Information share and component share weights: R implementation

Nidhi Aggarwal
Indira Gandhi Institute of Development Research, Mumbai

Abstract

This paper explains the implementation of function `pdshare` in the package **ifrogs**. The function can be employed to estimate the two most widely used approaches to measure price discovery: information share and component share. The implementation is illustrated with an example to determine the share of spot and futures market for a stock in price discovery. The function can currently be used for a bivariate case only.

Keywords: Price discovery, information share, component share.

1. Introduction

Price discovery is a process by which new information is timely and efficiently incorporated into market prices of the assets. When the same asset (or an asset with similar attributes) is traded in multiple markets, the question that arises is: which market incorporates the new information first, or, which market leads the price discovery process? In order to answer this question, two techniques that have been developed and most widely used in the literature are: Information Share *(IS)*, proposed by Hasbrouck (1995) and Component Share *(CS)* proposed by Booth, So, and Tse (1999), Chu, Hsieh, and Tse (1999), Harris, McInish, and Wood (2002) which is based on the permanent and transitory decomposition of Gonzalo and Granger (1995). The two approaches are based on a common implicit efficient price that is contained in the observed price of a security and can be estimated using a vector error correction model (VECM) framework. Hasbrouck's IS focuses on the variance of the efficient price innovation, and measures what proportion of the efficient price variance can be attributed to the innovations from different markets. The CS approach, on the other hand, focuses on the composition of the efficient price innovation and measures a market's contribution to price discovery as its contribution to the efficient price innovation.

At present, there is no standard package in R that implements the two approaches. The function `pdshare()` in the package **ifrogs** attempts to provide a tool for the same. It has been used for the estimations of the IS and CS weights in Aggarwal and Thomas (2011).[1] Currently, the function can be used for a bivariate case only. However, the future version of the package is aimed at extending the function to more than two markets.

The article proceeds as follows: Section 2 briefly describes the derivation of the two measures. Section 3 explains the function and the implementation method. Section 4 illustrates the

[1] An example dataset used to illustrate the function replicates the results of Table 10 in the paper.

function with an example. Section 5 and 6 show the accuracy and computational efficiency of the function. Section 7 summarises.

2. Model specification

Let $\mathbf{p}_{i,t} = (p_{1,t},\ p_{2,t})'$ denote a vector of (log) prices of a security traded on two distinct markets, 1 and 2, such that

$$\mathbf{p}_{i,t} = (p_{1,t}, p_{2,t}) \sim I(1)$$

Since the prices represent the same security, the two prices are linked by the force of arbitrage and cannot deviate from each other in the long run. The two price series are thus cointegrated and the linear relationship is expressed as

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

where $\mu_t \sim I(0)$ and β is the cointegrating vector, $\beta = (1, -\beta)'$. The VECM representation for 'k' lags can be given as

$$\Delta\mathbf{p}_t = \alpha\beta'\mathbf{p}_{t-1} + \Gamma_1\Delta\mathbf{p}_{t-1} + ...\Gamma_k\Delta\mathbf{p}_{t-k} + \epsilon_t \tag{1}$$

where α represents the coefficient associated with the error correction term. ϵ is a 2×1 vector of the residuals with $\epsilon_t \sim N(0, \Omega)$. Since the price changes are assumed to be covariance stationary, the vector moving average (VMA) or the Wold representation is given as:

$$\triangle\mathbf{p}_t = \Psi(L)\epsilon_t$$

where $\Psi(L) = \Sigma_{s=0}^{\infty}\Psi_s L^s$

The Beveridge-Nelson decomposition (Beveridge and Nelson 1981) can be used to derive the common trends representation in the levels prices:

$$\mathbf{p}_t = \mathbf{p}_0 + \Psi(1)\Sigma_{s=0}^{t}\epsilon_s + \Psi^*(L)\epsilon_t$$

where, $\Psi(1) = \Sigma_{k=0}^{\infty}\Psi_k$. In the above equation, the matrix $\Psi(1)$ contains the cumulative impact of innovation ϵ_t on all future price movements and thus measures the long run impact of ϵ_t on prices. Hasbrouck (1995) shows that since both the price series represent an identical security, the long run impact of ϵ_t on each of the price series should be the same. Thus, in principle, the rows of $\Psi(1)$ are identical.

2.1. Information share

Denote $\psi = (\psi_1, \psi_2)$ as the common row vector of $\Psi(1)$. $\psi\epsilon_t$ is the incremental change in price that is permanently impounded into the security prices and is presumably due to new information (captured in ϵ_t). Hasbrouck (1995) proposes the use of the structure of the

variance of this component to derive the measure of price discovery. The variance of $\psi\epsilon_t$ is written as:

$$\text{var}(\psi\epsilon_t) = \psi\Omega\psi'$$

If Ω is diagonal, then $\psi\Omega\psi'$ will consist of 'n' terms, each of which would represent contribution to the efficient price innovation from each market. The proportion of the variance of the efficient price innovation that can be attributed from an innovation from market 'j', is called 'j'*th* market's information share. It is defined as

$$IS_j = \frac{\psi_j^2\Omega_{jj}}{\psi\Omega\psi'}$$

where ψ_j is the j^{th} element of ψ.

However, if Ω is not diagonal, that is, if the price innovations are correlated, the proposed measure has the problem of attributing the covariance terms to each market. To overcome this problem, Hasbrouck (1995) suggested the use of triangularization/Cholesky decomposition of Ω and measure IS using the orthogonalized innovations. This can be accomplished in the following way:

Let 'F' be a lower triangular matrix such that $FF' = \Omega$. The IS for the j^{th} market is then defined as

$$IS_j = \frac{([\psi'F]_i)^2}{\Psi\Omega\Psi'}$$

The resulting IS will depend on the ordering of price variables. The upper bound of IS of a particular market can be obtained by placing that market's price first. Similarly, a lower bound can be obtained by placing that market's price the last. For 'n' markets, by doing all the permutations, one can obtain the obtain the upper and lower bound of IS for each market.

2.2. Component share

Under this approach, $\mathbf{p}_t$ takes the form:

$$\mathbf{p}_t = A_1 f_t + A_2 z_t$$

where $f_{=}\gamma'\mathbf{p}_t \sim I(1)$ is the permanent component and $z_t \sim I(0)$ is the transitory component. While A_1 and A_2 are the loading matrices, γ' is the matrix of common factor weights. Gonzalo-Granger defined $\gamma = (\alpha'_\perp\beta_\perp)^{-1}\alpha'_\perp$ such that $\alpha'_\perp\alpha = 0$ and $\beta'_\perp\beta = 0$. Note that α and β correspond to the VECM representation mentioned before. Booth *et al.* (1999), Chu *et al.* (1999), Harris *et al.* (2002), Baillie, Booth, Tse, and Zabotina (2002) suggested measuring price discovery in market 'j' using the component share as:

$$CS_j = \frac{\alpha_{\perp,j}}{\alpha_{\perp,j} + \beta_{\perp,j}}, j = 1, 2$$

3. R implementation

The two approaches described in the previous section can be implemented using the function `pdshare`. The function consumes a matrix of (log) prices and returns a list of IS and CS measures, the covariance matrix of residuals and the number of lags used for estimation. The function is described below:

```
> library(ifrogs)
> str(pdshare)

function (x, override.lags = NULL, lag.max = 10)
```

The data matrix of prices for which the measure is to be estimated is provided in the argument `x`. A user can specify the lag order to be used in VECM estimation by using the argument `override.lags`. Alternatively, the user can also use the Akaike information criterion (AIC) in order to automatically determine the number of lags. This can be achieved by specifying the highest lag order that one would want to use in the argument `lag.max`. Once an integer in `lag.max` is specified, the function `pdshare` uses the `VARselect` function in package **vars** to select the number of lags.

The function `pdshare` first estimates a VEC model for the underlying data in `x` using the functionality in package **urca**. The orthogonalized coefficient matrices are then derived using function `Psi` in package **vars**. Subsequently, the $\Psi(1)$ matrix is computed as $\beta_\perp(\alpha_\perp{}'\Gamma\beta_\perp)^{-1}\alpha_\perp$ where $\Gamma = (I - \Gamma_1 - \ldots - \Gamma_{k-1})$. Once these estimations are done, IS and CS weights are computed as specified in Section 2.

As we discussed in Section 2, the IS estimates depend on the ordering of the price variables. IS is first estimated as per the supplied ordering. Once the estimation on supplied ordering is done, the results are stored and then the ordering of the price variables is automatically reversed. That is, Price series 1 will be column 2 and Price series 2 will be column 1 in the data matrix. IS estimation is then done for the reversed ordering.

The function `pdshare` returns a list of five elements. These are: `is.original.ordering`, `is.reversed.ordering`, `component.share`, `var.covar.matrix` and `lags.used`. `is.original.ordering` returns the IS estimates for the supplied ordering, `is.reversed.ordering` returns the IS estimates for the reversed ordering. Note that the first and second element of the vector `is.original.ordering` specify the IS estimate of Market 1 and 2 respectively. In contrast, the first and the second element of the vector `is.reversed.ordering` specify the IS estimate of Market 2 and 1 respectively. `var.covar.matrix` returns the variance-covariance matrix of the VECM residuals. One can obtain the number of lags used in the VECM estimation using `lags.used`.

4. Example

This section illustrates the function with a data set on the spot and the futures market for a stock `RELIANCE`, traded on the National Stock Exchange, India. In the first step, the package **ifrogs** and dataset `idfc` are loaded.

```
> library(ifrogs)
```

```
> data(is_reliance)
> head(is_reliance)

             datetime   spot futures
1 2009-05-06 06:25:07 1873.9 1877.85
2 2009-05-06 06:25:08 1873.9 1879.00
3 2009-05-06 06:25:09 1865.0 1879.00
4 2009-05-06 06:25:10 1865.0 1877.50
5 2009-05-06 06:25:11 1867.5 1876.50
6 2009-05-06 06:25:12 1867.5 1876.50
```

Since the prices are of the same security traded on two different markets – spot and the futures, the two price series are cointegrated. We now proceed to estimate the IS and CS weights for the two markets using the function `pdshare`.

```
> compute.time <- system.time(ans <- pdshare(log(is_reliance[,-1]), lag.max=120))
> summary(is_reliance)

    datetime                       spot          futures
 Min.   :2009-05-06 06:25:07   Min.   :1862   Min.   :1868
 1st Qu.:2009-05-06 07:48:50   1st Qu.:1883   1st Qu.:1885
 Median :2009-05-06 09:12:33   Median :1896   Median :1898
 Mean   :2009-05-06 09:12:33   Mean   :1900   Mean   :1903
 3rd Qu.:2009-05-06 10:36:16   3rd Qu.:1914   3rd Qu.:1918
 Max.   :2009-05-06 12:00:00   Max.   :1938   Max.   :1944
```

We use the automatic lag selection based on the AIC criterion, and specify an upper bound for the maximum number of lags to be used for VECM estimation as 5. `ans.pds` shows the list of objects returned from the function.

```
> ans$is.original.ordering

                IS
spot    0.07401349
futures 0.92598651

> ans$is.reversed.ordering

               IS
futures 0.9842687
spot    0.0157313

> ans$component.share

               CS
spot    0.1092093
futures 0.8907907
```

```
> ans$var.covar.matrix

                spot      futures
spot    2.973507e-08 4.173273e-09
futures 4.173273e-09 2.630731e-08

> ans$lags.used

[1] 52
```

ans$is.original.ordering indicates that the spot market (Market 1) has an IS of 7% while that of the futures market (Market 2) is 93%. On reversing the ordering, the IS of spot and the futures market is estimated as 2% and 98% respectively. The average IS of the futures market is thus, 96%. The estimates indicate that the futures market leads the price discovery process. The component share weights also show that the futures market has a share of 89% while the spot market has a share of 11% in price discovery. These values replicate the results in Table 10 from the paper Aggarwal and Thomas (2011). The number of lags used in the VECM estimation is fifty-two

Alternatively, the user can fix the number of lags using override.lags. This can be done as follows:

```
> compute.time.1 <- system.time(ans1 <-
+                                pdshare(log(is_reliance[,-1]), override.lags=60))
> ans1

$is.original.ordering
                IS
spot    0.06864645
futures 0.93135355

$is.reversed.ordering
              IS
futures 0.986837
spot    0.013163

$component.share
               CS
spot    0.1006936
futures 0.8993064

$var.covar.matrix
                spot      futures
spot    2.964055e-08 4.174971e-09
futures 4.174971e-09 2.629263e-08

$lags.used
[1] 60
```

`ans1` shows the results of the estimation by imposing the number of lags to be used in the model as sixty.

5. Benchmarking accuracy

The accuracy of the code has been tested using the analytical examples presented in Baillie *et al.* (2002). The authors use an error correction model and derive the *IS* and *CS* measures for three different examples of the error correlation behavior. The paper uses the following model:

For two cointegrated series x_1 and x_2

$$\Delta x_1 = -\alpha_1(x_{1,t-1} - x_{2,t-1}) + \epsilon_{1t}$$

$$\Delta x_2 = \alpha_2(x_{1,t-1} - x_{2,t-1}) + \epsilon_{2t}$$

$$\text{cov}(e_{1t}, e_{2t}) = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

where α_1 and α_2 are positive constants and $e_{it} \sim N(0, 1)$.

For different values of α_1, α_2 and ρ, the authors report the IS and CS values in Table 1 of the paper.

We simulate the above model for different values of α_1, α_2 and ρ (as per the paper) and report the true and estimated values of IS and CS measures in Table 1.[2] As in Baillie *et al.* (2002), we report the values of these measures for x_1 only.

As we see from the table, there is negligible difference between the estimated values returned using the function `pdshare` and the true values.

6. Computational efficiency

This section discusses the performance of the function in terms of computational time taken to estimate the two measures. All computations are done on a 64bit Linux machine with R-2.15.3. `compute.time` and `compute.time.1` in Section 4 record the time taken to compute IS and CS for a day using one second data for RELIANCE. The results are as follows:

```
> compute.time

   user  system elapsed
 87.725   5.728  93.504

> compute.time.1  # on using override.lags=4

   user  system elapsed
  4.813   0.260   5.072
```

[2] 50 simulations for each model type were conducted. The table reports the mean value of estimated IS and CS.

Table 1 True and estimated IS and CS values

The table shows the true and estimated values of IS and CS for series x_1 obtained for the model used in the anaytical example in Baillie *et al.* (2002). UB indicates the upper bound while LB indicates the lower bound of the IS measure.

	True values			Estimated values		
Example	IS-UB	IS-LB	CS	$\widehat{\text{IS-UB}}$	$\widehat{\text{IS-LB}}$	$\widehat{\text{CS}}$
A: $\alpha_1 = \alpha_2 = 0.05$						
$\rho = 0$	0.50	0.50	0.50	0.50	0.50	0.50
$\rho = 0.1$	0.55	0.45	0.50	0.55	0.45	0.50
$\rho = 0.5$	0.75	0.25	0.50	0.75	0.25	0.50
$\rho = 0.9$	0.95	0.05	0.50	0.95	0.05	0.51
B: $\alpha_1 = 0.0, \alpha_2 = 0.05$						
$\rho = 0$	1.00	1.00	1.00	1.00	1.00	0.98
$\rho = 0.1$	1.00	0.99	1.00	1.00	0.99	0.98
$\rho = 0.5$	1.00	0.75	1.00	1.00	0.75	0.98
$\rho = 0.9$	1.00	0.19	1.00	1.00	0.19	0.96
C: $\alpha_1 = 0.025, \alpha_2 = 0.05$						
$\rho = 0$	0.80	0.80	0.67	0.80	0.80	0.67
$\rho = 0.1$	0.82	0.73	0.67	0.82	0.74	0.67
$\rho = 0.5$	0.89	0.43	0.67	0.89	0.43	0.67
$\rho = 0.9$	0.98	0.09	0.67	0.98	0.09	0.68

A large chunk of the computational time is taken in the automated selection of the number of lags using `VARselect`. The function `pdshare` uses a modified `VARselect` (called `MVARselect`). `MVARselect` replaces `lm` to `lm.fit` in the function `VARselect`.[3] This improves the performance of the code by 2x. In addition, specification of the number of lags in `override.lags` can substantially reduce the amount of time taken by the function in generating the output.

7. Summary

This document explains the functionality of `pdshare` in R to estimate the two most common techniques to measure price discovery: information share and component share. The function makes extensive use of the functions in **urca** and **vars** packages.

Currently the function is implemented for a bivariate case only. However, the future version of the function will be extended to multiple markets case.

8. Acknowledgements

I would like to thank Ajay Shah and Susan Thomas for helpful comments and suggestions.

[3] We replace `resids <- resid(lm(yendog ~1 + ys.lagged))` to `resids <- lm.fit(x=ys.lagged, y=yendog)$residuals` in the original function `VARselect`.

References

Aggarwal N, Thomas S (2011). "When do stock futures dominate price discovery?" *Technical report*, IGIDR WP-2011-016. URL http://www.igidr.ac.in/pdf/publication/WP-2011-016.pdf.

Baillie R, Booth G, Tse Y, Zabotina T (2002). "Price discovery and common factor models." *Journal of Financial Markets*, **5**(3), 309–321.

Beveridge S, Nelson C (1981). "A new approach to the decomposition of economic timeseries into permanent and transitory components with particular attention to the measurement of the 'businesscycle'." *Journal of Monetary Economics*, **7**, 151–174.

Booth G, So R, Tse Y (1999). "Price discovery in the German equity index derivatives." *Journal of Futures Markets*, **19(6)**, 619–643.

Chu QC, Hsieh WG, Tse Y (1999). "Price discovery on the S&P 500 index markets: An analysis of spot index, index futures and SPDRs." *International Review of Financial Analysis*, **8(1)**, 21–34.

Gonzalo J, Granger C (1995). "Estimation of common long-memory components in cointegrated systems." *Journal of Business and Economic Statistics*, **13(1)**, 27–35.

Harris F, McInish TH, Wood R (2002). "Security price adjustments across exchanges: An investigation of common factor components for Dow stocks." *Journal of Financial Markets*, **5(3)**, 277–308.

Hasbrouck J (1995). "One security, many markets: Determining the contributions to price discovery." *Journal of Finance*, **50(4)**, 1175–1199.

Yan B, Zivot E (2010). "A structural analysis of price discovery measures." *Journal of Financial Markets*, **13(1)**, 1–19.

Affiliation:

Nidhi Aggarwal
Indira Gandhi Institute of Development Research
Goregaon East, Mumbai
400 065, India
E-mail: nidhi@igidr.ac.in