Package: dfvad (via r-universe)

August 21, 2024

Type Package

Title Diewert and Fox's Method of Value Added Growth Decomposition

Version 0.3.6

Author Shipei Zeng

Maintainer Shipei Zeng <shipei.zeng@unswalumni.com>

Description Decomposing value added growth into explanatory factors. A cost constrained value added function is defined to specify the production frontier. Industry estimates can also be aggregated using a weighted average approach. Details about the methodology and data can be found in Diewert and Fox (2018) <doi:10.1093/oxfordhb/9780190226718.013.19> and Zeng, Parsons, Diewert and Fox (2018) <https://www.business.unsw.edu.au/research-site/centreforappliedeconomicresearch-site/Documents/emg2018-6_SZeng_EMG-Slides.pdf>.

Depends R (>= 3.5.0)

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.0.0

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

Imports stats

URL https://github.com/shipei-zeng/dfvad

BugReports https://github.com/shipei-zeng/dfvad/issues

Repository https://shipei-zeng.r-universe.dev

RemoteUrl https://github.com/shipei-zeng/dfvad

RemoteRef HEAD

RemoteSha 4ece0fbd7ec00930d97bce05d847a78b008cdfee

Contents

`dynamics` *Productivity Dynamics*

Description

Productivity dynamics reflect firm contributions to productivity growth over periods when firms enter or exit an industry. `dynamics` summarises a series of decomposition methods that are centred on the contributions from incumbents, entrants and exits. It applies to other weighted aggregation measures analogous to aggregate productivity as well.

Usage

```
dynamics(df, x, s, id, tm, typ = "df")
```

Arguments

`df`	A data frame sorted by the time period column.
`x`	A string indicating the productivity (or analogous measures) column.
`s`	A string indicating the market share column.
`id`	A string indicating the identity column.
`tm`	A string indicating the time period column.
`typ`	Relevant types of productivity dynamics. Options include "df" for Diewert-Fox decomposition (by default), "bhc" for Baily-Hulten-Campbell, "gr" for Griliches-Regev, "fhk" for Foster-Haltiwanger-Krizan, "bg" for Baldwin-Gu, and "mp" for Melitz-Polanec.

Value

A data frame consisting of the time period and firm contributions.

Examples

```
# Use the built-in data set "firms"
# DF decomposition of firm dynamics
dym_df <- dynamics(firms, "tfp", "s", "id", "t")
# BG decomposition of firm dynamics
dym_bg <- dynamics(firms, "tfp", "s", "id", "t", "bg")
```

`firms` *Sample data for firm dynamics*

Description

Firm productivity and market shares adopted to demonstrate the decomposition of firm dynamics

Usage

```
firms
```

Format

A data frame with the following columns:

tfp Firm productivity (or firm performance indicators)

s Market shares occupied the firm.

id A firm identity column.

t A time period column

References

Zeng, S. (2019). Frontier firms, inefficiency and productivity dynamics. Presented in EMG Workshop 2019, Sydney.

`mining` *Sample Data for Value Added Decomposition*

Description

Mining inputs and outputs adopted to demonstrate the decomposition of value added growth.

Usage

```
mining
```

Format

A data frame with the following columns:

year A time period column.

p2 Output prices.

w2 Wages of labour inputs.

u2 Prices of capital services.

y2 Output quantities.

h2 Labour input quantities.

x2 Capital services quantities.

References

Zeng, S., Parsons, S., Diewert, W. E. and Fox, K. J. (2018). Industry and state level value added and productivity decompositions. Presented in EMG Worshop 2018, Sydney.

`pb_index` *Bilateral Price Indexes*

Description

Bilateral indexes refer to the case when only two periods are compared each time. `pb_index()` computes price indexes in a bilateral approach.

Usage

```
pb_index(df, p, qty, id, tm, typ = "f", seq = "ch", bsk = "flx")
```

Arguments

`df`	A data frame sorted by the time period column.
`p`	A string indicating the price column.
`qty`	A string indicating the quantity column.
`id`	A string indicating the identity column.
`tm`	A string indicating the time period column. Each period must contain two observations at least.
`typ`	Relevant types of price indexes. Options include "f" for Fisher price indexes (by default), "t" for Tornqvist price indexes, "l" for Laspeyres price indexes, and "p" for Paasche price indexes.
`seq`	Index construction sequences when the number of periods is larger than 2. Options include "ch" for chained indexes (by default), and "fb" for fixed base indexes.
`bsk`	The choice of baskets when items are not matched over multiple periods. Options include "flx" (by default) for a flexible basket that varies depending on the maximal number of matched items in two periods each time, and "cst" for a constant basket that takes the maximal number of matched items across all periods.

Value

A data frame consisting of the time period and price indexes.

Examples

```
# Use the built-in data set "prices"
# Laspeyres fixed base indexes with a constant basket
df <- prices[[1]]
df <- df[order(df[,"t"]),]
index1 <- pb_index(df, "p", "q", "id", "t", typ = "l", seq = "fb", bsk = "cst")
# Fisher chained indexes with a flexible basket
df <- prices[[2]]
df <- df[order(df[,"t"]),]
index2 <- pb_index(df, "p", "q", "id", "t")
```

---

pm_index *Multilateral Price Indexes*

Description

Multilateral indexes refer to the case when more than two periods are compared each time. `pm_index()` computes price indexes in a multilateral approach.

Usage

```
pm_index(df, p, qty, id, tm, typ = "geks", len = NULL,
lnk = NULL, bsk = "flx", wd = "flx")
```

Arguments

`df`	A data frame sorted by the time period column.
`p`	A string indicating the price column.
`qty`	A string indicating the quantity column.
`id`	A string indicating the identity column.
`tm`	A string indicating the time period column. Each period must contain two observations at least.
`typ`	Relevant types of price indexes. Options include "geks" for GEKS price indexes (by default), "ccdi" for CCDI price indexes, "wtpd" for the weighted time product dummy method, and "gk" for the Geary-Khamis method.
`len`	Window length for linked indexes using rolling windows. A single window is set as NULL (by default).
`lnk`	Linking position in rolling windows, effective when ‘len‘ is not NULL. If no linking position is provided, it should be set as NULL (by default). Other options include "mean" for mean splices and numbers for specific cases.

bsk — The choice of baskets when items are not matched over multiple periods. Options include "flx" (by default) for a flexible basket that varies depending on the maximal number of matched items in two periods each time, and "cst" for a constant basket that takes the maximal number of matched items across all periods.

wd — The choice of windows when items are not matched over multiple windows. Options include "flx" (by default) for a flexible window that allows for different items in two windows each time, and "cst" for a constant window that takes the maximal number of matched items across all windows.

Value

A data frame consisting of the time period and price indexes.

Examples

```
# Use the built-in data set "prices"
# matched items
df <- prices[[1]]
df <- df[order(df[,"t"]),]
# GEKS price indexes with a constant basket over periods
index1 <- pm_index(df, "p", "q", "id", "t", typ = "geks", bsk = "cst")
# unmatched items
df_add <- matrix(c(1, 6, 12, 5, 6, 7, 0.5, 0.5, 0.5, 9, 9, 9), nrow=3)
df_add <- as.data.frame(df_add)
colnames(df_add) <- colnames(df)
df <- rbind(df, df_add)
df <- df[order(df[,"t"]),]
# CCDI price indexes with a flexible basket over periods
index2 <- pm_index(df, "p", "q", "id", "t", typ = "ccdi", bsk = "flx")
# CCDI price indexes with rolling windows (window length at 11, linking at the first observation)
index3 <- pm_index(df, "p", "q", "id", "t", typ = "ccdi", len = 11, lnk = 1)
# CCDI price indexes with rolling windows (window length at 11, linking with mean splices)
index4 <- pm_index(df, "p", "q", "id", "t", typ = "ccdi", len = 11, lnk = "mean")
```

---

prices *Sample Data for Price Indexes*

Description

Prices, quantities, identities and time periods adopted to demonstrate the computation of price indexes.

Usage

```
prices
```

Format

A list of data frames with the following columns:

t A time period column.

id An identity column.

p A price column.

q A quantity column.

These data frames are produced with different elasticities that can be specified by the sub-list names.

References

Diewert, W.E., and Fox, K. J. 2018. Substitution bias in multilateral methods for CPI construction using scanner data. Discussion Papers 2018-13, School of Economics, the University of New South Wales.

`roll_div` *Converting Level Values to Growth Values*

Description

`roll_div()` converts level values to growth values for a vector.

Usage

```
roll_div(x)
```

Arguments

`x`	A vector with level values.

Value

A vector of growth values.

Examples

```
table2 <- value_decom(c("h2","x2"), c("w2","u2"), "y2", "p2", "year", mining)[[2]]
roll_div(table2[, "TFP"])
```

roll_prod *Converting Growth Values to Level Values*

Description

`roll_prod()` converts growth values to level values for a vector.

Usage

```
roll_prod(x)
```

Arguments

x	A vector with growth values.

Value

A vector of level values.

Examples

```
table1 <- value_decom(c("h2","x2"), c("w2","u2"), "y2", "p2", "year", mining)[[1]]
roll_prod(table1[, "TFPG"])
```

sector *Sample Data for Weighted Average Aggregation*

Description

Explanatory factors of value added decomposition adopted to demonstrate the aggregation over industries.

Usage

```
sector
```

Format

A data frame with the following columns:

year A time period column.

p Output prices.

y Output quantities.

alpha Net output price indexes.

beta Input quantity indexes

gamma Input mix indexes.

epsilon Value added efficiency indexes.

tau Technical progress indexes.

industry Industry codes.

References

Zeng, S., Parsons, S., Diewert, W. E. and Fox, K. J. (2018). Industry and state level value added and productivity decompositions. Presented in EMG Worshop 2018, Sydney.

`t_weight` *Aggregation over Sectors with a Weighted Average Approach*

Description

This "bottom up" approach uses weighted averages of the sectoral decompositions to provide an approximate decomposition into explanatory components at the aggregate level. Specifically, the Tornqvist index is adopted in the aggregation.

Usage

```
t_weight(y, p, id, t, alpha, beta, gamma, epsilon, tau, data)
```

Arguments

`y`	A string (or a vector of strings) indicating the output quantity columns.
`p`	A string (or a vector of strings) indicating the output price columns.
`id`	A string indicating the industry column.
`t`	A string indicating the time period column.
`alpha`	A string indicating net output price indexes.
`beta`	A string indicating input quantity indexes.
`gamma`	A string indicating input mixe indexes.
`epsilon`	A string indicating value added efficiency indexes.
`tau`	A string indicating technical progress indexes.
`data`	A data frame containing input prices, input quantities, industry identities, the time period, and explanatory factors of value added growth.

Value

A list containing a growth-value table and a level-value table of explanatory factors for value added growth decomposition. It is sorted by the time period.

References

Diewert, W. E. and Fox, K. J. (2018). Decomposing value added growth into explanatory factors. In The Oxford Handbook of Productivity Analysis, chapter 19, page 625–662. Oxford University Press: New York.

Examples

```
# Use the built-in dataset "sector"
table1 <- t_weight("y", "p", "industry", "year", "alpha",
        "beta", "gamma", "epsilon", "tau", sector)[[1]]
table2 <- t_weight("y", "p", "industry", "year", "alpha",
        "beta", "gamma", "epsilon", "tau", sector)[[2]]
```

---

value_decom *Decomposing Value Added Growth into Explanatory Factors*

Description

This method for decomposing nominal value added growth is proposed by Diewert and Fox (2018), which identifies the contributions from efficiency change, growth of primary inputs, changes in output and input prices, technical progress and returns to scale.

Usage

```
value_decom(x, w, y, p, t, data)
```

Arguments

x	A string (or a vector of strings) indicating the quantity columns.
w	A string (or a vector of strings) indicating the input price columns.
y	A string (or a vector of strings) indicating the the output quantity columns.
p	A string (or a vector of strings) indicating the the output price columns.
t	A string indicating the time period column.
data	A data frame containing input prices, input quantities, output prices, output quantities, and the time period.

Value

A list containing a growth-value table and a level-value table of explanatory factors for value added growth decomposition. It is sorted by the time period.

References

Diewert, W. E. and Fox, K. J. (2018). Decomposing value added growth into explanatory factors. In The Oxford Handbook of Productivity Analysis, chapter 19, page 625–662. Oxford University Press: New York.

Examples

```
# Use the built-in dataset "mining"
table1 <- value_decom(c("h2","x2"), c("w2","u2"), "y2", "p2", "year", mining)[[1]]
table2 <- value_decom(c("h2","x2"), c("w2","u2"), "y2", "p2", "year", mining)[[2]]
```

Index