Nov 29 Notes

Hongdou Li

11/29/2018

Multivariate Time Series

Unitl now, we've only considered **univariate** time series, i.e, we have temporal observations on one variable which use to forecast that variable.

Now, imagine you have data on other variables collected at the same frequency and the same duration as your response series. Furthermore, if these extra variables are highly correlated with the response, we may exploit to incorporate their observation into a **multivariate** time series model. We hope that such a model provides more accurate forecasts than some other univariate model.

Depending on how we want to treat this external information determines which modeling approach to take:

- If we treat these variables as **exogenous**. i.e, they influence the response, but not the other way around, We can fit a *SARIMAX* model to account for this type of relationship

EX: Daily BART ridership and daily weather

- If we treat these variables as **endodenous**. i.e, they influence the response and the response influences them, then we can use vactor autoregression (VAR) to simultaneously account for all of these dependencies.

EX: Daily closing price of APPLE atock and daily closing price of Amazon stock.

SARIMAX MODEL

A SARIMAX model is thought of as a SARIMA with exponentory variables. To begin consider an ARMAX(p,q) model:

$$Y_t = \phi_1 Y_{t-1} + ... + \phi_p Y_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + ... + \theta_q \epsilon_{t-q} + \beta_1 X_{1,t} + \beta_2 X_{2,t} + ... + \beta_r X_{r,t}$$
$$= \sum_{i=1}^{p} \phi_i Y_{t-i} + \epsilon_t + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \sum_{k=1}^{r} \beta_k X_{k,t}$$

where $\{X_{k,t}\}$ is an explantory, "exogenous", time series, k=1,2,...,r

If $\{Y_t\}$ is not stationary, we difference it as necessary to become stationary. The same level of differencing is also automatically applied to the exogenous series. This gives rise to ARIMAX and SARIMAX models.

$\sum_{k=1}^{r} \beta_k X_{k,t} <= Corr(Y_t, X_{k,t+h}, h \in Z$ is called cross correlation, and can be useful in determining how the X information is used.

* In order to forecast a SARIMAX model we need future values of the exogenous variables, or predictions of them.

The main limitation of the SARIMAX model is that it cannot account for a bi-directional relationship if one exists. In this situation a vector autoregression would be more appropriate, and may provide more accurate forecasts.

Vector Autoregression

In this framework all variables are treated symmetrically and so we revise our notation and denote r endongenous variables as :$\{Y_{1,t}\}, \{Y_{2,t}\}, ...\{Y_{r,t}\}$. This model consists of r equations (one for each variable) that are each autoregressions of order p.

VAR(p):

$Y_{1,t} = c_1 \sum_{i=1}^{p} \phi_{11,i} Y_{1,t-i} + \sum_{i=1}^{p} \phi_{12,i} Y_{2,t-i} + ... + \sum_{i=1}^{p} \phi_{1r,i} Y_{r,t-i} + \epsilon_{1,t}$

.

.

.

$Y_{r,t} = c_1 \sum_{i=1}^{p} \phi_{r1,i} Y_{1,t-i} + \sum_{i=1}^{p} \phi_{r2,i} Y_{1,t-i} + ... + \sum_{i=1}^{p} \phi_{rr,i} Y_{r,t-i} + \epsilon_{r,t}$

where $\{\epsilon_{k,t}\} \sim WN(0, \sigma_k^2$

def $\overrightarrow{Y_t} = \begin{bmatrix} Y_{1,t} \\ Y_{2,t} \\ . \\ . \\ . \\ Y_{r,t} \end{bmatrix}$ $\overrightarrow{c} = \begin{bmatrix} c_1 \\ c_2 \\ . \\ . \\ . \\ c_r \end{bmatrix}$ $\overrightarrow{\epsilon_t} = \begin{bmatrix} \epsilon_{1,t} \\ \epsilon_{2,t} \\ . \\ . \\ . \\ \epsilon_{r,t} \end{bmatrix}$

$A_i = \begin{bmatrix} \phi_{11,i} & \phi_{12,i} & \cdots & \phi_{1r,i} \\ \phi_{21,i} & \phi_{22,i} & \cdots & \phi_{2r,i} \\ . & & & \\ . & & & \\ . & & & \\ \phi_{r1,i} & \phi_{r2,i} & \cdots & \phi_{rr,i} \end{bmatrix}$

Using this vector-matrix notation we can equivalently write a VAR(p) model more succinctly as:

$\overrightarrow{Y_t} = \overrightarrow{c} + a_1 \overrightarrow{Y_{t-1}} + A_2 \overrightarrow{Y_{t-2}} + ... + A_p \overrightarrow{Y_{t-p}} + \overrightarrow{\epsilon_t}$

EX: VAR(1) with two variables

$\overrightarrow{Y_{1,t}} = c_1 + \phi_{11,1} \overrightarrow{Y_{1,t-1}} + \phi_{12,1} \overrightarrow{Y_{2,t-1}} + \overrightarrow{\epsilon_{1,t}}$

$\overrightarrow{Y_{2,t}} = c_2 + \phi_{21,1} \overrightarrow{Y_{1,t-1}} + \phi_{22,1} \overrightarrow{Y_{2,t-1}} + \overrightarrow{\epsilon_{2,t}}$

$\overrightarrow{Y_t} = \begin{bmatrix} Y_{1,t} \\ Y_{2,t} \end{bmatrix}$ $\overrightarrow{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$ $\overrightarrow{\epsilon_t} = \begin{bmatrix} \epsilon_{1,t} \\ \epsilon_{2,t} \end{bmatrix}$ $A = \begin{bmatrix} \phi_{11,1} & \phi_{12,1} \\ \phi_{21,1} & \phi_{22,1} \end{bmatrix}$ $\overrightarrow{Y_{t-1}} = \begin{bmatrix} Y_{1,t-1} \\ Y_{2,t-1} \end{bmatrix}$

The general VAR(p) model contains $2r + pr^2$ parameters. To simplify estimation avoid overfitting we typically try to keep r and/or p small. Order selection in this is based on predictive accuracy and/or goodness-of-fit. Estimation is typically carried out with leadt squares.

* VARMA models exist, but are not commonly used in practice

* VARX models are commonly used when you have exofenous and endogenous variables

```r
library(forecast)
library(vars)
```

```
## Loading required package: MASS
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

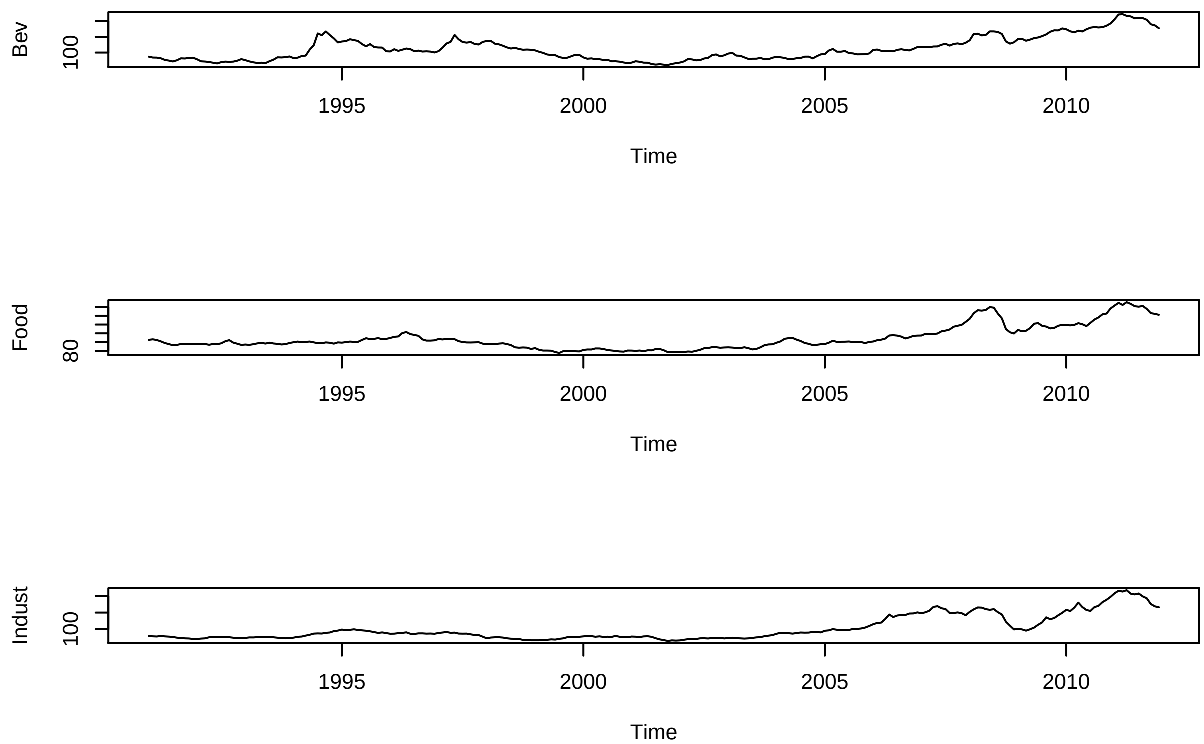
```
## Loading required package: sandwich

## Loading required package: urca

## Loading required package: lmtest
```

```r
data <- read.csv("consumer.csv", header=T)
head(data)
```

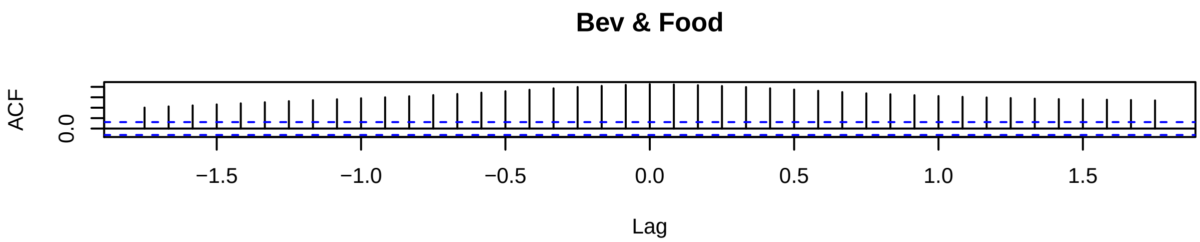
```
##       Date FoodInd BevInd IndustInd
## 1 1991M01  105.18  87.10     79.30
## 2 1991M02  106.44  84.19     78.51
## 3 1991M03  104.77  83.92     77.95
## 4 1991M04  101.73  81.81     79.59
## 5 1991M05   97.97  76.69     78.38
## 6 1991M06   95.54  74.53     77.58
```

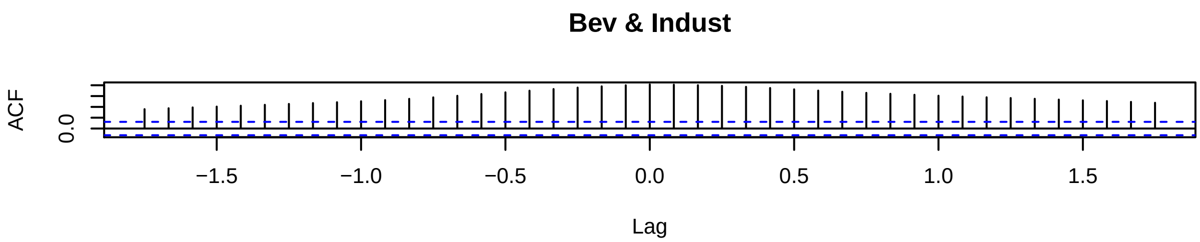
```r
train <- data[1:252,]
test <- data[253:255,]
Bev <- ts(train$BevInd, start = c(1991, 1), frequency = 12)
Food <- ts(train$FoodInd, start = c(1991, 1), frequency = 12)
Indust <- ts(train$IndustInd, start = c(1991, 1), frequency = 12)
```

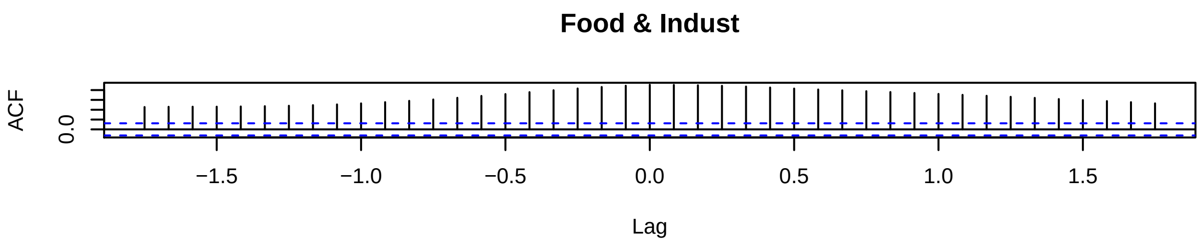
```r
par(mfrow=c(3,1))
plot(Bev)
plot(Food)
plot(Indust)
```



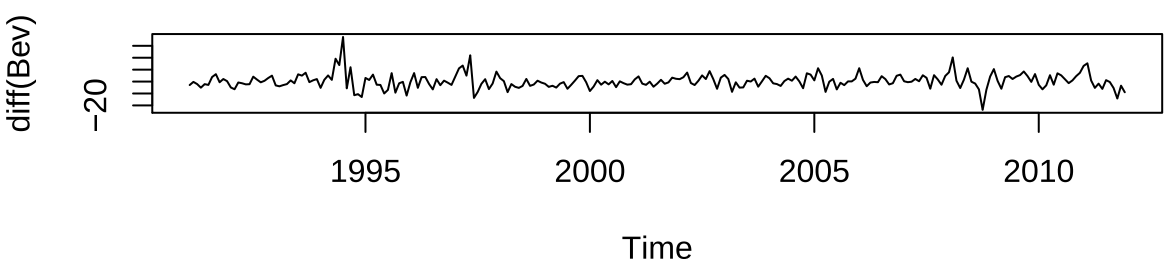

```r
par(mfrow=c(3,1))
ccf(Bev, Food)
ccf(Bev, Indust)
ccf(Food, Indust)
```

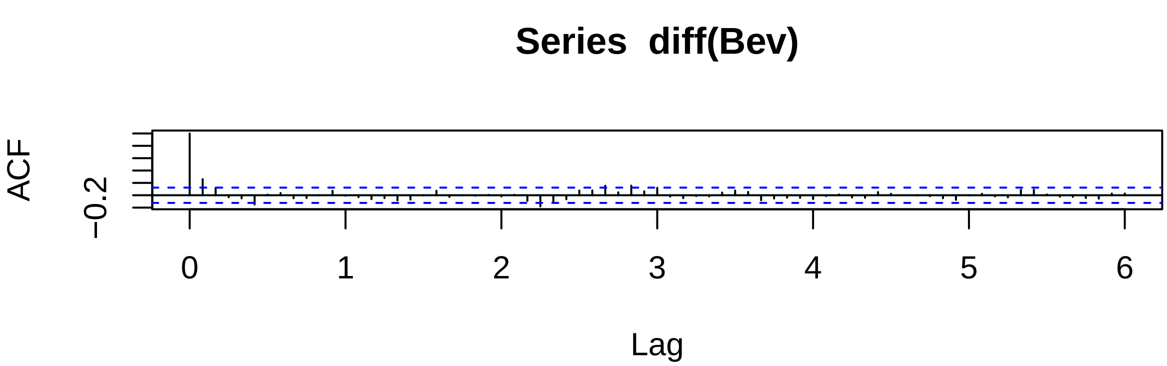




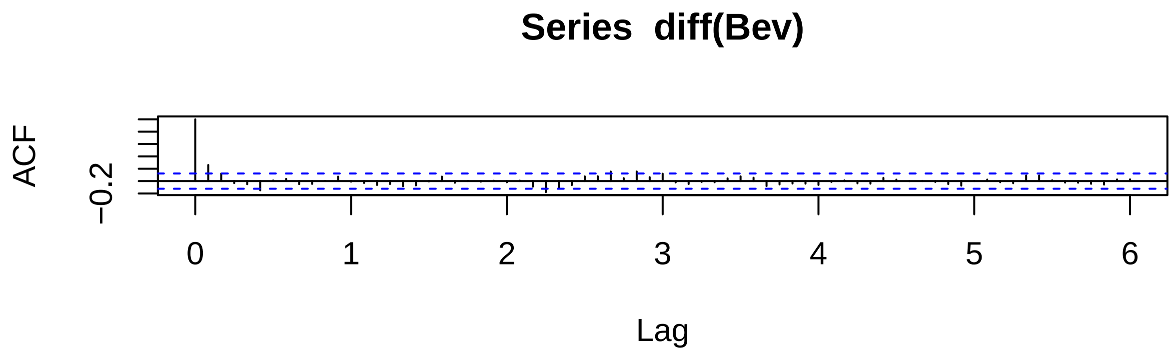


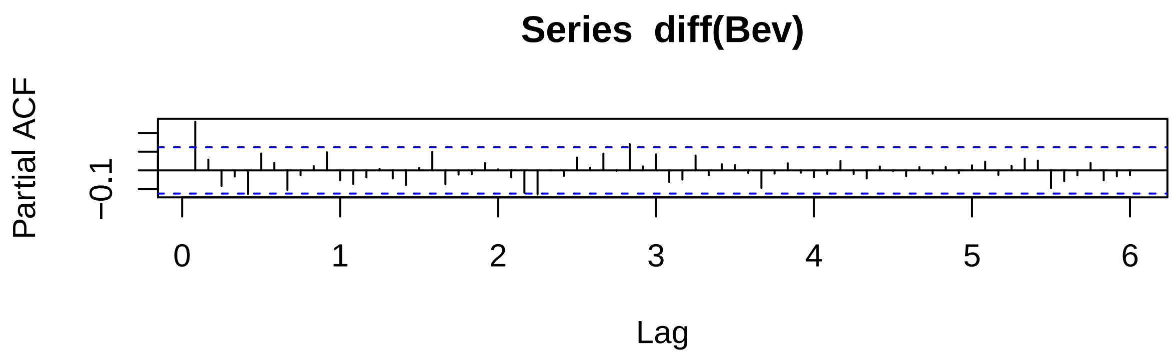

```
# ordinary difference:
par(mfrow=c(2,1))
plot(diff(Bev))
acf(diff(Bev), lag.max = 72)
```





```r
# order selection:
par(mfrow=c(2,1))
acf(diff(Bev), lag.max = 72)
pacf(diff(Bev), lag.max = 72)
```



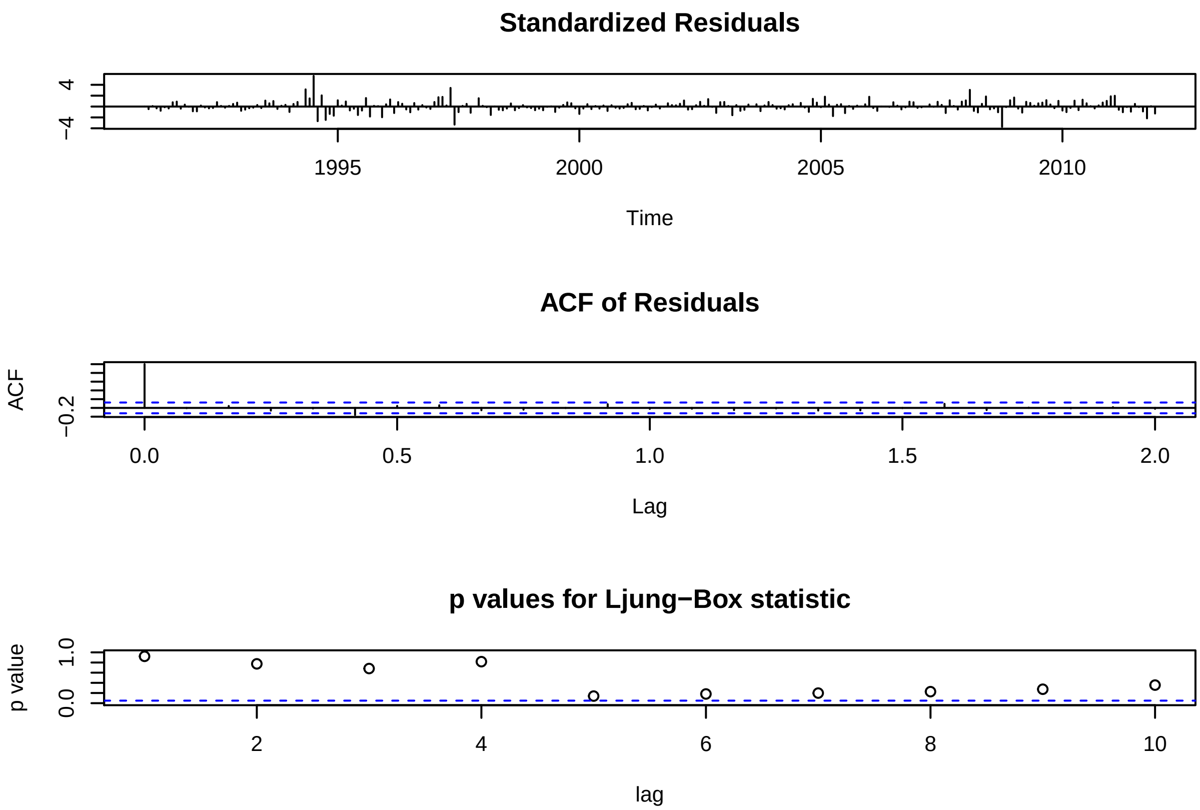



```r
#p=q=1 seems fine
```

```r
# Fit an ARIMA(1,1,1) model
m1 <- arima(Bev, order = c(1,1,1))
m1
```

```
##
## Call:
## arima(x = Bev, order = c(1, 1, 1))
##
## Coefficients:
##          ar1      ma1
##       0.3932  -0.1363
## s.e.  0.1667   0.1748
##
## sigma^2 estimated as 34.88:  log likelihood = -801.97,  aic = 1609.94
```

```r
tsdiag(m1)
```



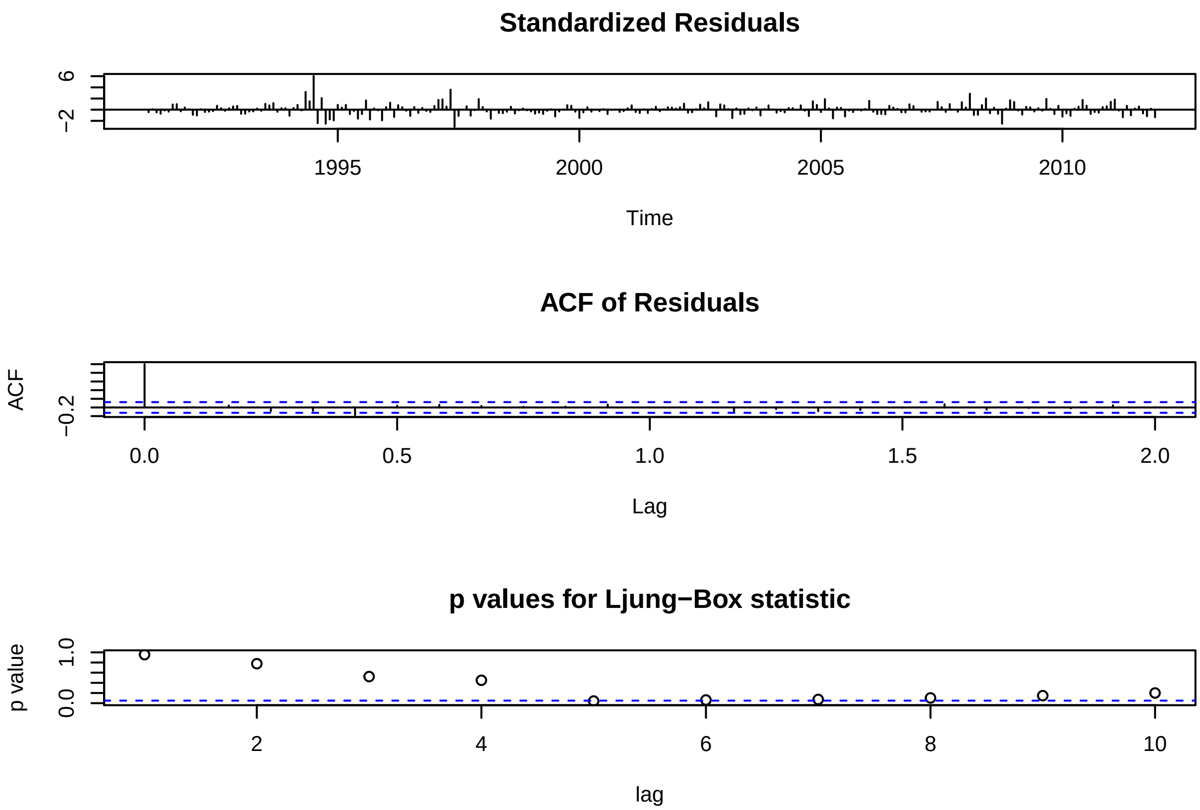
ACF: $H_0 : \rho(0) = 0$ vs. $H_a : \rho(h) \neq 0$

Ljung-Box: $H_0 : \rho(1) = \rho(2) = ... = \rho(H) = 0$ vs. $H_A : \rho(h) \neq 0$ for same h=1,2. . . .H

```r
# Fit an ARIMAX(1,1,1) model with covariate information
m2 <- arima(Bev, order = c(1,1,1), xreg = data.frame(Indust))
m2
```

```
##
## Call:
## arima(x = Bev, order = c(1, 1, 1), xreg = data.frame(Indust))
##
## Coefficients:
##          ar1      ma1  Indust
##       0.2951  -0.0820  0.4042
## s.e.  0.1911   0.1932  0.0863
##
## sigma^2 estimated as 32.12:  log likelihood = -791.61,  aic = 1591.23
```

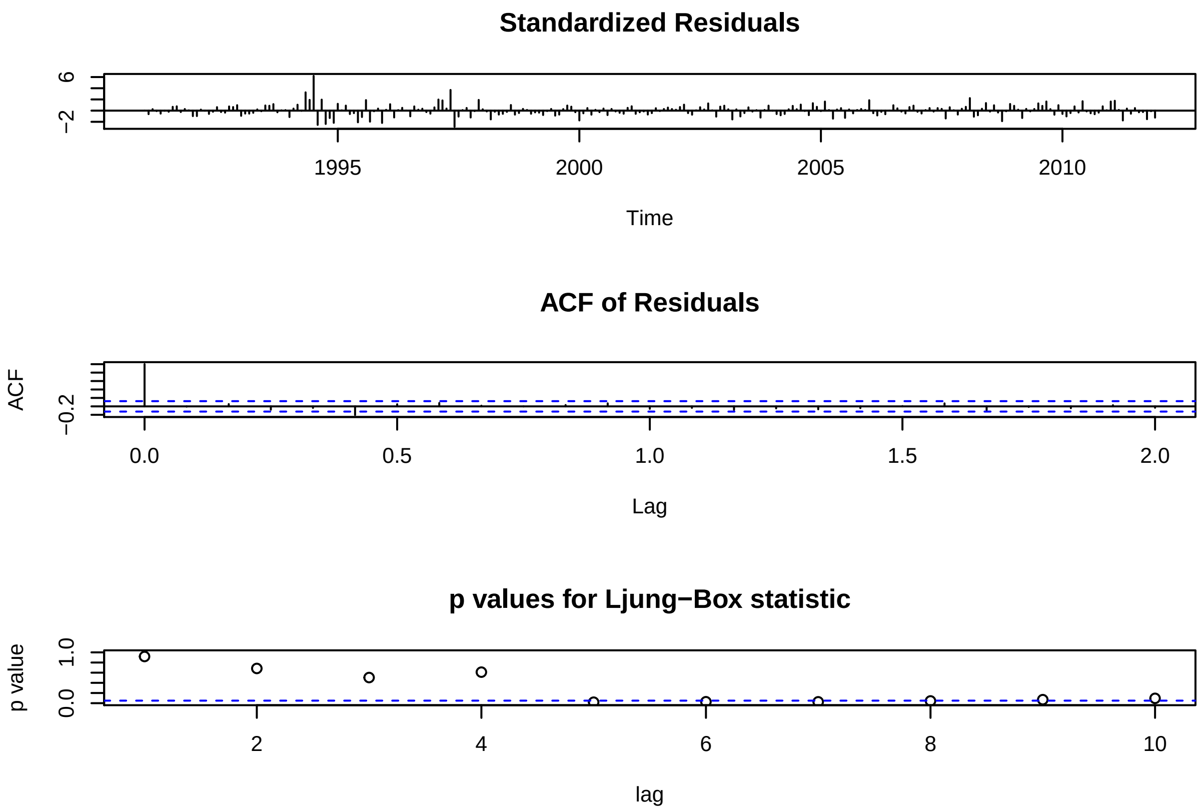
```r
tsdiag(m2)
```




```r
m3 <- arima(Bev, order = c(1,1,1), xreg = data.frame(Food))
m3
```

```
##
## Call:
## arima(x = Bev, order = c(1, 1, 1), xreg = data.frame(Food))
##
## Coefficients:
##          ar1      ma1    Food
##       0.3469  -0.1326  0.5479
## s.e.  0.1833   0.1880  0.1046
##
## sigma^2 estimated as 31.5:  log likelihood = -789.14,  aic = 1586.28
```

```r
tsdiag(m3)
```

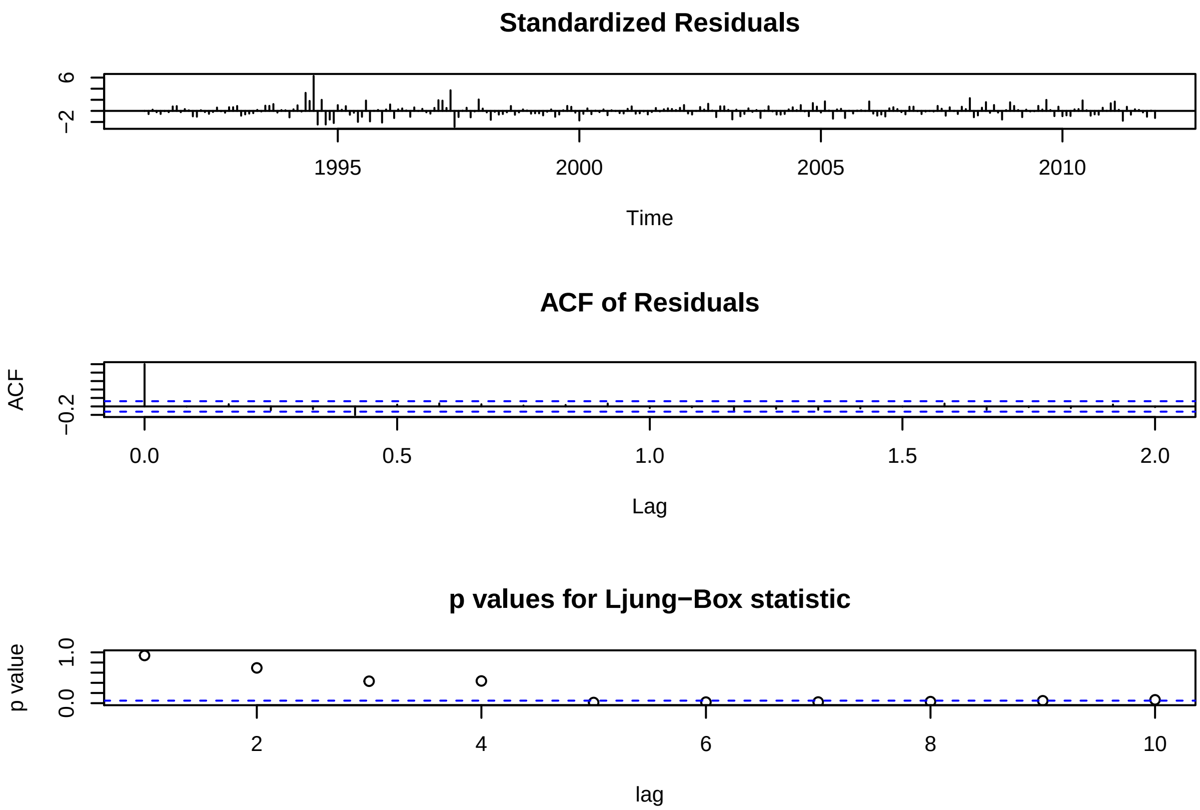


both seem to be important, so we add both to the model

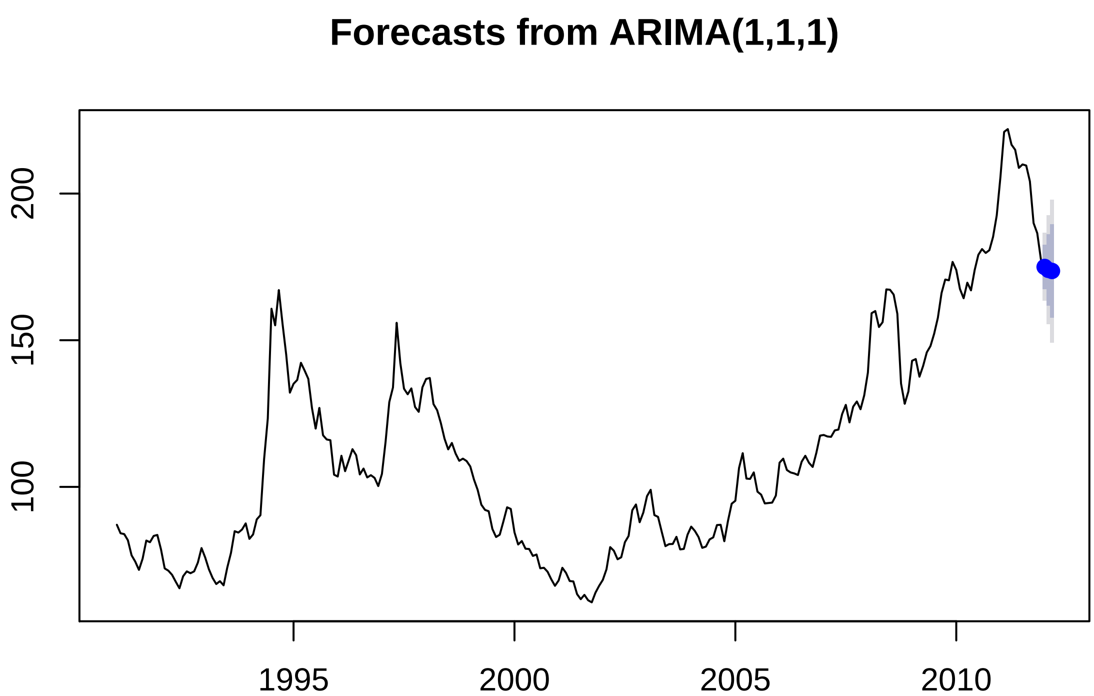
```
m4 <- arima(Bev, order = c(1,1,1), xreg = data.frame(Food, Indust))
m4
```

```
##
## Call:
## arima(x = Bev, order = c(1, 1, 1), xreg = data.frame(Food, Indust))
##
## Coefficients:
##          ar1      ma1    Food  Indust
##       0.3113  -0.1069  0.4041  0.2493
## s.e.  0.1890   0.1917  0.1166  0.0950
##
## sigma^2 estimated as 30.66:  log likelihood = -785.75,  aic = 1581.5
```

```
tsdiag(m4)
```



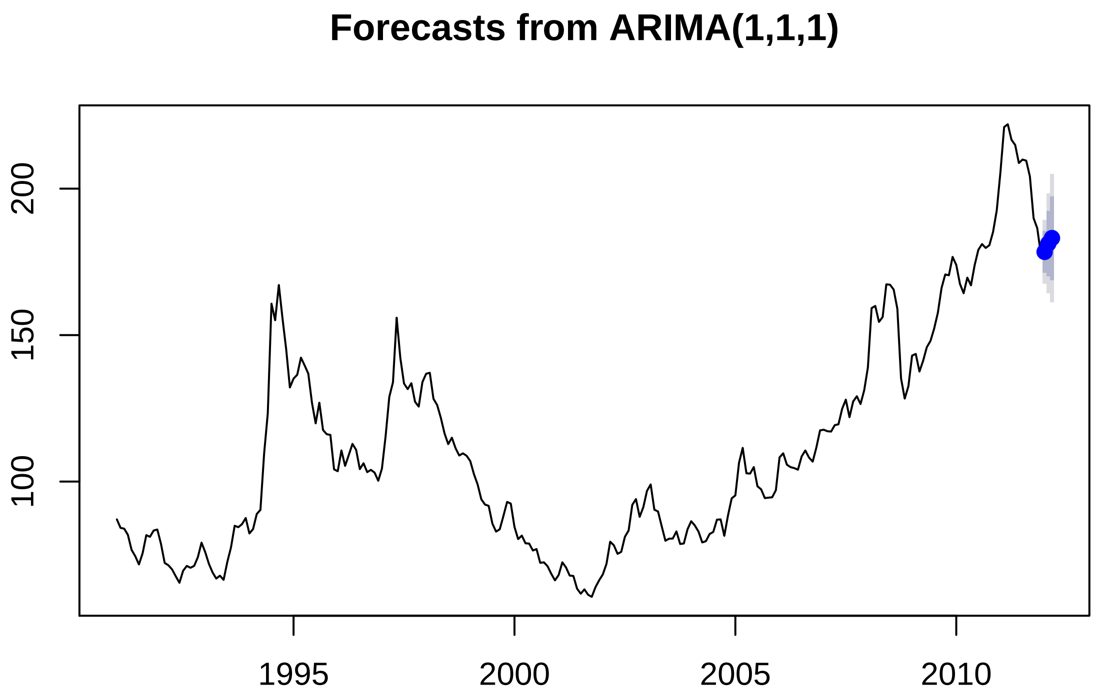

```r
# Prediction
par(mfrow = c(1,1))
f.arima <- forecast(m1, h = 3)
plot(f.arima)
```




```r
rmse.arima <- sqrt(mean((f.arima$mean - test$BevInd)^2))
rmse.arima
```

```
## [1] 2.766078
```

```r
par(mfrow = c(1,1))
f.arimax <- forecast(m4, h = 3, xreg = data.frame(Food = test$FoodInd, Indust = test$IndustInd))
plot(f.arimax)
```




```r
rmse.arimax <- sqrt(mean((f.arimax$mean - test$BevInd)^2))
rmse.arimax
```

```
## [1] 7.537393
```

```r
summary(VAR(y = data.frame(Bev, Food, Indust), p = 1))
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: Bev, Food, Indust
## Deterministic variables: const
## Sample size: 251
## Log Likelihood: -2155.572
## Roots of the characteristic polynomial:
## 0.9939 0.957 0.957
## Call:
## VAR(y = data.frame(Bev, Food, Indust), p = 1)
##
##
## Estimation results for equation Bev:
## ====================================
## Bev = Bev.l1 + Food.l1 + Indust.l1 + const
##
##           Estimate Std. Error t value Pr(>|t|)
## Bev.l1     0.94734    0.01980  47.836   <2e-16 ***
## Food.l1    0.04824    0.03836   1.258    0.210
## Indust.l1  0.01897    0.02475   0.767    0.444
```

```
## const     -0.91892    1.82569  -0.503    0.615
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 6.078 on 247 degrees of freedom
## Multiple R-Squared: 0.9745,  Adjusted R-squared: 0.9741
## F-statistic:  3141 on 3 and 247 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation Food:
## =====================================
## Food = Bev.l1 + Food.l1 + Indust.l1 + const
##
##            Estimate Std. Error t value Pr(>|t|)
## Bev.l1    -0.005942   0.011836  -0.502 0.616098
## Food.l1    0.942819   0.022928  41.121  < 2e-16 ***
## Indust.l1  0.049593   0.014792   3.353 0.000926 ***
## const      2.050464   1.091184   1.879 0.061406 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 3.632 on 247 degrees of freedom
## Multiple R-Squared: 0.9822,  Adjusted R-squared: 0.982
## F-statistic:  4547 on 3 and 247 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation Indust:
## =======================================
## Indust = Bev.l1 + Food.l1 + Indust.l1 + const
##
##            Estimate Std. Error t value Pr(>|t|)
## Bev.l1     0.003678   0.014559   0.253    0.801
## Food.l1   -0.033880   0.028202  -1.201    0.231
## Indust.l1  1.017186   0.018195  55.906   <2e-16 ***
## const      1.859972   1.342179   1.386    0.167
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 4.468 on 247 degrees of freedom
## Multiple R-Squared: 0.9861,  Adjusted R-squared: 0.986
## F-statistic:  5857 on 3 and 247 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##           Bev   Food Indust
## Bev    36.938  7.653  8.927
## Food    7.653 13.195  8.368
## Indust  8.927  8.368 19.963
##
## Correlation matrix of residuals:
```

```
##           Bev   Food Indust
## Bev    1.0000 0.3467 0.3287
## Food   0.3467 1.0000 0.5156
## Indust 0.3287 0.5156 1.0000
```

```r
VAR(y = data.frame(Bev, Food, Indust), p = 2)
```

```
##
## VAR Estimation Results:
## =======================
##
## Estimated coefficients for equation Bev:
## ========================================
## Call:
## Bev = Bev.l1 + Food.l1 + Indust.l1 + Bev.l2 + Food.l2 + Indust.l2 + const
##
##      Bev.l1     Food.l1   Indust.l1      Bev.l2     Food.l2   Indust.l2
##  1.17040231  0.26117475  0.04603840 -0.22398302 -0.21358026 -0.03302609
##       const
## -0.27900137
##
##
## Estimated coefficients for equation Food:
## =========================================
## Call:
## Food = Bev.l1 + Food.l1 + Indust.l1 + Bev.l2 + Food.l2 + Indust.l2 + const
##
##       Bev.l1      Food.l1    Indust.l1       Bev.l2      Food.l2
## -0.006060746  1.419993654 -0.011112792  0.011767386 -0.492526590
##    Indust.l2        const
##  0.055255773  2.866819839
##
##
## Estimated coefficients for equation Indust:
## ===========================================
## Call:
## Indust = Bev.l1 + Food.l1 + Indust.l1 + Bev.l2 + Food.l2 + Indust.l2 + const
##
##      Bev.l1     Food.l1   Indust.l1      Bev.l2     Food.l2   Indust.l2
##  0.02515920  0.29879281  1.22242706 -0.01322219 -0.32634842 -0.22380973
##       const
##  1.97848108
```

```r
VAR(y = data.frame(Bev, Food, Indust), p = 3)
```

```
##
## VAR Estimation Results:
## =======================
##
## Estimated coefficients for equation Bev:
## ========================================
## Call:
## Bev = Bev.l1 + Food.l1 + Indust.l1 + Bev.l2 + Food.l2 + Indust.l2 + Bev.l3 + Food.l3 + Indust.l3 + co
##
##      Bev.l1     Food.l1   Indust.l1      Bev.l2     Food.l2   Indust.l2
```

```
##  1.14955569  0.33277086  0.01762716 -0.12946705 -0.46090479  0.12560247
##      Bev.l3     Food.l3   Indust.l3       const
## -0.08017826  0.19289113 -0.13703118 -0.75227778
##
##
## Estimated coefficients for equation Food:
## =========================================
## Call:
## Food = Bev.l1 + Food.l1 + Indust.l1 + Bev.l2 + Food.l2 + Indust.l2 + Bev.l3 + Food.l3 + Indust.l3 + c
##
##        Bev.l1       Food.l1     Indust.l1        Bev.l2       Food.l2
## -0.0073950173  1.4001966351 -0.0161107552  0.0131010121 -0.4134540943
##     Indust.l2        Bev.l3       Food.l3     Indust.l3         const
##  0.0448252061  0.0009495142 -0.0621131772  0.0159099175  3.0232795766
##
##
## Estimated coefficients for equation Indust:
## ===========================================
## Call:
## Indust = Bev.l1 + Food.l1 + Indust.l1 + Bev.l2 + Food.l2 + Indust.l2 + Bev.l3 + Food.l3 + Indust.l3 +
##
##      Bev.l1     Food.l1   Indust.l1      Bev.l2     Food.l2   Indust.l2
##  0.02226847  0.26890392  1.17366282 -0.01782015 -0.19904757 -0.08244652
##      Bev.l3     Food.l3   Indust.l3       const
##  0.01013619 -0.09689744 -0.09758048  2.10369751
```