Package 'ioanalysis'

February 1, 2015

Title Input-Output Analysis

Version 0.1

Author Ignacio Sarmiento-Barbieri

Maintainer Ignacio Sarmiento-Barbieri <srmntbr2@illinois.edu>

Description Input and Output Analysis.

Depends R (>= 3.1.1), ggplot2, xlsx

License GNU GENERAL PUBLIC LICENSE

LazyData true

R topics documented:

`agg.matrix` *Aggregation Matrix*

Description

Aggregates Inpuy-Output Matrices. Construct an agregation matrix of kxn dimension.

Usage

```
agg.matrix<-function(mat,d.mip)
```

Arguments

`mat`	Matrix. How sectors should be agregated. First column should be the new sector in order, the remaining columns the sectors that are foing to be agregated
`dimcol`	Numeric. Column dimension

Author(s)

Ignacio Sarmiento-Barbieri

References

Nazara, Suahasil & Guo, Dong & Hewings, Geoffrey J.D., & Dridi, Chokri, 2003. "PyIO. Input-Output Analysis with Python". REAL Discussion Paper 03-T-23. University of Illinois at Urbana-Champaign. (http://www.real.illinois.edu/d-paper/03/03-T-23.pdf)

Examples

```
temp<-matrix(sample(1:40), 14)
temp[,1]<-seq(1:14)
S<-agg.matrix(temp,40)
```

`key.sector` *Impact Analysis: Backward and Forward linkages*

Description

Computation of Backward and Forward linkages. It aims to identify thos secotrs whose economic activity exerts a greater than average influence on the whole economy. Key sectors are identified by calculating backard and forward linkages. Let

$$B = (I - A)^{-1} = [b_{ij}]$$

be the Leontief inverse matrix and let B_j and B_i be the column and row multipliers of this Leontief inverse. The sector j's backward linkage (BL_j) and forward linkage (FL_i) are defined as:

$$BL_j = \frac{\frac{1}{n}\sum_{i=1}^{n} b_{ij}}{\frac{1}{n^2}\sum_{j=1}^{n}\sum_{i=1}^{n} b_{ij}}$$

$$FL_i = \frac{\frac{1}{n}\sum_{j=1}^{n} b_{ij}}{\frac{1}{n^2}\sum_{j=1}^{n}\sum_{i=1}^{n} b_{ij}}$$

The usual interpretation is to propose that, if

$$BL_j > 1$$

, a unit change in final demand in sector j will generate an above-average increase in activity in the economy. Similarly, for

$$FL_i > 1$$

, it is asserted that a unit change in all sector's final demand would create an above average increase in sector i. Thus, a key sector is identified as one having both indices grater than one.

Usage

```
key.sector(mip, X, epsilon=0.1, key=TRUE, cutoff=1, write.xlsx=TRUE, name="Key_sector.xlsx")
```

Arguments

mip	Input-output matrix
X	Total input or output
epsilon	Replaces zeros in X
key	Logical. If TRUE identifies key sectors
cutoff	Cutoff above which are the key sectors
write.xlsx	Logical. If TRUE writes an excel file
name	String. Name of the excel file

Details

The function takes the sector names from the column names of the Input-output matrix

Value

Returns a vector with the calculated Backward and Forward linkages for each sector

Author(s)

Ignacio Sarmiento-Barbieri

References

Nazara, Suahasil & Guo, Dong & Hewings, Geoffrey J.D., & Dridi, Chokri, 2003. "PyIO. Input-Output Analysis with Python". REAL Discussion Paper 03-T-23. University of Illinois at Urbana-Champaign. (http://www.real.illinois.edu/d-paper/03/03-T-23.pdf)

See Also

See Also leontief.inv gosh.inv

Examples

```
#Uses the 40x40 matrix included in the package

mip<-mat_40x40[1:40,2:41] #Input-output coeffcients
X<-mat_40x40$DT.a.PB[1:40]  #Total output vector

key<-key.sector(mip,X, key=FALSE, write.xlsx=FALSE)
```

`leontief.inv` *Leontief Inverse*

Description

Computes the Leontief Inverse and the Backward Linkage

Usage

```
leontief.inv(mip,X.j,write.xlsx=TRUE,name="Leontief_Inv.xlsx")
```

Arguments

`mip`	Matrix. Input output matrix
`X.j`	Vector. Input in each column
`write.xlsx`	Logical. if TRUE writes an excel file
`name`	String. name of the excel file

Details

The Leontief inverse is derived from the input-output table A=[a_ij] where a_ij=z_ij/X_j

where z_ij is the input from i required in the production of j. X_j is the corresponding input in each coulumn. The leontief inverse is then computed as (I-A)^(-1)

Falta describir Backward Linkage

Author(s)

Ignacio Sarmiento-Barbieri

References

Nazara, Guo, Hewing and Dridi (2003). *PyIO. Input-Output Analysis with Python*. http://www.real.illinois.edu/pyio/

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (x)
{
  }
```

upstream *upstreamness*

Description

Measures upstreamness as of equation 9 in Antras et al. (2012)

Usage

```
upstream(linv,y,x,m, write.xlsx=TRUE, name="Upstream.xlsx")
```

Arguments

linv	Matrix. Input output matrix
y	Output Vector
x	Exports Vector
m	Imports Vector
write.xlsx	Logical. if TRUE writes an excel file
name	String. name of the excel file

Author(s)

Ignacio Sarmiento-Barbieri

References

Pol Antràs & Davin Chor & Thibault Fally & Russell Hillberry, 2012. "Measuring the Upstreamness of Production and Trade Flows," NBER Working Papers 17819, National Bureau of Economic Research, Inc.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (x)
{
  }
```

vs.io *VS share of total exports*

Description

Vertical Specialization (VS) share of total exports. The formula for VS as a share fo total exports for country k is

$$VS\ share\ of\ total\ exports = \frac{VS}{X_k} = \frac{A * L * X}{X_k}$$

where A is the nxn imported coefficient matrix,L is the Leontief inverse, X is an nx1 vector of exports, n is the number of secotrs, adn Xk is the sum of exports across the n sectors.

Usage

```
vs.io(imp,exp,leon=1,namesector, write.xlsx=TRUE, name="Level_Verticalization.xlsx")
```

Arguments

imp	Is the nxn imported coefficient matrix
exp	Numeric Vector. Exports
leon	Leontief Inverse as output of funciton leontief.inv()
namesector	Character. Name of Sector i
write.xlsx	Logical. if TRUE writes an excel file
name	String. name of the excel file

Value

Returns a data frame with the A*L*X product, the total exports and the level of verticalization for each sector. Total.exp. returns a scalar, equals to the sum of exports across sectors

Author(s)

Ignacio Sarmiento-Barbieri

References

Hummels, David & Ishii, Jun & Yi, Kei-Mu, 2001. "The nature and growth of vertical specialization in world trade," Journal of International Economics, Elsevier, vol. 54(1), pages 75-96, June.

See Also

See Also leontief.inv

Examples

```
#Uses the 40x40 matrix included in the package

exp<-mat_40x40$X[1:40] #Exports Vector
class(exp) #the class is numeric
imp<-mat_imp_40x40[1:40,2:41]/mat_40x40$DT.a.PB[1:40] #Imports Coef Matrix
class(imp) #the class is data.frame
namesector<-colnames(mat_40x40[2:41])
class(namesector) #character

hum<-vs.io(imp,exp,namesector, write.xlsx=FALSE)

#Using the Leontief Inverse
leon<-leontief.inv(mat_40x40[1:40,2:41], as.matrix(mat_40x40$DT.a.PB[1:40]))[,1:40]
hum2<-vs.io(imp,exp,namesector,leon=leon, write.xlsx=FALSE)
```

---

vs.ki *Vertical Specialization*

Description

VS is the imported input content of exports for country k in sector i. The Vertical specialization chain is VS_ki=(imported intermediats/gross output)*exports. The first term is the share of imported inputs into gross production. Multiplying this ratio by the amount that iseported provides the monetary value for the imported input content of exports.

Usage

```
vs.ki(imp,exp,out,namesector, write.xlsx=TRUE, name="Level_Verticalization.xlsx")
```

Arguments

imp	Vector. Imported intermediates of sector i
exp	Vector. Exports
out	Vector. Gross Output
namesector	String. Name of Sector i
write.xlsx	Logical. if TRUE writes an excel file
name	String. name of the excel file

Details

The dimension of the four arguments must coincide.

Value

Returns a data frame

Author(s)

Ignacio Sarmiento-Barbieri

References

Hummels, David & Ishii, Jun & Yi, Kei-Mu, 2001. "The nature and growth of vertical specialization in world trade," Journal of International Economics, Elsevier, vol. 54(1), pages 75-96, June.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (x)
{
  }
```

Index