season + 0)
summary(season_invent)
plot(season_invent$coefficients, type = "l", main = "Seasonal Factors for Retail Inventories",
     xlab = "Month", ylab = "Seasonal Factor")

# check the ACF and PACF of the residual of trend models
par(mfrow = c(2,2))
Acf(logQuad_sales$resid, lag = 60)
Pacf(logQuad_sales$resid, lag = 60)

Acf(logQuad_invent$resid, lag = 60)
Pacf(logQuad_invent$resid, lag = 60)

# fit s-arma model for sales
season_sales1 <- arima(logQuad_sales$resid, seasonal = list(order = c(2, 0, 0)))
summary(season_sales1)
ts.plot(logQuad_sales$resid, fitted(season_sales1), gpars = list(col = c("darkblue", "darkred")),
        main = "Fitted values of Seasonality of Sales vs. Residual of Trend Model")

# fit s-arma model for inventories
season_invent1 <- arima(logQuad_invent$resid, seasonal = list(order = c(2, 0, 0)))
summary(season_invent1)
ts.plot(logQuad_invent$resid, fitted(season_invent1), gpars = list(col = c("darkblue", "darkred")),
        main = "Fitted values of Seasonality of Inventories vs. Residual of Trend Model")

# TREND + SEASONALITY
sT_sales1 <- arima(lsales, xreg = cbind(t, t2), seasonal = list(order = c(2, 0, 0)))
sT_invent1 <- arima(linvent, xreg = cbind(t, t2), seasonal = list(order = c(2, 0, 0)))

par(mfrow = c(2,1), mar = c(2, 4, 2, 2), oma = c(2, 2, 2, 2))
ts.plot(lsales, fitted(sT_sales1), gpars = list(col = c("darkblue", "darkred")), ylab = "Log(sales)",
        main = "fitted values for trend + seasonality model vs. log(sales)")
ts.plot(linvent, fitted(sT_invent1), gpars = list(col = c("darkblue", "darkred")),
        ylab = "log(inventories",
        main = "fitted values for trend + seasonality model vs. log(inventories)")

# --------- (c) CYCLE ---------
# check stationary
```

```r
adf.test(sT_sales1$resid, k = 36)
adf.test(sT_invent1$resid, k = 36)

# ACF and PACF plots of residuals of T+S model
par(mfrow = c(2,2))
Acf(sT_sales1$resid, lag = 60)
Pacf(sT_sales1$resid, lag = 60)

Acf(sT_invent1$resid, lag = 60)
Pacf(sT_invent1$resid, lag = 60)

# CYCLE
cycle_sales <- arima(sT_sales1$resid, order = c(3, 0, 0))
cycle_invent <- arima(sT_invent1$resid, order = c(2, 0, 0))

summary(cycle_sales)
summary(cycle_invent)

par(mfrow = c(2,1), mar = c(2, 4, 2, 2), oma = c(2, 2, 2, 2))
ts.plot(sT_sales1$resid, fitted(cycle_sales), gpars = list(col = c("darkblue", "darkred")),
        main = "Fitted values of cycles of sales vs residuals of T+S model")
ts.plot(sT_invent1$resid, fitted(cycle_invent), gpars = list(col = c("darkblue", "darkred")),
        main = "Fitted values of cycles of inventories vs residuals of T+S model")

# --------- (d) FULL MODEL ---------
# TREND + SEASONALITY + CYCLE
tsc_sales <- Arima(lsales, order = c(3, 0, 0), xreg = cbind(t, t2), seasonal = list(order = c(2, 0, 0)))
tsc_invent <- Arima(linvent, order = c(2, 0, 0), xreg = cbind(t, t2), seasonal = list(order = c(2, 0, 0


############ [3] MODEL DIAGNOSIS ############
# residuals vs fitted values
par(mfrow = c(2, 1), mar = c(2, 4, 2, 2), oma = c(2, 2, 2, 2))
ts.plot(lsales, fitted(tsc_sales), gpars = list(col = c("darkblue", "darkred")),
        main = "Fitted values vs. residuals")
plot(tsc_sales$resid, ylab = "residuals")

ts.plot(linvent, fitted(tsc_invent), gpars = list(col = c("darkblue", "darkred")),
        main = "Fitted values vs. residuals")
plot(tsc_invent$resid, ylab = "residuals")

# ACF and PACF of residuals of full model
par(mfrow = c(2,2))
Acf(tsc_sales$resid, lag = 60)
Pacf(tsc_sales$resid, lag = 60)

Acf(tsc_invent$resid, lag = 60)
Pacf(tsc_invent$resid, lag = 60)

# CUSUM
y_sales <- strucchange::recresid(tsc_sales$resid ~ 1)
y_invent <- strucchange::recresid(tsc_invent$resid ~ 1)
```

```r
plot(strucchange::efp(tsc_sales$resid ~ 1, type = "Rec-CUSUM"), main = "CUSUM of Model for Sales")
plot(strucchange::efp(tsc_invent$resid ~ 1, type = "Rec-CUSUM"), main = "CUSUM of Model for Inventories

# recursive residuals
plot(y_sales, pch = 16, ylab = "Recursive Residuals",
     main = "Recursive Residuals of Model for Sales")
plot(y_invent, pch = 16, ylab = "Recursive Residuals",
     main = "Recursive Residuals of Model for Inventories")

summary(tsc_sales)
summary(tsc_invent)

############ [4] FORCAST USING FULL MODEL ############
tf <- seq(2020.8, 2021.75, length = 12)
tf2 <- tf^2

forecast(tsc_sales, xreg = cbind(t = tf, t2 = tf2), h = 12) %>% plot()
forecast(tsc_invent, xreg = cbind(t = tf, t2 = tf2), h = 12) %>% plot()

############ [5] VAR MODEL ############
ts.plot(sales, inventories, gpars = list(col = c("darkblue", "darkred")),
        main = "Retail Sales and Retail Inventories")
legend("topleft",legend = c("Sales","Inventories"), text.col = c("darkblue", "darkred"), bty = "n")

# ccf
ccf(lsales, linvent, ylab="Cross-Correlation Function", main = "Sales and Inventories CCF")

lsi <- data.frame(cbind(lsales, linvent))
VARselect(lsi, 20) # select var with smallest information criteria

var_mod <- vars::VAR(lsi, p = 16)
summary(var_mod)

plot(var_mod)

# irf
plot(irf(var_mod, n.ahead=36))

# granger-causality test
grangertest(sales ~ inventories, order = 16)
grangertest(inventories ~ sales, order = 16)

# forecast using VAR
var_predict = predict(object = var_mod, n.ahead = 12)
plot(var_predict)

############ [6] BACKTEST ARIMA ############
compute_mape <- function(data, model_order, t, h, seasonal_order, train_length, isFixed) {
  ################################
  # input:
  #
  # data: the data we need to model
  # model_order: chosen arima model order
```

```
  # t: date corresponding to data
  # h: forecast horizon
  # seasonal_order: chosen seasonal model order
  # train_length: length of data to train model
  # isFixed: boolean. True: forecast according to adding new data. False: forecast according to moving
  ################################
  # output:
  #
  # MAPE data
  ######################

  length = length(data)
  forecast_length = length - train_length - h + 1
  t2 <- t ^ 2
  result <- rep(0, length = forecast_length)
  xreg_t = cbind(t = t, t2 = t2)

  if (isFixed == TRUE) {
    for (i in 1: forecast_length) {
      begin_i = 1
      end_i = train_length + i - 1
      model <- Arima(data[begin_i: end_i],
                     order = model_order,
                     xreg = xreg_t[begin_i: end_i, ],
                     seasonal = list(order = seasonal_order))
      f_begin_i = train_length + i
      f_end_i = train_length + i + h - 1
      prediction <- forecast(model, h = h, xreg = matrix(xreg_t[f_begin_i: f_end_i, ], ncol = 2))
      actual <- data[f_begin_i: f_end_i]
      result[i] <- 100 * mean(abs((actual - prediction$mean) / actual))
    }
  } else {
    for (i in 1: forecast_length) {
      begin_i = i
      end_i = train_length + i - 1
      model <- Arima(data[begin_i: end_i],
                     order = model_order,
                     xreg = xreg_t[begin_i: end_i, ],
                     seasonal = list(order = seasonal_order))
      f_begin_i = train_length + i
      f_end_i = train_length + i + h - 1
      prediction <- forecast(model, h = h, xreg = matrix(xreg_t[f_begin_i: f_end_i, ], ncol = 2))
      actual <- data[f_begin_i: f_end_i]
      result[i] <- 100 * mean(abs((actual - prediction$mean) / actual))
    }
  }
  return(result)
}

# recursive backtesting 12-steps
MAPE_sales12 <- compute_mape(lsales, model_order = c(3, 0, 0), t = t, h = 12,
                             seasonal_order = c(2, 0, 0),   train_length = 276, isFixed = TRUE)
```

```r
MAPE_invent12 <- compute_mape(linvent, model_order = c(2, 0, 0), t = t, h = 12,
                              seasonal_order = c(2, 0, 0), train_length = 276, isFixed = TRUE)


plot(MAPE_sales12, type = "l", , ylab = "MAPE(%)", ylim = c(0.25, 1),
     main = "MAPE of Sales and Invetories Using 12-step Recursive Backtesting")
lines(MAPE_invent12, type = "l", col = "darkblue")
legend("topright",legend = c("Sales","Inventories"), text.col = c("black", "darkblue"), bty = "n")

# recursive backtesting 1-step
MAPE_sales1 <- compute_mape(lsales, model_order = c(3, 0, 0), t = t, h = 1,
                            seasonal_order = c(2, 0, 0), train_length = 276, isFixed = TRUE)

MAPE_invent1 <- compute_mape(linvent, model_order = c(2, 0, 0), t = t, h = 1,
                              seasonal_order = c(2, 0, 0), train_length = 276, isFixed = TRUE)

plot(MAPE_sales1, type = "l", ylab = "MAPE(%)",
     main = "MAPE of Sales and Invetories Using 1-step Recursive Backtesting")
lines(MAPE_invent1, type = "l", col = "darkblue")
legend("topleft",legend = c("Sales","Inventories"), text.col = c("black", "darkblue"), bty = "n")

# compare between long and short horizon
plot(MAPE_sales12, type = "l", ylim = c(0, 1.3), main = "Recursive Backtest of Sales",
     ylab = "MAPE(%)")
lines(MAPE_sales1, type = "l", lty = 2)
legend("topright",legend = c("12-steps","1-step"), text.col = c("black", "black"),
       lty = c(1, 2), bty = "n")

plot(MAPE_invent12, type = "l", col = "darkblue", ylim = c(0,1),
     main = "Recursive Backtest of Sales", ylab = "MAPE(%)")
lines(MAPE_invent1, type = "l", col = "darkblue", lty = 2)
legend("topright",legend = c("12-steps","1-step"), text.col = c("darkblue", "darkblue"),
       lty = c(1, 2), bty = "n")


# moving window backtesting
MAPE_sales_r12 <- compute_mape(lsales, model_order = c(3, 0, 0), t = t, h = 12,
                               seasonal_order = c(2, 0, 0),   train_length = 276, isFixed = FALSE)

MAPE_invent_r12 <- compute_mape(linvent, model_order = c(2, 0, 0), t = t, h = 12,
                              seasonal_order = c(2, 0, 0), train_length = 276, isFixed = FALSE)

MAPE_sales_r1 <- compute_mape(lsales, model_order = c(3, 0, 0), t = t, h = 1,
                            seasonal_order = c(2, 0, 0), train_length = 276, isFixed = FALSE)

MAPE_invent_r1 <- compute_mape(linvent, model_order = c(2, 0, 0), t = t, h = 1,
                              seasonal_order = c(2, 0, 0), train_length = 276, isFixed = FALSE)

plot(MAPE_sales_r12, type = "l", ylim = c(0, 1.3), main = "Moving Window Backtest of Sales",
     ylab = "MAPE(%)")
lines(MAPE_sales_r1, type = "l", lty = 2)
legend("topright",legend = c("12-steps","1-step"), text.col = c("black", "black"),
       lty = c(1, 2), bty = "n")
```

```r
plot(MAPE_invent_r12, type = "l", col = "darkblue", ylim = c(0,1),
     main = "Moving Window Backtest of Sales", ylab = "MAPE(%)")
lines(MAPE_invent_r1, type = "l", col = "darkblue", lty = 2)
legend("topright",legend = c("12-steps","1-step"), text.col = c("darkblue", "darkblue"),
       lty = c(1, 2), bty = "n")


# compare rolling and recursive backtesting
# all MAPE plots of sales
plot(MAPE_sales1, type = "l", col = "blue4", ylab = "MAPE(%)", ylim = c(0,1.4),
     main = "MAPE of Recursive vs. Moving Window Backtesting of Sales", lty = 2)
lines(MAPE_sales12, type = "l", col = "blue4")
lines(MAPE_sales_r1, type = "l", col = "red4", lty = 2)
lines(MAPE_sales_r12, type = "l", col = "red4")
legend("topright",legend = c("1-step Recursive","12-steps Recursive",
                             "1-step Moving Window","12-steps Moving Window"),
       text.col = c("blue4", "blue4", "red4", "red4"), lty = c(2, 1, 2, 1), bty = "n")

# all MAPE plots of inventories
plot(MAPE_invent1, type = "l", col = "blue4", ylab = "MAPE(%)", ylim = c(0,1),
     main = "MAPE of Recursive vs. Moving Window Backtesting of Inventories", lty = 2)
lines(MAPE_invent12, type = "l", col = "blue4")
lines(MAPE_invent_r1, type = "l", col = "red4", lty = 2)
lines(MAPE_invent_r12, type = "l", col = "red4")
legend("topright",legend = c("1-step Recursive","12-steps Recursive", "1-step Moving Window",
                             "12-steps Moving Window"),
       text.col = c("blue4", "blue4", "red4", "red4"), lty = c(2, 1, 2, 1), bty = "n")
```