Price Analysis

AUTHOR
Dejene, G.

Introduction

- Analysis of price is an important area of research in economics.
- Price transmission is one type of price analysis

What is Price Transmission?

Price transmission is when a change in one price causes another price to change.

Types of Price Transmission?

Three types of price transmission:

a) **Vertical price transmission**:

- b/n two points (marketing stages) along the supply chain.
- For instance, between farm and retail level markets.

  e.g., price of wheat —> price of flour

- **Articles to read** about vertical price transmission: Asche et al. (2011), Asche et al. (2014), Vavra and Goodwin (2005),

b) **Spatial price transmission**:

- b/n two markets for same/homogeneous commodity.

e.g., price of wheat in US and price of wheat in Canada

- **Articles to read spatial price transmission**: Goodwin & Schroeder (1991), Ardeni (1989), Asche et al. (2004)

c) **Cross-commodity price transmission**:

- b/n two commodities

e.g., price of maize and price of rice

- **Note:** Spatial and cross price transmission often called Horizontal price transmission

Why is it useful to study price transmission?

- It helps diagnose poorly functioning markets
  - If two markets are close together, but show little price transmission, this might indicate problems with transportation network or monopolistic practices.
- It helps to assess the extent of competition in a market
  - Quick and complete price transmission b/n markets can be taken as an indication of high competition.
- Study of price transmission may help forecast prices based on trends in related prices
  - If changes in soybean prices transmit...

Price Transmission Methodology

- A wide variety of empirical techniques are used in the literature to study price transmission.
- Early studies use correlation to investigate the relationship between prices in different markets.
- If $p_{1,t}$ and $p_{2,t}$ are prices in two distinct markets at time t (expressed in log form) , the basic price transmission model:

$$p_{2,t} = \beta_1 + \beta_2 p_{1,t} + \mu_t$$

where μ_t is the error term.

- β_2 - is interpreted as elasticity of price transmission and defines the relationship b/n the prices
- $\beta_2 = 1$, perfect price transmission / Law of One Price (LOP)
- $\beta_2 = 0$, completely segmented markets.
- $0 < \beta_2 < 1$, integrated but not perfectly integrated
- **Concerns with the above model?**
  - static - price adjustment towards a long-run equilibrium take time.
  - Price series are often appear to be non-stationary (or to contain unit root).
  - The model above is often called the long-run relationship between the prices
- **Solution:** use co-integration models together with vector error correction model (VECM).

Cointegration test: The Engle-Granger Approach

- Retrieve the residuals from the long-run model above and use OLS to estimate the equation:

$$\Delta\mu_t = \rho\mu_{t-1} + \sum_{i=1}^{p} \gamma_i \Delta\mu_{t-i} + v_t$$

- Hypothesis Test: H_0 : no co-integration ($\rho = 0$) vs H_1 : co-integration

Vector Error Correction Model

- Taking $p_{2,t}$ as a dependent variable, the VECM can be given as:

$$\Delta p_{2,t} = \alpha_0 + \rho ECT_{t-1} + v_t$$

- If we include lags of our variables, the above model will be:

$$\Delta p_{2,t} = \alpha_0 + \rho ECT_{t-1} + \sum_{i=1}^{p} \alpha_i \Delta p_{2,t-i} + \sum_{i=1}^{p} \alpha_i \Delta p_{1,t-i} + v_t$$

Price Transmission along the supply chain of salmon

```
#get the data
sok3008 <- read.csv("C:/Users/dki007/Desktop/Sok-3008/mydata.csv", sep=";")

#rename
mydata <- sok3008
head(mydata)
```

```
  Year Month Export fresh smoked
1 2008     1   3.27  8.83  14.36
2 2008     2   3.21  8.95  13.39
3 2008     3   3.37  8.38  13.32
4 2008     4   3.25  8.95  13.32
5 2008     5   3.41  8.53  13.42
6 2008     6   3.31  8.66  13.30
```

```
#View(mydata)
```

```
# Descriptive statistics of the prices
summary(cbind(Export=mydata$Export,Fresh=mydata$fresh, Smoked=mydata$smoked))
```

```
     Export          Fresh            Smoked
 Min.   :2.910   Min.   : 8.19   Min.   :12.69
 1st Qu.:3.660   1st Qu.: 9.13   1st Qu.:14.40
 Median :4.790   Median :10.69   Median :15.61
 Mean   :4.834   Mean   :10.86   Mean   :16.05
 3rd Qu.:5.665   3rd Qu.:11.71   3rd Qu.:17.27
 Max.   :8.010   Max.   :15.37   Max.   :21.46
```

Transform the prices to log prices

```
Export <- mydata$Export
Fresh <- mydata$fresh
Smoked <- mydata$smoked
```

Transform the prices to log form:

```
lexport <- log(Export)
lfresh <- log(Fresh)
lsmoked <- log(Smoked)
```

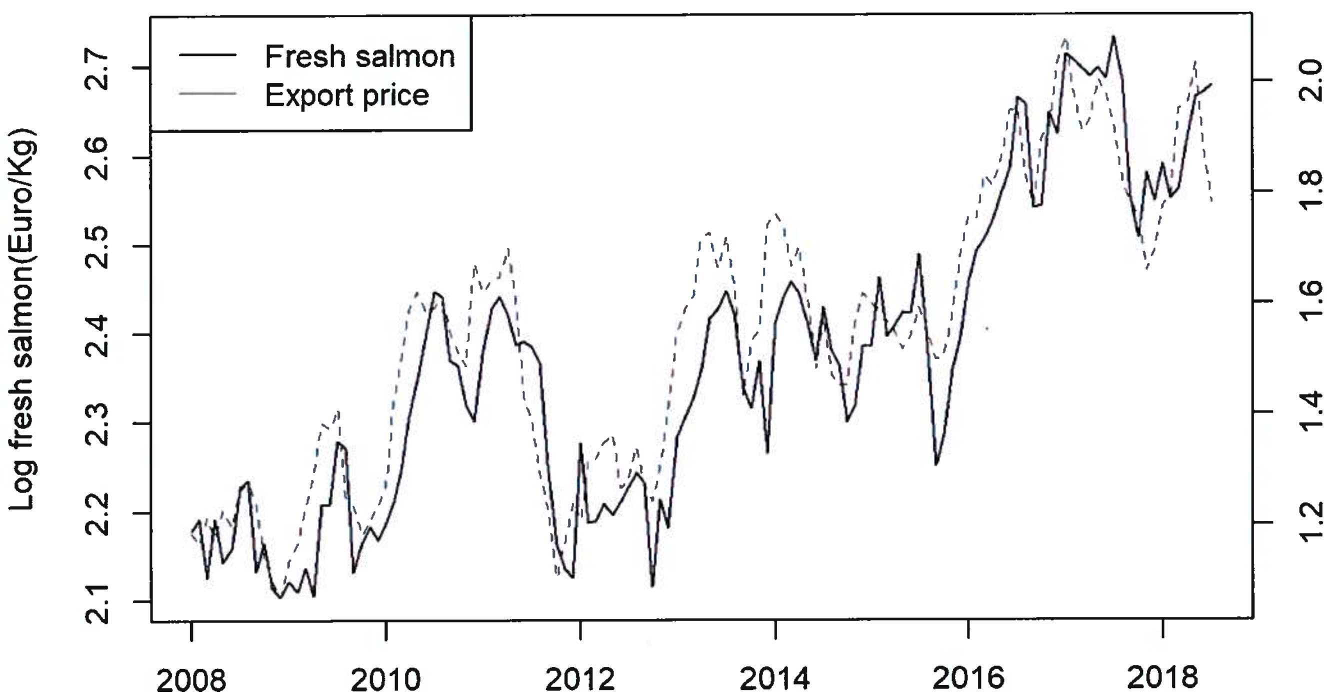
Convert the prices to time series data:

```
lexport=ts(lexport,start=c(2008, 1), end=c(2018, 7), frequency=12)
lfresh=ts(lfresh,start=c(2008, 1), end=c(2018, 7), frequency=12)
lsmoked=ts(lsmoked,start=c(2008, 1), end=c(2018, 7), frequency=12)


# plot the log prices

#plot of log of prices, Fresh salmon vs export price
plot(lfresh, type ="l", ylab = "Log fresh salmon(Euro/Kg)",
     main = "", xlab = "",
     col = "black")
par(new=TRUE)
plot(lexport, type = "l", xaxt = "n", yaxt = "n",
     ylab = "", xlab = "", col = "red", lty = 2)
axis(side = 4)
mtext("Log export Price(Euro/Kg)", side = 4, line = 3)

legend("topleft", c("Fresh salmon", "Export price"),
       col = c("black", "red"), lty = c(1, 1))
```



Stationarity Test : Augmented Dickey Test

Stationarity Test : prices in levels

```r
#' Test for stationarity,
#' H0: Unit-root, i.e. nonstationary.
suppressPackageStartupMessages(library(tseries))

# ADF on export prices
adf.test(lexport)
```

```
    Augmented Dickey-Fuller Test

data:  lexport
Dickey-Fuller = -3.0917, Lag order = 5, p-value = 0.1228
alternative hypothesis: stationary
```

```r
#ADF test on Fresh salmon price
adf.test(lfresh)
```

```
    Augmented Dickey-Fuller Test

data:  lfresh
Dickey-Fuller = -2.7786, Lag order = 5, p-value = 0.2529
alternative hypothesis: stationary
```

```r
#ADF test on Smoked salmon prices
adf.test(lsmoked)
```

```
    Augmented Dickey-Fuller Test

data:  lsmoked
Dickey-Fuller = -2.7335, Lag order = 5, p-value = 0.2717
alternative hypothesis: stationary
```

Stationarity Test : prices in first differences

```r
#First differences
adf.test(diff(lexport))
```

```
Warning in adf.test(diff(lexport)): p-value smaller than printed p-value

    Augmented Dickey-Fuller Test

data:  diff(lexport)
Dickey-Fuller = -5.7545, Lag order = 4, p-value = 0.01
alternative hypothesis: stationary
```

```r
#First differences
adf.test(diff(lfresh))
```

```
Warning in adf.test(diff(lfresh)): p-value smaller than printed p-value


    Augmented Dickey-Fuller Test

data:  diff(lfresh)
Dickey-Fuller = -5.818, Lag order = 4, p-value = 0.01
alternative hypothesis: stationary
```

```r
# First differences
adf.test(diff(lsmoked))
```

```
Warning in adf.test(diff(lsmoked)): p-value smaller than printed p-value


    Augmented Dickey-Fuller Test

data:  diff(lsmoked)
Dickey-Fuller = -6.6758, Lag order = 4, p-value = 0.01
alternative hypothesis: stationary
```

Cointegration test : Engle-Granger Approach

```r
suppressPackageStartupMessages(library(dynlm))
#Estimate long-run relationship
model_fresh=dynlm(lfresh~lexport)
summary(model_fresh)
```

```


Time series regression with "ts" data:
Start = 2008(1), End = 2018(7)

Call:
dynlm(formula = lfresh ~ lexport)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.228574 -0.044040 -0.004528  0.038078  0.159721 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.40265    0.03557   39.44   <2e-16 ***
lexport      0.62670    0.02274   27.56   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.06518 on 125 degrees of freedom
```

```
Multiple R-squared:  0.8587,	Adjusted R-squared:  0.8575 
F-statistic: 759.5 on 1 and 125 DF,  p-value: < 2.2e-16
```

```
#extract residuals from the long-run model, and convert to time series object
r_f=ts(resid(model_fresh),start=c(2008, 1), end=c(2018, 7), frequency=12)

# Cointegration test
# H0: the series are not cointegrated  vs H1: the series are cointegrated
coint_fresh=dynlm(diff(r_f)~ L(r_f,1)+L(diff(r_f),1))
summary(coint_fresh)
```

```
Time series regression with "ts" data:
Start = 2008(3), End = 2018(7)

Call:
dynlm(formula = diff(r_f) ~ L(r_f, 1) + L(diff(r_f), 1))

Residuals:
     Min       1Q   Median       3Q      Max 
-0.22050 -0.03145  0.00331  0.02648  0.14400 

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)    
(Intercept)      0.0001746  0.0050440   0.035    0.972    
L(r_f, 1)       -0.4391244  0.0920737  -4.769 5.17e-06 ***
L(diff(r_f), 1) -0.0795455  0.0922681  -0.862    0.390    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.05636 on 122 degrees of freedom
Multiple R-squared:  0.2337,	Adjusted R-squared:  0.2212 
F-statistic: 18.61 on 2 and 122 DF,  p-value: 8.845e-08
```

- **Compare** the τ (i.e., t-value) with the critical value for the cointegration test from the book (on page 583).
- Reject H0 if $\tau \leq \tau_c$ value .
- The value of the τ statistic in this case is : -4.769
- The 5% critical value with intercept included in the long-run relationship is -3.37
- **Conclusion** : Reject H_0: no cointegration since τ is less than τ_c

The result-that the export price and fresh salmon price are co-integrated- has major economic implications! It means that when export price changes, the retail prices also changes, and vice verse. To find the speed of price adjustment estimate vector error correction model.

Vector Error Correction Model

Estimate each models separately

```
Fresh_vecm <- dynlm(d(lfresh)~ L(r_f))
Export_vecm <- dynlm(d(lexport)~ L(r_f))

summary(Fresh_vecm)
```

```
Time series regression with "ts" data:
Start = 2008(2), End = 2018(7)

Call:
dynlm(formula = d(lfresh) ~ L(r_f))

Residuals:
     Min       1Q   Median       3Q      Max
-0.13404 -0.02668  0.00151  0.02525  0.11560

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.003244   0.003785   0.857    0.393
L(r_f)      -0.576383   0.059751  -9.646   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.04248 on 124 degrees of freedom
Multiple R-squared:  0.4287,    Adjusted R-squared:  0.4241
F-statistic: 93.05 on 1 and 124 DF,  p-value: < 2.2e-16
```

```
summary(Export_vecm)
```

```
Time series regression with "ts" data:
Start = 2008(2), End = 2018(7)

Call:
dynlm(formula = d(lexport) ~ L(r_f))

Residuals:
      Min        1Q    Median        3Q       Max
-0.205389 -0.042139  0.001594  0.053248  0.180143

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.004536   0.006658   0.681    0.497
L(r_f)      -0.159244   0.105105  -1.515    0.132

Residual standard error: 0.07472 on 124 degrees of freedom
Multiple R-squared:  0.01818,   Adjusted R-squared:  0.01026
F-statistic: 2.296 on 1 and 124 DF,  p-value: 0.1323
```

```
Fresh_vecm <- dynlm(d(lfresh)~ L(r_f)+L(d(lfresh),1)+ L(d(lexport),1))
Export_vecm <- dynlm(d(lexport)~ L(r_f)+L(d(lfresh),1)+ L(d(lexport),1))
```

```
summary(Fresh_vecm)


Time series regression with "ts" data:
Start = 2008(3), End = 2018(7)

Call:
dynlm(formula = d(lfresh) ~ L(r_f) + L(d(lfresh), 1) + L(d(lexport),
    1))

Residuals:
      Min        1Q    Median        3Q       Max
-0.137317 -0.022336  0.001583  0.029338  0.113083

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       0.002931   0.003836   0.764    0.446
L(r_f)           -0.523526   0.080982  -6.465 2.24e-09 ***
L(d(lfresh), 1)  -0.057903   0.073138  -0.792    0.430
L(d(lexport), 1)  0.070481   0.071128   0.991    0.324
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.04271 on 121 degrees of freedom
Multiple R-squared:  0.4364,    Adjusted R-squared:  0.4224
F-statistic: 31.23 on 3 and 121 DF,  p-value: 5.067e-15

summary(Export_vecm)


Time series regression with "ts" data:
Start = 2008(3), End = 2018(7)

Call:
dynlm(formula = d(lexport) ~ L(r_f) + L(d(lfresh), 1) + L(d(lexport),
    1))

Residuals:
      Min        1Q    Median        3Q       Max
-0.192681 -0.043570  0.000975  0.050052  0.185678

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       0.003905   0.006746   0.579    0.564
L(r_f)           -0.073414   0.142408  -0.516    0.607
L(d(lfresh), 1)   0.066530   0.128614   0.517    0.606
L(d(lexport), 1)  0.116402   0.125079   0.931    0.354

Residual standard error: 0.0751 on 121 degrees of freedom
Multiple R-squared:  0.03143,   Adjusted R-squared:  0.007421
F-statistic: 1.309 on 3 and 121 DF,  p-value: 0.2747
```

Estimate both equations together

```
suppressPackageStartupMessages(library(tsDyn))
vecm_fe=VECM(cbind(lfresh,lexport),lag = 1,r=1,include = "both")
summary(vecm_fe)
```

```
#############
###Model VECM
#############
Full sample size: 127   End sample size: 125
Number of variables: 2  Number of estimated slope parameters 10
AIC -1412.979   BIC -1381.868   SSR 0.913674
Cointegrating vector (estimated by 2OLS):
   lfresh   lexport
r1      1 -1.511549

                 ECT                Intercept          Trend
Equation lfresh  -0.0848(0.0299)**  0.0271(0.0133)*    -0.0003(0.0002).
Equation lexport 0.1357(0.0453)**   -0.0380(0.0201).   0.0006(0.0003)*
                 lfresh -1          lexport -1
Equation lfresh  -0.2115(0.0799)**  0.2997(0.0657)***
Equation lexport 0.0906(0.1209)     0.2851(0.0995)**
```

Estimate Vector Error Correction Model using all the three prices

```
vecm_fe=VECM(cbind(lfresh,lsmoked, lexport),lag = 1,r=1,include = "both")
summary(vecm_fe)
```

```
#############
###Model VECM
#############
Full sample size: 127   End sample size: 125
Number of variables: 3  Number of estimated slope parameters 18
AIC -2307.131   BIC -2250.565   SSR 0.9570211
Cointegrating vector (estimated by 2OLS):
   lfresh    lsmoked    lexport
r1      1 -0.6346367 -0.3971838

                 ECT                Intercept          Trend
Equation lfresh  -0.4677(0.0807)*** 0.0180(0.0085)*    -0.0002(0.0001).
Equation lsmoked 0.1743(0.0533)**   -0.0053(0.0056)    0.0001(7.9e-05).
Equation lexport -0.4237(0.1341)**  0.0249(0.0142).    -0.0003(0.0002)
                 lfresh -1          lsmoked -1         lexport -1
Equation lfresh  0.0004(0.0793)     0.0110(0.1337)     0.1263(0.0689).
Equation lsmoked -0.1686(0.0524)**  0.0225(0.0883)     0.0538(0.0455)
Equation lexport 0.2092(0.1317)     -0.0638(0.2221)    -0.0666(0.1146)
```