Package ‘decompr’

October 13, 2022

Version 6.4.0

Title Global Value Chain Decomposition

Description Three global value chain (GVC) decompositions are implemented. The Leontief decomposition derives the value added origin of exports by country and industry as in Hummels, Ishii and Yi (2001). The Koopman, Wang and Wei (2014) decomposition splits country-level exports into 9 value added components, and the Wang, Wei and Zhu (2013) decomposition splits bilateral exports into 16 value added components. Various GVC indicators based on these decompositions are computed in the complimentary 'gvc' package.

--- References: ---

Maintainer Bastiaan Quast <bquast@gmail.com>

Depends R (&gt;= 3.5.0)
Imports matrixStats
License GPL-3


BugReports https://github.com/bquast/decompr/issues

Suggests gvc, testthat, knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.2.0

Encoding UTF-8

NeedsCompilation yes
Author  Bastiaan Quast [aut, cre] (<https://orcid.org/0000-0002-2951-3577>),  
       Fei Wang [aut],  
       Victor Stolzenburg [aut],  
       Oliver Reiter [ctb],  
       Sebastian Krantz [ctb]  
Repository  CRAN  
Date/Publication  2022-06-19 09:50:02 UTC  

R topics documented:  
deompr-package  ................................................................. 2  
decomp  ................................................................. 3  
kww  ................................................................. 5  
leather  ................................................................. 7  
leontief  ................................................................. 7  
load_tables  ................................................................. 9  
load_tables_vectors  ........................................................... 10  
wwz  ................................................................. 12  
wwz2kww  ................................................................. 15  

Index  18  

---  
deompr-package  

Global Value Chain Decomposition  

Description  
Three global value chain (GVC) decompositions are implemented. The Leontief decomposition derives the value added origin of exports by country and industry as in Hummels, Ishii and Yi (2001). The Koopman, Wang and Wei (2014) decomposition splits country-level exports into 9 value added components, and the Wang, Wei and Zhu (2013) decomposition splits bilateral exports into 16 value added components. Various GVC indicators based on these decompositions are computed in the complimentary ‘gvc’ package.  

Contents  
Interface function for quick analysis  
decomp()  
Function to load ICIO table and create a ‘deompr’ object  
load_tables_vectors()  
Functions to perform GVC decompositions on a ‘deompr’ object  
leontief()  
kww()  
wwz()  
Function to obtain KWW decomposition from WWZ decomposition
decomp

wwz2kww()
Example ICIO data
data("leather")

Author(s)
Bastiaan Quast <bquast@gmail.com>
Fei Wang
Victor Stolzenburg
Sebastian Krantz

References

See Also
https://qua.st/decompr/

---

**decomp**

*Interface Function for Decompositions*

**Description**

This function loads an ICIO table and runs a specified decomposition. It provides a compact interface for quick analysis.

**Usage**

```r
decomp(
  iot,
  x,
  y,
  k,
  i,
  o = NULL,
  v = NULL,
  method = c("leontief", "kww", "wwz"),
  ...
)
```
Arguments

- **iot**
  - A Input Output Table object - a list with elements 'inter' (= x), 'final' (= y), 'output' (= o), 'countries' (= k) and 'industries' (= i) of class 'iot'. Alternatively these objects can be passed directly to the function, at least x, y, k and i need to be supplied.

- **x**
  - Intermediate demand table supplied as a numeric matrix of dimensions GN x GN (G = no. of country, N = no. of industries). Both rows and columns should be arranged first by country, then by industry (e.g. C1I1, C1I2, ..., C2I1, C2I2, ...) and should match (symmetry), such that rows and columns refer to the same country-industries.

- **y**
  - Final demand table supplied as a numeric matrix of dimensions GN x MN (M = no. of final demand categories recorded for each country). The rows of y need to match the rows of x, and the columns should also be arranged first by country, then by final demand category (e.g. C1FD1, C1FD2, ..., C2FD1, C2FD2, ...) with the order of the countries the same as in x.

- **k**
  - Character. A vector of country or region names of length G, arranged in the same order as they occur in the rows and columns of x, y.

- **i**
  - Character. A vector of country or region names of length N, arranged in the same order as they occur in the rows and columns of x and rows of y.

- **o**
  - Numeric. A vector of final outputs for each country-industry matching the rows of x and y. If not provided it will be computed as rowSums(x) + rowSums(y).

- **v**
  - Numeric. A vector of value added for each country-industry matching the columns of x. If not provided it will be computed as o - colSums(x).

- **method**
  - Character. The decomposition method, either "leontief", "kww" or "wwz".

- ... further arguments passed to leontief, kww or wzw.

Details

For more detailed analysis with multiple decompositions consider using load_tables_vectors to create a 'decompr' class object and then run the decomposition functions leontief, kww and wzw on the object.

Value

Depends on the decomposition, see leontief, kww or wzw.

Author(s)

Bastiaan Quast

References


**kww**

**See Also**

decompr-package

**Examples**

```r
# Load leather example data
data(leather)

data(leather)

# Explore the data
str(leather)

## Decomposing gross exports:

# Perform the Leontief decomposition
decomp(leather, method = "leontief")

# Perform the KWW decomposition
decomp(leather, method = "kww")

# Perform the WWZ decomposition
decomp(leather, method = "wwz")
```

---

**kww**

**Koopman-Wang-Wei Decomposition of Gross Exports**

**Description**

This function performs the Koopman-Wang-Wei (2014) decomposition of a countries gross exports into 9 separate value added components.

**Usage**

```r
kww(x)
```

**Arguments**

- `x` an object of the class `decompr` obtained from `load_tables_vectors`.

**Value**

A data frame where a countries gross exports is decomposed into 9 components (columns), as detailed in Figure 1 of the AER paper:

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DVA_FIN</td>
<td>Domestic VA in final goods exports.</td>
</tr>
</tbody>
</table>
DVA_INT  Domestic VA in intermediate exports absorbed by direct importers (used to produce a locally consumed final good).

DVA_INTrex  Domestic VA in intermediate exports reexported to third countries and absorbed there.

RDV_FIN  Domestic VA in intermediate exports that returns home via final imports.

RDV_INT  Domestic VA in intermediate exports that returns home via intermediate imports (used to produce a domestically consumed final good).

DDC  Double counted DVA in intermediate exports (arising from 2-way trade in intermediate goods).

FVA_FIN  Foreign VA in final goods exports.

FVA_INT  Foreign VA in intermediate exports.

FDC  Double counted FVA in intermediate exports (arising from 2-way trade in intermediate goods).

Author(s)
Sebastian Krantz

References

See Also
wwz, wwz2kww, decompr-package

Examples
```r
# Load example data
data(leather)

# Create intermediate object (class 'decompr')
decompr_object <- load_tables_vectors(leather)

# Perform the KWW decomposition
kww(decompr_object)
```
Leather Example ICIO Data

Description

An example 3 x 3 ICIO table describing a GVC for leather products with industries 'Agriculture', 'Textile and Leather' and 'Transport Equipment' for the countries 'Argentina', 'Turkey' and 'Germany'.

Usage

data("leather")

Format

A list of class 'iot' with the following elements:

inter 9 x 9 input output matrix where each column gives the value of inputs supplied to the corresponding country-industry by each row country-industry.

final 9 x 3 final demand matrix showing the final demand in each country (column) for each country-industry’s (rows) produce.

countries character vector of country names (matching columns of final).

industries character vector of industries, such that as.vector(t(outer(countries, industries, FUN = paste, sep = "."))) generates the row- and column-names of inter and the row-names of final.

out A vector of gross country-industry output. In a complete productive system it should be equal to rowSums(inter) + rowSums(final).

See Also
decompr-package

Leontief Decomposition

Description

The Leontief decomposition of gross flows (exports, final demand, output) into their value added origins.

Usage

leontief(x, post = c("exports", "output", "final_demand", "none"), long = TRUE)
Arguments

x  
an object of class decompr.
post  
post-multiply the value added multiplier matrix \([VB = V(I - A)^{-1}]\) with something to deduce the value added origins thereof. The default is "exports" \(VAE = V(I - A)^{-1}E\), where \(E\) is a diagonal matrix with exports along the diagonal yielding the country-industry level sources of value added (rows) for each using (column) country-industry; similarly for "output". Option "final_demand" computes value added origins of final demand by source country-industry and importing country, by computing \(VAY = V(I - A)^{-1}Y\) where \(Y\) is the corresponding GN x G matrix contained in \(x\). Option "none" just returns \(VB\) which gives the value added shares.

long  
logical. Transform the output data into a long (tidy) data set or not, default is TRUE.

Details

The Leontief decomposition is obtained by pre-multiplying the flow measure (e.g. exports) with the value added multiplier matrix \([VB = V(I - A)^{-1}]\), obtained by pre-multiplying the Leontief Inverse matrix \([B = (I - A)^{-1}]\) with a diagonal matrix \([V]\) containing the direct value added share in each industries output.

\(V\) is obtained as diag(\(v / o\)) where \(o\) is total industry output. \(v\) is either supplied to load_tables_vectors or computed as \(o - \text{colSums}(x)\) with \(x\) the raw IO matrix. If \(o\) is not supplied to load_tables_vectors, it is computed as \(\text{rowSums}(x) + \text{rowSums}(y)\) where \(y\) is the matrix of final demands. If both \(o\) and \(v\) are not supplied to load_tables_vectors, this is equivalent to computing \(V\) as diag(1 - \(\text{colSums}(A)\)), with \(A\) is the row-normalized IO matrix also used to compute the Leontief Inverse [B].

Value

If \(\text{long} = \text{TRUE}\) a molten data frame containing the elements of the decomposed flows matrix in the final column, preceded by several identifier columns. If \(\text{long} = \text{FALSE}\) the decomposed flows matrix is simply returned.

Author(s)

Bastiaan Quast

References


See Also

kww, wwz, decompr-package
Examples

# Load example data
data(leather)

# Create intermediate object (class 'decompr')
decompr_object <- load_tables_vectors(leather)

# Perform the Leontief decomposition of each country-industries
# exports into their value added origins by country-industry
leontief(decompr_object)

Description

This function loads the demand tables and defines all variables for the decomposition. It is kept in the package for backward compatibility. New users should use `load_tables_vectors` instead.

Usage

load_tables(x, y)

Arguments

x
the intermediate demand table, it has dimensions GN x GN (G = no. of country, N = no. of industries), excluding the first row and the first column which contains the country names, and the second row and second column which contain the industry names for each country. In addition, an extra row at the end should contain final demand.

y
the final demand table it has dimensions GN x MN, excluding the first row and the first column which contains the country names, the second column which contains the industry names for each country, and second row which contains the five decomposed final demands (M).

Details

Adapted from code by Fei Wang.

Value

A 'decompr' class object.

Author(s)

Bastiaan Quast
See Also

load_tables_vectors

---

load_tables_vectors  Load the Input-Output and Final Demand Tables

Description

This function loads the demand tables and creates all matrices and variables required for the GVC decompositions.

Usage

```r
load_tables_vectors(
  iot,
  x,
  y,
  k,
  i,
  o = NULL,
  v = NULL,
  null_inventory = FALSE
)
```

Arguments

- `iot`: a Input Output Table object - a list with elements 'inter' (= x), 'final' (= y), 'output' (= o), 'countries' (= k) and 'industries' (= i) of class 'iot'. Alternatively these objects can be passed directly to the function, at least x, y, k and i need to be supplied.

- `x`: intermediate demand table supplied as a numeric matrix of dimensions GN x GN (G = no. of country, N = no. of industries). Both rows and columns should be arranged first by country, then by industry (e.g. C1I1, C1I2, ..., C2I1, C2I2, ...) and should match (symmetry), such that rows and columns refer to the same country-industries.

- `y`: final demand table supplied as a numeric matrix of dimensions GN x MN (M = no. of final demand categories recorded for each country). The rows of y need to match the rows of x, and the columns should also be arranged first by country, then by final demand category (e.g. C1FD1, C1FD2, ..., C2FD1, C2FD2, ...) with the order of the countries the same as in x.

- `k`: character. A vector of country or region names of length G, arranged in the same order as they occur in the rows and columns of x and y.

- `i`: character. A vector of country or region names of length N, arranged in the same order as they occur in the rows and columns of x and y.
numeric. A vector of final outputs for each country-industry matching the rows of x and y. If not provided it will be computed as rowSums(x) + rowSums(y).

v numeric. A vector of value added for each country-industry matching the columns of x. If not provided it will be computed as o - colSums(x).

null_inventory logical. TRUE sets the inventory (last final demand category for each country) to zero.

Details
Adapted from code by Fei Wang.

Value
A 'decompr' class object - a list with the following elements:

- \( A \) Imported/Exported goods IO shares matrix (x column-normalized by output o, with domestic entries set to 0).
- \( B \) Leontief Inverse matrix \((I - A)^{-1}\) where \( A \) is x column-normalized by output o.
- \( Bd \) Domestic part of Leontief Inverse matrix (inter-country elements of \( B \) set to 0, needed for WWZ decomposition).
- \( Bm \) Imported/Exported part of Leontief Inverse matrix (domestic elements of \( B \) set to 0, needed for WWZ decomposition).
- \( L \) Domestic economy Leontief Inverse matrix \((I - Ad)^{-1}\) where \( Ad \) is \( A \) with all inter-country elements set to 0.
- \( E \) Total Exports (output of each country-industry servicing foreign production or foreign final demand).
- \( ESR \) Total Exports by destination country.
- \( Eint \) Exports for intermediate production by destination country.
- \( Ef \) Exports for final demand by destination country.
- \( Vc \) Value added content of output \((v / o)\).
- \( G \) Number of countries.
- \( N \) Number of industries.
- \( GN \) Number of country-industries.
- \( k \) Vector of country names.
- \( i \) Vector of industry names.
- rownam Unique country-industry names identifying the rows / columns of x and rows of y.
- \( X \) Total Output \((= o)\).
- \( Y \) Total Final Demand by destination country.
- \( Yd \) Domestic Final Demand.
- \( YM \) Foreign Final Demand.

Author(s)
Bastiaan Quast

See Also
leontief, kww, wwz, decompr-package

Examples
# Load example data
data(leather)
# Create intermediate object (class 'decompr')
decompr_object <- load_tables_vectors(leather)

# Examine the object
str(decompr_object)

---

**Wang-Wei-Zhu Decomposition of Gross Exports**

**Description**

This function performs the Wang-Wei-Zhu decomposition of country-sector level gross exports into 16 value added components by importing country.

**Usage**

```r
wwz(x, verbose = FALSE)
```

**Arguments**

- `x`: an object of the class 'decompr' obtained from `load_tables_vectors`.
- `verbose`: logical, should timings of the calculation be displayed? Default is FALSE

**Details**

Adapted from code by Fei Wang.

**Value**

A data frame with exports decomposed into 16 components (columns), as detailed in Table E1 in the appendix of the paper, and additional diagnostic items:

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DVA_FIN</td>
<td>Domestic VA in final goods exports.</td>
</tr>
<tr>
<td>DVA_INT</td>
<td>Domestic VA in intermediate exports used by direct importer to produce domestic final goods consumed at home.</td>
</tr>
<tr>
<td>DVA_INTrexI1</td>
<td>Domestic VA in intermediate exports used by the direct importer to produce intermediate exports for consumption in third countries.</td>
</tr>
<tr>
<td>DVA_INTrexF</td>
<td>Domestic VA in intermediate exports used by the direct importer to produce final goods exports to third countries.</td>
</tr>
<tr>
<td>Diagnostic Item</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>texp</td>
<td>Total Exports (matrix 'ESR' from load_tables_vectors).</td>
</tr>
<tr>
<td>texpint</td>
<td>Exports for intermediate production (matrix 'Eint' from load_tables_vectors).</td>
</tr>
</tbody>
</table>
Exports for final demand (matrix 'Efd' from \texttt{load_tables_vectors}).

Difference between Total Exports and the sum of the 16 terms.

... in percent of total exports.

Difference between Final Exports and the sum of terms DVA\_FIN, OVA\_FIN and MVA\_FIN.

... in percent of final exports.

Difference between Intermediate Exports and the sum of all the remaining terms (except DVA\_FIN, OVA\_FIN and MVA\_FIN).

... in percent of intermediate exports.

DVA embodied in gross exports based on forward linkage.

Bastiaan Quast


kww, ww2kww, decompr-package

# Load example data
data(leather)

decompr.object <- load_tables_vectors(leather)

wwz(decompr.object)
wwz2kww

Koopman-Wang-Wei from Wang-Wei-Zhu Decomposition

Description

This function by default returns a disaggregated version of the Koopman-Wang-Wei (KWW) decomposition breaking up sector-level gross exports into 9 value added terms, from an already computed and more detailed (16 term) Wang-Wei-Zhu decomposition of sector-level gross exports. An aggregation option also allows obtaining the aggregate KWW decomposition.

Usage

wwz2kww(x, aggregate = FALSE)

Arguments

x

a data frame with the WWZ decomposition obtained from wwz. Alternatively a 'decompr' class object from load_tables_vectors can be supplied, which will toggle calling wwz() first.

aggregate

logical. TRUE aggregates the KWW decomposition to the country level, giving exactly the same output as kww. FALSE maintains the sector level decomposition in KWW format.

Details

The mapping of the 16 terms in the WWZ decomposition to the 9 terms in the KWW decomposition is provided in table E2 in the appendix of the WWZ (2013) paper. The table is reproduced here using the term naming conventions followed in this package.

<table>
<thead>
<tr>
<th>WWZ Terms</th>
<th>KWW Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DVA_FIN</td>
<td>DVA_FIN</td>
<td>Domestic VA in final goods exports.</td>
</tr>
<tr>
<td>DVA_INT, DVA_INTrexl</td>
<td>DVA_INT</td>
<td>Domestic VA in intermediate exports absorbed by direct importer</td>
</tr>
<tr>
<td>DVA_INTrexF, DVA_INTrexl2</td>
<td>DVA_INTrexF</td>
<td>Domestic VA in intermediate exports reexported to third countries</td>
</tr>
<tr>
<td>RDV_FIN, RDV_FIN2</td>
<td>RDV_FIN</td>
<td>Domestic VA in intermediate exports that returns home via final imports</td>
</tr>
<tr>
<td>RDV_INT</td>
<td>RDV_INT</td>
<td>Domestic VA in intermediate exports that returns via intermediates</td>
</tr>
</tbody>
</table>
### Value

A data frame with exports decomposed into 9 components (columns), see the table above and `kww` for a shorter description of the 9 terms.

### Note

If both WWZ and KWW decompositions are required, it is computationally more efficient to call `wwz2kww(x, aggregate = TRUE)` on an already computed WWZ decomposition, than to call `kww` on a ‘decompr’ object.

### Author(s)

Sebastian Krantz

### References


### See Also

`wwz, kww, decompr-package`

### Examples

```r
# Load example data
data(leather)

# Create intermediate object (class 'decompr')
decompr_object <- load_tables_vectors(leather)

# Perform the WWZ decomposition
WWZ <- wwz(decompr_object)
```
# Obtain a disaggregated KWW decomposition
KWW <- wzw2kww(WWZ)

# Aggregate KWW
wzw2kww(WWZ, aggregate = TRUE)

# Same as running KWW directly, but the former is more efficient
# if we already have the WWZ
kww(decompr_object)
Index

data(leather), 3
decomp, 3
decomp(), 2
decompr-package, 2

kww, 4, 5, 8, 11, 14–16
kww(), 2

leather, 7
leontief, 4, 7, 11
leontief(), 2
load_tables, 9
load_tables_vectors, 4, 5, 8–10, 10, 12–15
load_tables_vectors(), 2

wwz, 4, 6, 8, 11, 12, 15, 16
wwz(), 2
wwz2kww, 6, 14, 15
wwz2kww(), 3