Package ‘npsf’

July 22, 2025

Type Package

Title Nonparametric and Stochastic Efficiency and Productivity Analysis

Version 0.8.0

Date 2020-11-22

Author Oleg Badunenko [aut, cre],
Pavlo Mozharovskyi [aut],
Yaryna Kolomiytseva [aut]

Maintainer Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

Description Nonparametric efficiency measurement and statistical inference via DEA type estimators (see Färe, Grosskopf, and Lovell (1994) <doi:10.1017/CBO9780511551710>, Kneip, Simar, and Wilson (2008) <doi:10.1017/S0266466608080651> and Badunenko and Mozharovskyi (2020) <doi:10.1080/01605682.2019.1
tic Frontier estimators for both cross-sectional data and 1st, 2nd, and 4th generation models for panel data (see Kumbhakar and Lovell (2003) <doi:10.1017/CBO9781139174411>, Badunenko and Kumbhakar (2016) <doi:10.1016/j.ejor.2016.04.049>). The stochastic frontier estimators can handle both half-normal and truncated normal models with conditional mean and heteroskedasticity. The marginal effects of determinants can be obtained.

Depends Formula

Suggests snowFT, Rmpi

Encoding UTF-8

License GPL-2

LinkingTo Rcpp

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-11-22 22:10:02 UTC

Contents

`npsf-package` *Introduction to Nonparametric and Stochastic Frontier Analysis*

Description

This package provides a variety of tools for nonparametric and parametric efficiency measurement.

Details

The nonparametric models in `npsf` comprise nonradial efficiency measurement (`tenonradial`), where non-proportional reductions (expansions) in each positive input (output) are allowed, as well as popular radial efficiency measurement (`teradial`), where movements to the frontier are proportional.

Using bootstrapping techniques, `teradialbc`, `tenonradialbc`, `nptestrts`, `nptestind` deal with statistical inference about the radial efficiency measurement. `nptestind` helps in deciding which type of the bootstrap to employ. Global return to scale and individual scale efficiency is tested by `nptestrts`. Finally, `teradialbc` and `tenonradialbc`, performs bias correction of the radial Debrue-Farrell and nonradial Russell input- or output-based measure of technical efficiency, computes bias and constructs confidence intervals.

Computer intensive functions `teradialbc` and `nptestrts` allow making use of parallel computing, even on a single machine with multiple cores. Help files contain examples that are intended to introduce the usage.

The parametric stochastic frontier models in `npsf` can be estimated by `sf`, which performs maximum likelihood estimation of the frontier parameters and technical or cost efficiencies. Inefficiency error component can be assumed to be have either half-normal or truncated normal distribution. `sf` allows modelling multiplicative heteroskedasticity of either inefficiency or random noise component, or both. Additionally, marginal effects of determinants on the expected value of inefficiency term can be computed.

For details of the respective method please see the reference at the end of this introduction and of the respective help file.

All function in `npsf` accept formula with either names of variables in the data set, or names of the matrices. Except for `nptestind`, all function return `esample`, a logical vector length of which is determined by `data` and `subset` (if specified) or number of rows in matrix `outputs`. `esample` equals `TRUE` if this data point parted in estimation procedure, and `FALSE` otherwise.

Results can be summarized using `summary.npsf`.

Author(s)

Oleg Badunenko, <oleg.badunenko@brunel.ac.uk>

Pavlo Mozharovskyi, <pavlo.mozharovskyi@telecom-paris.fr>

Yaryna Kolomiytseva, <kolomiytseva@wiso.uni-koeln.de>

Maintainer: Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

References

Badunenko, O. and Kumbhakar, S.C. (2016), When, Where and How to Estimate Persistent and Transient Efficiency in Stochastic Frontier Panel Data Models, *European Journal of Operational Research*, **255**(1), 272–287, doi: 10.1016/j.ejor.2016.04.049

Badunenko, O. and Mozharovskyi, P. (2016), Nonparametric Frontier Analysis using Stata, *Stata Journal*, **16**3, 550–89, doi: 10.1177/1536867X1601600302

Badunenko, O. and Mozharovskyi, P. (2020), Statistical inference for the Russell measure of technical efficiency, *Journal of the Operational Research Society*, **71**3, 517–527, doi: 10.1080/01605682.2019.1599778

Bartelsman, E.J. and Gray, W. (1996), The NBER Manufacturing Productivity Database, *National Bureau of Economic Research*, Technical Working Paper Series, doi: 10.3386/t0205

Battese, G., Coelli, T. (1988), Prediction of firm-level technical effiiencies with a generalized frontier production function and panel data. *Journal of Econometrics*, **38**, 387–399

Battese, G., Coelli, T. (1992), Frontier production functions, technical efficiency and panel data: With application to paddy farmers in India. *Journal of Productivity Analysis*, **3**, 153–169

Charnes, A., W. W. Cooper, and E. Rhodes. 1981. Evaluating Program and Managerial Efficiency: An Application of Data Envelopment Analysis to Program Follow Through. *Management Science* **27**: 668–697

Caudill, S., Ford, J., Gropper, D. (1995), Frontier estimation and firm-specific inefficiency measures in the presence of heteroscedasticity. *Journal of Business and Economic Statistics*, **13**, 105–111

Debreu, G. 1951. The Coefficient of Resource Utilization. *Econometrica* **19**: 273–292

Färe, R. and Lovell, C. A. K. (1978), Measuring the technical efficiency of production, *Journal of Economic Theory*, **19**, 150–162, doi: 10.1016/00220531(78)900601

Färe, R., Grosskopf, S. and Lovell, C. A. K. (1994), *Production Frontiers*, Cambridge U.K.: Cambridge University Press, doi: 10.1017/CBO9780511551710

Farrell, M. J. 1957. The Measurement of Productive Efficiency. *Journal of the Royal Statistical Society. Series A (General)* **120**(3): 253–290

Heston, A., and R. Summers. 1991. The Penn World Table (Mark 5): An Expanded Set of International Comparisons, 1950-1988. *The Quarterly Journal of Economics* **106**: 327–368

Horrace, W. and Schmidt, P. (1996), On ranking and selection from independent truncated normal distributions. *Journal of Productivity Analysis*, **7**, 257–282

Jondrow, J., Lovell, C., Materov, I., Schmidt, P. (1982), On estimation of technical inefficiency in the stochastic frontier production function model. *Journal of Econometrics*, **19**, 233–238

Kneip, A., Simar L., and P.W. Wilson (2008), Asymptotics and Consistent Bootstraps for DEA Estimators in Nonparametric Frontier Models, *Econometric Theory*, **24**, 1663–1697, doi: 10.1017/S0266466608080651

Koetter, M., Kolari, J., and Spierdijk, L. (2012), Enjoying the quiet life under deregulation? Evidence from adjusted Lerner indices for U.S. banks. *Review of Economics and Statistics*, **94**, 2, 462–480

Kumbhakar, S. (1990), Production Frontiers, Panel Data, and Time-varying Technical Inefficiency. *Journal of Econometrics*, **46**, 201–211

Kumbhakar, S. and Lovell, C. (2003), *Stochastic Frontier Analysis.* Cambridge: Cambridge University Press, doi: 10.1017/CBO9781139174411

Restrepo-Tobon, D. and Kumbhakar, S. (2014), Enjoying the quiet life under deregulation? Not Quite. *Journal of Applied Econometrics*, **29**, 2, 333–343

Simar, L. and P.W. Wilson (1998), Sensitivity Analysis of Efficiency Scores: How to Bootstrap in Nonparametric Frontier Models, *Management Science*, **44**, 49–61, doi: 10.1287/mnsc.44.1.49

Simar, L. and P.W. Wilson (2000), A General Methodology for Bootstrapping in Nonparametric Frontier Models, *Journal of Applied Statistics*, **27**, 779–802, doi: 10.1080/02664760050081951

Simar, L. and P.W. Wilson (2002), Nonparametric Tests of Return to Scale, *European Journal of Operational Research*, **139**, 115–132

Wang, H.-J. (2002), Heteroskedasticity and non-monotonic efficiency effects of a stochastic frontier model. *Journal of Productivity Analysis*, **18**, 241–253

Wilson P.W. (2003), Testing Independence in Models of Productive Efficiency, *Journal of Productivity Analysis*, **20**, 361–390, doi: 10.1023/A:1027355917855

`banks00_07` *U.S. Commercial Banks Data*

Description

`banks00_07` data frame contains selected variables for 500 (randomly sampled from around 5000) U.S. commercial banks from data of Koetter et al. (2012) for years 2000-2007. This data are used for illustrution purposes and are not suitable for research purposes.

Usage

```
data(banks00_07)
```

Format

This data frame contains the following variables (columns):

`year` Year.

`id` Entity ID.

`TA` Gross total assets.

`LLP` Loan loss provisions.

`Y1` Total securities (in thousands of US dollars).

`Y2` Total loans and leases (in thousands of US dollars).

`W1` Cost of fixed assets divided by the cost of borrowed funds.

`W2` Cost of labor (in thousands of US dollars) divided by the cost of borrowed funds.

`ER` Gross total equity to gross total assets ratio.

`TC` Total operating cost.

`LA` Total loans and leases to gross total assets ratio.

`Ti` Times bank is observed.

`TA_ave` Mean value of TA.

`TA_initial` Value of TA in the first observed year.

`LLP_ave` Mean value of LLP.

`LLP_initial` Value of LLP in the first observed year.

`ER_ave` Mean value of ER.

`ER_initial` Value of ER in the first observed year.

`LA_ave` Mean value of LA.

`LA_initial` Value of LA in the first observed year.

Details

The data were sampled and generated as shown in section "Examples".

Source

http://qed.econ.queensu.ca/jae/2014-v29.2/restrepo-tobon-kumbhakar/.

References

Koetter, M., Kolari, J., and Spierdijk, L. (2012), Enjoying the quiet life under deregulation? Evidence from adjusted Lerner indices for U.S. banks. *Review of Economics and Statistics*, **94**, 2, 462–480.

Restrepo-Tobon, D. and Kumbhakar, S. (2014), Enjoying the quiet life under deregulation? Not Quite. *Journal of Applied Econometrics*, **29**, 2, 333–343.

Examples

```
## Not run:

# Download data from the link in "Source"

banks00_07 <- read.delim("2b_QLH.txt")

# rename 'entity' to 'id'
colnames(banks00_07) [colnames(banks00_07) == "entity"] <- "id"

table(banks00_07$year)
# keep if 2000 -- 2007

banks00_07 <-
 banks00_07[(banks00_07$year >= 2000 & banks00_07$year <= 2007),]
dim(banks00_07)

q1q3 <- quantile(banks00_07$TA, probs = c(.25,.75))

banks00_07 <-
 banks00_07[(banks00_07$TA >= q1q3[1] & banks00_07$TA <= q1q3[2]),]
dim(banks00_07)


# generate required variables
banks00_07$TC <-banks00_07$TOC
banks00_07$ER <- banks00_07$Z / banks00_07$TA
banks00_07$LA <- banks00_07$Y2 / banks00_07$TA

banks00_07 <-
 banks00_07[, colnames(banks00_07)
 c("id", "year", "Ti", "TC", "Y1", "Y2", "W1","W2", "ER", "LA", "TA", "LLP")]
dim(banks00_07)

t0 <- as.vector( by(data = banks00_07$id,
                    INDICES = banks00_07$id,
                    FUN = function(qq) length(qq)) )
banks00_07$Ti <- rep(t0, times = t0)
banks00_07 <- banks00_07[banks00_07$Ti > 4,]

# complete observations
banks00_07 <- banks00_07[complete.cases(banks00_07),]
dim(banks00_07)

id_names <- unique(banks00_07$id)
N_total <- length(id_names)
set.seed(816376586)
ids_n2choose <- sample(1:N_total, 500)
ids2choose <- id_names[ids_n2choose]
banks00_07 <- banks00_07[banks00_07$id
dim(banks00_07)
```

```
t0 <- as.vector( by(data = banks00_07$id,
                    INDICES = banks00_07$id,
                    FUN = function(qq) length(qq)) )
length(rep(t0, times = t0))

banks00_07$Ti <- rep(t0, times = t0)

banks00_07[1:50,c("id","year","Ti")]

# keep if Ti > 4

banks00_07 <- banks00_07[banks00_07$Ti > 4,]
dim(banks00_07)

# sort
banks00_07 <- banks00_07[order(banks00_07$id, banks00_07$year),]

# TC = TOC
#
# ER = Z / TA
# Gross total equity to gross total assets ratio.
#
# LA = Y2 / TA
# Total loans and leases to gross total assets ratio.

banks00_07$TA_ave <-
 rep(as.vector( by(data = banks00_07$TA,
                   INDICES = banks00_07$id,
                   FUN = function(qq) mean(qq))), times = t0)

banks00_07$TA_initial <-
 rep(as.vector( by(data = banks00_07$TA,
                   INDICES = banks00_07$id,
                   FUN = function(qq) qq[1])), times = t0)

banks00_07$LLP_ave <-
 rep(as.vector( by(data = banks00_07$LLP,
                   INDICES = banks00_07$id,
                   FUN = function(qq) mean(qq))), times = t0)

banks00_07$LLP_initial <-
 rep(as.vector( by(data = banks00_07$LLP,
                   INDICES = banks00_07$id,
                   FUN = function(qq) qq[1])), times = t0)

banks00_07$ER_ave <-
 rep(as.vector( by(data = banks00_07$ER,
                   INDICES = banks00_07$id,
                   FUN = function(qq) mean(qq))), times = t0)

banks00_07$ER_initial <-
 rep(as.vector( by(data = banks00_07$ER,
                   INDICES = banks00_07$id,
```

```
                    FUN = function(qq) qq[1])), times = t0)

banks00_07$LA_ave <-
 rep(as.vector( by(data = banks00_07$LA,
                    INDICES = banks00_07$id,
                    FUN = function(qq) mean(qq))), times = t0)

banks00_07$LA_initial <-
 rep(as.vector( by(data = banks00_07$LA,
                    INDICES = banks00_07$id,
                    FUN = function(qq) qq[1])), times = t0)

cols2export <- c("id","year","Ti","TA","TA_ave",
                  "TA_initial","LLP","LLP_ave",
                  "LLP_initial","ER_ave","ER_initial","LA_ave","LA_initial")

write.table(x = banks00_07, file = "banks00_07.txt", row.names = FALSE)


## End(Not run)
```

---

banks05 *U.S. Commercial Banks Data*

Description

banks05 data frame contains selected variables from the U.S. commercial banks data of Koetter et al. (2012) for year 2005 and 500 banks randomly sampled from around 5000. Dependent variable was randomly generated, as described under 'Details', to satisfy the assumptions of doubly heteroskedastic stochastic cost frontier model. This data is, therefore, not suitable for research purposes.

Usage

```
data(banks05)
```

Format

This data frame contains the following variables (columns):

`lnC` Randomly generated total operating costs.

`lnw1` Logarithm of the cost of fixed assets divided by the cost of borrowed funds.

`lnw2` Logarithm of the cost of labor (in thousands of US dollars) divided by the cost of borrowed funds.

`lny1` Logarithm of total securities (in thousands of US dollars).

`lny2` Logarithm of total loans and leases (in thousands of US dollars).

`ER` Gross total equity to gross total assets ratio.

`LA` Total loans and leases to gross total assets ratio.

Details

The variable representing total operating costs was generated as follows:

$$lnC = \beta 0 + \beta 1 * lnw1 + \beta 2 * lnw2 + \gamma 1 * lny1 + \gamma 2 * lny2 + \nu + u,$$

where ν $N(0, exp(\alpha 0 + \alpha 1 * LA))$ and u $N+(\delta 0 + \delta 1 * ER, exp(\omega 0 + \omega 1 * ER))$. More detailed description of input prices, outputs, and exogenous variables is provided in Koetter et al. (2012). See also related study of Restrepo-Tobon and Kumbhakar (2014).

Source

http://qed.econ.queensu.ca/jae/2014-v29.2/restrepo-tobon-kumbhakar/.

References

Koetter, M., Kolari, J., and Spierdijk, L. (2012), Enjoying the quiet life under deregulation? Evidence from adjusted Lerner indices for U.S. banks. *Review of Economics and Statistics*, **94**, 2, 462–480.

Restrepo-Tobon, D. and Kumbhakar, S. (2014), Enjoying the quiet life under deregulation? Not Quite. *Journal of Applied Econometrics*, **29**, 2, 333–343.

---

ccr81 *Program Follow Through at Primary Schools*

Description

The data set is from an US federally sponsored program for providing remedial assistance to disadvantaged primary school students. The data comprises 70 school sites.

Usage

```
data( ccr81 )
```

Format

This data frame contains the following variables (columns):

`nu` School Site Number

`y1` Total Reading Score

`y2` Total Math Score

`y3` Total Coopersmith Score

`x1` Education Level of Mother

`x2` Occupation Index

`x3` Parental Visit Index

`x4` Counseling Index

`x5` Number of Teachers

Details

The data were originally used to evaluate the efficiency of public programs and their management.

Source

A. Charnes, W. W. Cooper and E. Rhodes (1981), Evaluating Program and Managerial Efficiency: An Application of Data Envelopment Analysis to Program Follow Through, *Management Science*, **27**, 668–697.

References

Charnes, A., W. W. Cooper, and E. Rhodes. 1981. Evaluating Program and Managerial Efficiency: An Application of Data Envelopment Analysis to Program Follow Through. *Management Science* **27**: 668–697

---

coef.npsf *'coef' method for class 'npsf'*

Description

Extracts the ML parameters of a stochastic frontier model estimated by `sf`.

Usage

```
## S3 method for class 'npsf'
coef( object, ... )
```

Arguments

`object`	an object of class `npsf` returned by the function `sf`.
`...`	currently unused.

Value

`coef.npsf` returns a named vector of the ML parameters of a stochastic frontier model.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

See Also

`vcov.npsf`, `nobs.npsf`, `summary.npsf`, and `sf`.

Examples

```
require( npsf )

# Load Penn World Tables 5.6 dataset

data( pwt56 )
head( pwt56 )

# Create some missing values

pwt56 [4, "K"] <- NA

# Stochastic production frontier model with
# homoskedastic error components (half-normal)

# Use subset of observations - for year 1965

m1 <- sf(log(Y) ~ log(L) + log(K), data = pwt56,
 subset = year == 1965, distribution = "h")
coef( m1 )
```

---

halton *'halton' method for class 'npsf'*

Description

Provides Halton draws, deviates from a uniform distribution.

Usage

```
halton(n = 1, n.bases = 1, bases = NULL,
                   start = 0, random.primes = FALSE, seed = 7,
                   scale.coverage = FALSE, shuffle = FALSE)
```

Arguments

`n`	number of prime numbers to be returned (the row number in the value).
`n.bases`	numeric. number of bases used. (the column number in the value).
`bases`	numeric. Supply specific order numbers for getting primes, see `primes`. See examples.
`start`	numeric. from which value in the halton sequence to start. Default is 0, which is actually 0.
`random.primes`	logical. if `TRUE`, the `n.bases` primes are chosen on a random basis from 100008 available prime numbers. See `primes`.
`seed`	set seed for replicability. Default is 17345168.
`scale.coverage`	logical. at larger primes not whole [0,1] interval is covered. if `TRUE`, `rescale` is used to fill the coverage.
`shuffle`	logical. if `TRUE`, each column in the value is randomly reshuffled (seed is used).

Value

halton returns Halton draws.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

See Also

sf and primes.

Examples

```
require( npsf )

# obtain first 10 x 7 matrix with the first 7 primes as bases
npsf::halton(10, n.bases = 7)

# obtain first 10 x 7 matrix with the randomly chosen 7 primes as bases
npsf::halton(10, n.bases = 7, random.primes = TRUE, seed = 17345168)

# just one column with desired prime
npsf::halton(10, bases = 1)

# or 2 columns
npsf::halton(10, bases = c(1,7))

# if bases are large
npsf::halton(10, bases = c(1,7)*1000)

# the coverage is not great
npsf::halton(10, bases = c(1,7)*1000, scale.coverage = TRUE)

# reshuffle them, use seed for replicability
npsf::halton(10, bases = c(1,7)*1000, scale.coverage = TRUE, shuffle = TRUE, seed = 17345168)
```

---

mroz *Female labor force participation*

Description

Instructional dataset, N=753, cross-sectional labor force participation data Accompanying Introductory Econometrics: A Modern Approach, Jeffrey M. Wooldridge, South-Western College Publishing, (c) 2000 and Jeffrey M. Wooldridge, Econometric Analysis of Cross Section and Panel Data, MIT Press,(c) 2001.

Usage

```
data( mroz )
```

Format

This data frame contains the following variables (columns):

`inlf` =1 if in labor force, 1975

`hours` hours worked, 1975

`kidslt6` # kids < 6 years

`kidsge6` # kids 6-18

`age` woman's age in yrs

`educ` years of schooling

`wage` estimated wage from earns., hours

`repwage` reported wage at interview in 1976

`hushrs` hours worked by husband, 1975

`husage` husband's age

`huseduc` husband's years of schooling

`huswage` husband's hourly wage, 1975

`faminc` family income, 1975

`mtr` fed. marginal tax rate facing woman

`motheduc` mother's years of schooling

`fatheduc` father's years of schooling

`unem` unem. rate in county of resid.

`city` =1 if live in SMSA

`exper` actual labor mkt exper

`nwifeinc` (faminc - wage*hours)/1000

Details

Instructional dataset, N=753, cross-sectional labor force participation data Accompanying Introductory Econometrics: A Modern Approach, Jeffrey M. Wooldridge, South-Western College Publishing, (c) 2000 and Jeffrey M. Wooldridge, Econometric Analysis of Cross Section and Panel Data, MIT Press,(c) 2001.

Source

Datasets accessible from http://wooldridge.swcollege.com, http://courses.bus.msu.edu/econ/821/001/index.cfm?action=mat, and http://www.cengage.com/aise/economics/wooldridge_3e_datasets/

References

Mroz, T.A. 1987. The Sensitiviy of an Empirical Model of Married Women's Hours of Work to Economic and Statistical Assumptions. *Econometrica* **55**: 765-799

nobs.npsf *'nobs' method for class 'npsf'*

Description

Extracts the number of observations for which efficiencies are estimated by SF or DEA model estimated by sf, teradial, tenonradial, or teradialbc.

Usage

```
 ## S3 method for class 'npsf'
nobs( object, ... )
```

Arguments

object	an object of class npsf returned by the function sf).
...	currently unused.

Value

nobs.npsf returns the number of observations for which efficiencies are estimated by SF or DEA model.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

See Also

vcov.npsf, coef.npsf, summary.npsf, and sf.

Examples

```
require( npsf )

# Load Penn World Tables 5.6 dataset

data( pwt56 )
head( pwt56 )

# Create some missing values

pwt56 [4, "K"] <- NA

# Stochastic production frontier model with
# homoskedastic error components (half-normal)

# Use subset of observations - for year 1965
```

```
m1 <- sf(log(Y) ~ log(L) + log(K), data = pwt56,
 subset = year == 1965, distribution = "h")
nobs( m1 )

# DEA

t1 <- teradialbc ( Y ~ K + L, data = pwt56, subset = Nu < 10)
nobs(t1)
```

---

nptestind — *Nonparametric Test of Independence*

Description

In output based efficiency measurement, routine `nptestind` perform test that radial (Debreu-Farrell) output-based measure of technical efficiency under chosen assumption about the technology and mix of outputs are independent. In input-based efficiency measurement, routine `nptestind` perform test that radial (Debreu-Farrell) input-based measure of technical efficiency under chosen assumption about the technology and mix of inputs are independent. Testing is performed using bootstrap technique.

Usage

```
nptestind(formula, data, subset,
 rts = c("C", "NI", "V"), base = c("output", "input"),
 reps = 999, alpha = 0.05,
 print.level = 1, dots = TRUE)
```

Arguments

`formula`	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model. The details of model specification are given under 'Details'.
`data`	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment (`formula`), typically the environment from which `teradial` is called.
`subset`	an optional vector specifying a subset of observations for which technical efficiency is to be computed.
`rts`	character or numeric. string: first letter of the word "c" for constant, "n" for non-increasing, or "v" for variable returns to scale assumption. numeric: 3 for constant, 2 for non-increasing, or 1 for variable returns to scale assumption.
`base`	character or numeric. string: first letter of the word "o" for computing output-based or "i" for computing input-based technical efficiency measure. string: 2 for computing output-based or 1 for computing input-based technical efficiency measure

`reps`	specifies the number of bootstrap replications to be performed. The default is 999. The minimum is 100. Adequate estimates of confidence intervals using bias-corrected methods typically require 1,000 or more replications.
`alpha`	sets significance level; default is `alpha=0.05`.
`dots`	logical. Relevant if `print.level>=1`. If TRUE, one dot character is displayed for each successful replication; if FALSE, display of the replication dots is suppressed.
`print.level`	numeric. 0 - nothing is printed; 1 - print summary of the model and data. 2 - print summary of technical efficiency measures. 3 - print estimation results observation by observation. Default is 1.

Details

In output based efficiency measurement, routine `nptestind` perform test that radial (Debreu-Farrell) output-based measure of technical efficiency under chosen assumption about the technology and mix of outputs are independent. In input-based efficiency measurement, routine `nptestind` perform test that radial (Debreu-Farrell) input-based measure of technical efficiency under chosen assumption about the technology and mix of inputs are independent.

Testing is performed using bootstrap technique (see Wilson, 2003).

Results can be summarized using `summary.npsf`.

Value

`nptestrts` returns a list of class `npsf` containing the following elements:

`K`	numeric: number of data points.
`M`	numeric: number of outputs.
`N`	numeric: number of inputs.
`rts`	string: RTS assumption.
`base`	string: base for efficiency measurement.
`reps`	numeric: number of bootstrap replications.
`alpha`	numeric: significance level.
`t4n`	numeric: value of the T4n statistic.
`pval`	numeric: p-value of the test of independence.

Note

Results can be summarized using `summary.npsf`.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

References

Färe, R. and Lovell, C. A. K. (1978), Measuring the technical efficiency of production, *Journal of Economic Theory*, **19**, 150–162, doi: 10.1016/00220531(78)900601

Färe, R., Grosskopf, S. and Lovell, C. A. K. (1994), *Production Frontiers*, Cambridge U.K.: Cambridge University Press

Wilson P.W. (2003), Testing Independence in Models of Productive Efficiency, *Journal of Productivity Analysis*, **20**, 361–390, doi: 10.1023/A:1027355917855

See Also

`teradial`, `tenonradial`, `teradialbc`, `tenonradialbc`, `nptestrts`, `sf`

Examples

```
## Not run:

require( npsf )

# Prepare data and matrices

data( ccr81 )
head( ccr81 )

# Create some missing values

ccr81 [64, "x4"] <- NA # just to create missing
ccr81 [68, "y2"] <- NA # just to create missing

Y2 <- as.matrix( ccr81[ , c("y1", "y2", "y3"), drop = FALSE] )
X2 <- as.matrix( ccr81[ , c("x1", "x2", "x3", "x4", "x5"), drop = FALSE] )

# Perform nonparametric test that radial (Debreu-Farrell)
# output-based measure of technical efficiency under assumption of
# NIRS technology and mix of outputs are independent. Test is
# performed based on 999 replications at the 5

t1 <- nptestind ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
data = ccr81, base = "o", rts = "n",
reps = 999, dots = TRUE)


# Really large data-set

data(usmanuf)
head(usmanuf)

nrow(usmanuf)
table(usmanuf$year)

# This will take some time depending on computer power
```

```
data(usmanuf)
head(usmanuf)

t2 <- nptestind ( Y ~ K + L + M, data = usmanuf,
subset = year >= 1999 & year <= 2000,
reps = 999, dots = TRUE, base = "i", rts = "v")


## End(Not run)
```

---

nptestrts *Nonparametric Test of Returns to Scale*

Description

Routine `nptestrts` performs nonparametric tests the returns to scale of the underlying technology via bootstrapping techniques.

Usage

```
nptestrts(formula, data, subset,
 base = c("output", "input"),
 homogeneous = TRUE, test.two = TRUE,
 reps = 999, alpha = 0.05,
 core.count = 1, cl.type = c("SOCK", "MPI"),
 print.level = 1, dots = TRUE)
```

Arguments

`formula`	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model. The details of model specification are given under 'Details'.
`data`	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment (`formula`), typically the environment from which `teradial` is called.
`subset`	an optional vector specifying a subset of observations for which technical efficiency is to be computed.
`base`	character or numeric. string: first letter of the word "o" for computing output-based or "i" for computing input-based technical efficiency measure. string: 2 for computing output-based or 1 for computing input-based technical efficiency measure
`homogeneous`	logical. If TRUE, the reference set is bootstrapped with homogeneous smoothing; if FALSE, the reference set is bootstrapped with heterogeneous smoothing.
`test.two`	logical. If TRUE, test 2, where efficiency measures under assumption of non-increasing and variable returns to scale technology are compared; if FALSE, `nptestrts` stops after test 1 is completed.

`reps`	specifies the number of bootstrap replications to be performed. The default is 999. The minimum is 100. Adequate estimates of confidence intervals using bias-corrected methods typically require 1,000 or more replications.
`alpha`	sets significance level; default is `alpha=0.05`.
`core.count`	positive integer. Number of cluster nodes. If `core.count=1`, the process runs sequentially. See `performParallel` in package `snowFT` for more details.
`cl.type`	Character string that specifies cluster type (see `makeClusterFT` in package `snowFT`). Possible values are 'MPI' and 'SOCK' ('PVM' is currently not available). See `performParallel` in package `snowFT` for more details.
`dots`	logical. Relevant if `print.level>=1`. If TRUE, one dot character is displayed for each successful replication; if FALSE, display of the replication dots is suppressed.
`print.level`	numeric. 0 - nothing is printed; 1 - print summary of the model and data. 2 - print summary of technical efficiency measures. 3 - print estimation results observation by observation. Default is 1.

Details

Routine `nptestrts` performs nonparametric tests the returns to scale of the underlying technology (see Simar L. and P.W. Wilson (2002), Nonparametric Tests of Return to Scale, *European Journal of Operational Research*, **139**, 115–132, doi: 10.1016/S03772217(01)001679).

If `test.two` is not specified, `nptestrts` performs only Test #1, which consists of two parts. First, the null hypothesis that the technology is globally CRS (vs VRS) is tested. Second, the null hypothesis that the data point is scale efficient is tested.

If `test.two` is specified, `nptestrts` may perform Test #2. If the null hypothesis that the technology is CRS is rejected, `test.two` requests that `nptestrts` tests the null hypothesis that the technology is NIRS (vs VRS). If not all data points are scale efficient, `nptestrts` tests that the reason for scale inefficiency is DRS. If the null hypothesis that the technology is CRS is not rejected and all data points are scale efficient, `nptestrts` will not perform Test #2 even if `test.two` is specified.

Models for `nptestrts` are specified symbolically. A typical model has the form `outputs ~ inputs`, where `outputs` (`inputs`) is a series of (numeric) terms which specifies outputs (inputs). Refer to the examples.

If `core.count>=1`, `nptestrts` will perform bootstrap on multiple cores. Parallel computing requires package `snowFT`. By the default cluster type is defined by option `cl.type="SOCK"`. Specifying `cl.type="MPI"` requires package `Rmpi`.

On some systems, specifying option `cl.type="SOCK"` results in much quicker execution than specifying option `cl.type="MPI"`. Option `cl.type="SOCK"` might be problematic on Mac system.

Parallel computing make a difference for large data sets. Specifying option `dots=TRUE` will indicate at what speed the bootstrap actually proceeds. Specify `reps=100` and compare two runs with option `core.count=1` and `core.count>1` to see if parallel computing speeds up the bootstrap. For small samples, parallel computing may actually slow down the `nptestrts`.

Results can be summarized using `summary.npsf`.

Value

nptestrts returns a list of class npsf containing the following elements:

- **K**: numeric: number of data points.
- **M**: numeric: number of outputs.
- **N**: numeric: number of inputs.
- **rts**: string: RTS assumption.
- **base**: string: base for efficiency measurement.
- **reps**: numeric: number of bootstrap replications.
- **alpha**: numeric: significance level.
- **teCrs**: numeric: measures of technical efficiency under the assumption of CRS.
- **teNrs**: numeric: measures of technical efficiency under the assumption of NiRS.
- **teVrs**: numeric: measures of technical efficiency under the assumption of VRS.
- **sefficiency**: numeric: scale efficiency.
- **sefficiencyMean**: numeric: ratio of means of technical efficiency measures under CRS and VRS.
- **pGlobalCRS**: numeric: p-value of the test that the technology is globally CRS.
- **psefficient**: numeric: p-value of the test that data point is statistically scale efficient.
- **sefficient**: logical: returns TRUE, if statistically scale efficient; FALSE otherwise.
- **nsefficient**: numeric: number of statistically scale efficient.
- **nrsOVERvrsMean**: numeric: ratio of means of technical efficiency measures under NIRS and VRS (if test.two=TRUE).
- **pGlobalNRS**: numeric: p-value of the test the technology is globally NIRS (if test.two=TRUE).
- **sineffdrs**: logical: returns TRUE if statistically scale inefficient due to DRS and FALSE if statistically scale inefficient due to IRS (if test.two=TRUE and not all data points are statistically scale efficient nsefficient<K).
- **pineffdrs**: numeric: p-value of the test that data point is scale inefficient due to DRS (if test.two=TRUE and not all data points are statistically scale efficient nsefficient<K).
- **nrsOVERvrs**: numeric: ratio of measures of technical efficiency under NiRS and VRS (if test.two=TRUE and not all data points are statistically scale efficient nsefficient<K).
- **esample**: logical: returns TRUE if the observation in user supplied data is in the estimation subsample and FALSE otherwise.

Note

Before specifying option homogeneous it is advised to preform the test of independence (see nptestind).

Results can be summarized using summary.npsf.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>, Pavlo Mozharovskyi <pavlo.mozharovskyi@telecom-paris.fr>

References

Badunenko, O. and Mozharovskyi, P. (2016), Nonparametric Frontier Analysis using Stata, *Stata Journal*, **16**3, 550–89, doi: 10.1177/1536867X1601600302

Färe, R. and Lovell, C. A. K. (1978), Measuring the technical efficiency of production, *Journal of Economic Theory*, **19**, 150–162, doi: 10.1016/00220531(78)900601

Färe, R., Grosskopf, S. and Lovell, C. A. K. (1994), *Production Frontiers*, Cambridge U.K.: Cambridge University Press, doi: 10.1017/CBO9780511551710

Simar L. and P.W. Wilson (2002), Nonparametric Tests of Return to Scale, *European Journal of Operational Research*, **139**, 115–132, doi: 10.1016/S03772217(01)001679

See Also

teradial, tenonradial, teradialbc, tenonradialbc, nptestind, sf

Examples

```
## Not run:

require( npsf )

# Prepare data and matrices

data( ccr81 )
head( ccr81 )

# Create some missing values

ccr81 [64, "x4"] <- NA # just to create missing
ccr81 [68, "y2"] <- NA # just to create missing

Y2 <- as.matrix( ccr81[ , c("y1", "y2", "y3"), drop = FALSE] )
X2 <- as.matrix( ccr81[ , c("x1", "x2", "x3", "x4", "x5"), drop = FALSE] )

# Perform output-based test of returns to scale smoothed
# homogeneous bootstrap with 999 replications at the 5
# significance level.  Also perform Test #2

t1 <- nptestrts(y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
data = ccr81, homogeneous = TRUE,
reps = 999, dots = TRUE, base = "o")

# suppress printing replication dots
t2 <- nptestrts(Y2 ~ X2,
homogeneous = TRUE,
reps = 100, dots = FALSE, base = "o")


# heterogeneous
t3 <- nptestrts(Y2 ~ X2,
homogeneous = FALSE,
```

```
reps = 100, dots = TRUE, base = "o")


# ===========================
# ===  Parallel computing ===
# ===========================

# Perform previous test but use 8 cores and
# cluster type SOCK

t3 <-  nptestrts(y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
data = ccr81, homogeneous = FALSE,
reps = 100, dots = TRUE, base = "o",
core.count = 8, cl.type = "SOCK")


# Really large data-set

data(usmanuf)
head(usmanuf)

nrow(usmanuf)
table(usmanuf$year)

# Figure industries to include in the sample (first quarter)

summary(usmanuf[usmanuf$year >= 1999 & usmanuf$year < 2000, "naics"])

# This test is quite demanding and it will take some time
# depending on computer power

t4 <- nptestrts(Y ~ K + L + M, data = usmanuf,
subset = year >= 1999 & year < 2000 & naics < 321900,
homogeneous = FALSE, reps = 100, dots = TRUE, base = "o",
core.count = 8, cl.type = "SOCK")

# This is very computer intensive task

t5 <- nptestrts(Y ~ K + L + M, data = usmanuf,
subset = year >= 1999 & year < 2000,
homogeneous = FALSE, reps = 100, dots = TRUE, base = "o",
core.count = 8, cl.type = "SOCK")


## End(Not run)
```

---

primes — *'primes' method for class 'npsf'*

Description

Provides prime numbers.

Usage

```
primes(n = NULL, which = NULL, random.primes = FALSE, seed = 7)
```

Arguments

`n`	number of prime numbers to be returned. Should be smaller than 100008.
`which`	numeric. if specific prime numbers are requred. See examples.
`random.primes`	logical. if `n` is supplied and `random.primes = TRUE`, the `n` primes are chosen on a random basis from 100008 available prime numbers.
`seed`	set seed for replicability. Default is 17345168.

Details

`primes` just returns prime numbers, which come from `https://primes.utm.edu/lists/small/100000.txt`, see `https://primes.utm.edu`

Value

`primes` returns prime numbers.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

Source

`https://primes.utm.edu/lists/small/100000.txt` and `https://primes.utm.edu`

See Also

`sf` and `halton`.

Examples

```
require( npsf )

# obtain first 30 prime numbers
npsf::primes( 30 )

# the same as
npsf::primes( n = 30 )

# the result in both case above are 30 prime numbers

# if we use
npsf::primes( which = 30 )
```

```
# the 30th prime is returns, just a scalar

# both cannot be used
# npsf::primes(n = 30, which = 30, random.primes = FALSE, seed = 17345168)
# will give a mistake

# you can get random 30 primes, use seed for replicability
npsf::primes(n = 30, random.primes = TRUE, seed = 17345168)

# obtain specific primes: which take order number(s)
npsf::primes(which = c(3,67,30, 100008))
```

---

pwt56 *Penn World Tables 5.6 (compiled in 1995)*

Description

The data set is from Penn World Tables (PWT) 5.6. This data set provides only selected variables for years 1965 and 1990.

Usage

```
data( pwt56 )
```

Format

This data frame contains the following variables (columns):

`Nu` Order Number

`Country` Country Name

`year` 1965 or 1990

`Y` Real GDP chain, international prices of 1985

`K` Capital stock, international prices of 1985

`L` Number of workers, in thousands

Details

The Penn World Table was developed by Robert Summers and Alan Heston (and others) to facilitate consistent national accounts comparisons across countries as well as over time. The data can be used to evaluate the efficiency of economies of various countries in years 1965 and 1990.

Source

http://www.rug.nl/research/ggdc/data/pwt/pwt-5.6. These data were originally hosted on the website of the Center for International Comparisons at the University of Pennsylvania.

References

Heston, A. and Summers, R. (1991), The Penn World Table (Mark 5): An Expanded Set of International Comparisons, 1950-1988, *The Quarterly Journal of Economics*, **106**, 327–368.

`rescale` *'rescale' method for class 'npsf'*

Description

rescales a vector.

Usage

```
rescale(x, lb = min(x), ub = max(x))
```

Arguments

`x`	a numeric vector.
`lb`	numeric. lower bound.
`ub`	numeric. upper bound.

Value

`rescale` returns rescaled vector.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

See Also

`sf`, `primes`, and `halton`.

Examples

```
require( npsf )

# obtain first 30 prime numbers
set.seed(8265897)
t1 <- runif(10, min = 1, max = 2)
summary(t1)
summary(rescale(t1, 0, 10))
```

sf *Stochastic Frontier Models Using Cross-Sectional and Panel Data*

Description

sf performs maximum likelihood estimation of the parameters and technical or cost efficiencies in cross-sectional stochastic (production or cost) frontier models with half-normal or truncated normal distributional assumption imposed on inefficiency error component.

Usage

```
sf(formula, data, it = NULL, subset,
 prod = TRUE, model = "K1990", distribution = c("h"),
 eff.time.invariant = FALSE,
 mean.u.0i.zero     = FALSE,
 mean.u.0i          = NULL,
 ln.var.u.0i        = NULL,
 ln.var.v.0i        = NULL,
 ln.var.v.it        = NULL,
 simtype = c("halton", "rnorm"), halton.base = NULL, R = 500,
 simtype_GHK = c("halton", "runif"), R_GHK = 500,
 random.primes = FALSE,
 cost.eff.less.one  = FALSE, level = 95, marg.eff = FALSE,
 start.val = NULL, maxit = 199, report.ll.optim = 10,
 reltol = 1e-8, lmtol = sqrt(.Machine$double.eps),
 digits = 4, print.level = 4, seed = 17345168,
 only.data = FALSE)
```

Arguments

formula	an object of class “formula” (or one that can be coerced to that class): a symbolic description of the model. The details of model specification are given under ‘Details’.
data	an optional data frame containing variables in the model. If not found in data, the variables are taken from environment (formula), typically the environment from which sf is called.
it	vector with two character entries. E.g., c("ID", "TIME"), where "ID" defines individuals that are observed in time periods defined by "TIME". The default is NULL. At default, cross-sectional model will be estimated.
subset	an optional vector specifying a subset of observations for which technical or cost efficiencies are to be computed.
prod	logical. If TRUE, the estimates of parameters of stochastic production frontier model and of technical efficiencies are returned; if FALSE, the estimates of parameters of stochastic cost frontier model and of cost efficiencies are returned.

`model` character. Five panel data models are estimated for now. "K1990" and "K1990modified" (see Kumbhakar, 1990), "BC1992" (see Battese and Coelli, 1992), "4comp" (see Badunenko and Kumbhakar (2016) and Filippini and Greene, 2016). They specify common evolution of inefficiency. Deffault is "K1990". The time functions are "(1 + exp(b*t + c*t^2))^-1", "1 + d*(t-T_i) + e*(t-T_i)^2", and "exp(-f*(t-T_i))", respectively.

`distribution` either `"h"` (half-normal), `"t"` (truncated normal), or `"e"` (exponential, only crosssectional models), specifying the distribution of inefficiency term.

`eff.time.invariant` logical. If `TRUE`, the 1st generation of panel data models is estimated, otherwise, the 2nd generation or 4 components panel data model is estimated.

`mean.u.0i.zero` logical. If `TRUE`, normal-half normal model is estimated, otherwise, normal-truncated model is estimated.

`mean.u.0i` one-sided formula; e.g. `mean.u.0i ~ z1 + z2`. Specifies whether the mean of pre-truncated normal distribution of inefficiency term is a linear function of exogenous variables. In cross-sectional models, used only when `distribution = "t"`. If `NULL`, mean is assumed to be constant for all ids.

`ln.var.u.0i` one-sided formula; e.g. `ln.var.u.0i ~ z1 + z2`. Specifies exogenous variables entering the expression for the log of variance of inefficiency error component. If `NULL`, inefficiency term is assumed to be homoskedastic, i.e. $\sigma_u^2 = exp(\gamma[0])$. Time invariant variables are expected.

`ln.var.v.0i` one-sided formula; e.g. `ln.var.v.0i ~ z1 + z2`. Specifies exogenous variables entering the expression for variance of random noise error component. If `NULL`, random noise component is assumed to be homoskedastic, i.e. $\sigma_v^2 = exp(\gamma[0])$. Time invariant variables are expected.

`ln.var.v.it` one-sided formula; e.g. `ln.var.v.it ~ z1 + z2`. Specifies exogenous variables entering the expression for variance of random noise error component. If `NULL`, random noise component is assumed to be homoskedastic, i.e. $\sigma_v^2 = exp(\gamma[0])$. Time invariant variables are expected.

`simtype` character. Type of random deviates for the 4 components model. 'halton' draws are default. One can specify 'rnorm.'

`halton.base` numeric. The prime number which is the base for the Halton draws. If not used, different bases are used for each id.

`R` numeric. Number of draws. Default is 500. Can be time consuming.

`simtype_GHK` character. Type of random deviates for use in GHK for efficiency estimating by approximation. 'halton' draws are default. One can specify 'runif.'

`R_GHK` numeric. Number of draws for GHK. Default is 500. Can be time consuming.

`random.primes` logical. If `TRUE`, and `halton.base = NULL`, the primes are chosen on a random basis for each ID from 100008 available prime numbers.

`cost.eff.less.one` logical. If `TRUE`, and `prod = FALSE`, reported cost efficiencies are larger than one. Interpretation: by what factor is total cost larger than the potential total cost.

`level` numeric. Defines level% two-sided prediction interval for technical or cost efficiencies (see Horrace and Schmidt 1996). Default is 95.

marg.eff — logical. If TRUE, unit-specific marginal effects of exogenous variables on the mean of distribution of inefficiency term are returned.

start.val — numeric. Starting values to be supplied to the optimization routine. If NULL, OLS and method of moments estimates are used (see Kumbhakar and Lovell 2000).

maxit — numeric. Maximum number of iterations. Default is 199.

report.ll.optim — numeric. Not used for now.

reltol — numeric. One of convergence criteria. Not used for now.

lmtol — numeric. Tolerance for the scaled gradient in ML optimization. Default is sqrt(.Machine$double.eps).

digits — numeric. Number of digits to be displayed in estimation results and for efficiency estimates. Default is 4.

print.level — numeric. 1 - print estimation results. 2 - print optimization details. 3 - print summary of point estimates of technical or cost efficiencies. 7 - print unit-specific point and interval estimates of technical or cost efficiencies. Default is 4.

seed — set seed for replicability. Default is 17345168.

only.data — logical. If TRUE, only data are returned. Default is FALSE

Details

Models for sf are specified symbolically. A typical model has the form y ~ x1 + ..., where y represents the logarithm of outputs or total costs and {x1,...} is a series of inputs or outputs and input prices (in logs).

Options ln.var.u.0i and ln.var.v.0i can be used if multiplicative heteroskedasticity of either inefficiency or random noise component (or both) is assumed; i.e. if their variances can be expressed as exponential functions of (e.g. size-related) exogenous variables (including intercept) (see Caudill et al. 1995).

If marg.eff = TRUE and distribution = "h", the marginal effect of *k*th exogenous variable on the expected value of inefficiency term of unit *i* is computed as: $\gamma[k]\sigma[i]/\sqrt{2}\pi$, where $\sigma_u[i] = \sqrt{e}xp(z[i]'\gamma)$. If distribution = "t", marginal effects are returned if either mean.u.0i or ln.var.u.0i are not NULL. If the same exogenous variables are specified under both options, (non-monotonic) marginal effects are computed as explained in Wang (2002).

See references and links below.

Value

sf returns a list of class npsf containing the following elements:

coef — numeric. Named vector of ML parameter estimates.

vcov — matrix. Estimated covariance matrix of ML estimator.

ll — numeric. Value of log-likelihood at ML estimates.

efficiencies — data frame. Contains point estimates of unit-specific technical or cost efficiencies: exp(-E(u|e)) of Jondrow et al. (1982), E(exp(-u)|e) of Battese and Coelli

	(1988), and exp(-M(ule)), where M(ule) is the mode of conditional distribution of inefficiency term. In addition, estimated lower and upper bounds of (1-α)100% two-sided prediction intervals are returned.
`marg.effects`	data frame. Contains unit-specific marginal effects of exogenous variables on the expected value of inefficiency term.
`sigmas_u`	matrix. Estimated unit-specific variances of inefficiency term. Returned if `ln.var.u.0i` is not `NULL`.
`sigmas_v`	matrix. Estimated unit-specific variances of random noise component. Returned if `ln.var.v.0i` is not `NULL`.
`mu`	matrix. Estimated unit-specific means of pre-truncated normal distribution of inefficiency term. Returned if `mean.u.0i` is not `NULL`.
`esample`	logical. Returns TRUE if the observation in user supplied data is in the estimation subsample and FALSE otherwise.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

References

Badunenko, O. and Kumbhakar, S.C. (2016), When, Where and How to Estimate Persistent and Transient Efficiency in Stochastic Frontier Panel Data Models, *European Journal of Operational Research*, **255**(1), 272–287, doi: 10.1016/j.ejor.2016.04.049.

Battese, G., Coelli, T. (1988), Prediction of firm-level technical effiiencies with a generalized frontier production function and panel data. *Journal of Econometrics*, **38**, 387–399.

Battese, G., Coelli, T. (1992), Frontier production functions, technical efficiency and panel data: With application to paddy farmers in India. *Journal of Productivity Analysis*, **3**, 153–169.

Caudill, S., Ford, J., Gropper, D. (1995), Frontier estimation and firm-specific inefficiency measures in the presence of heteroscedasticity. *Journal of Business and Economic Statistics*, **13**, 105–111.

Filippini, M. and Greene, W.H. (2016), Persistent and transient productive inefficiency: A maximum simulated likelihood approach. *Journal of Productivity Analysis*, **45** (2), 187–196.

Horrace, W. and Schmidt, P. (1996), On ranking and selection from independent truncated normal distributions. *Journal of Productivity Analysis*, **7**, 257–282.

Jondrow, J., Lovell, C., Materov, I., Schmidt, P. (1982), On estimation of technical inefficiency in the stochastic frontier production function model. *Journal of Econometrics*, **19**, 233–238.

Kumbhakar, S. (1990), Production Frontiers, Panel Data, and Time-varying Technical Inefficiency. *Journal of Econometrics*, **46**, 201–211.

Kumbhakar, S. and Lovell, C. (2003), *Stochastic Frontier Analysis.* Cambridge: Cambridge University Press, doi: 10.1017/CBO9781139174411.

Wang, H.-J. (2002), Heteroskedasticity and non-monotonic efficiency effects of a stochastic frontier model. *Journal of Productivity Analysis*, **18**, 241–253.

See Also

`teradial`, `tenonradial`, `teradialbc`, `tenonradialbc`, `nptestrts`, `nptestind`, `halton`, `primes`

Examples

```
require( npsf )

# Cross-sectional examples begin ------------------------------------------

# Load Penn World Tables 5.6 dataset

data( pwt56 )
head( pwt56 )

# Create some missing values

pwt56 [4, "K"] <- NA

# Stochastic production frontier model with
# homoskedastic error components (half-normal)

# Use subset of observations - for year 1965

m1 <- sf(log(Y) ~ log(L) + log(K), data = pwt56,
         subset = year == 1965, distribution = "h")

m1 <- sf(log(Y) ~ log(L) + log(K), data = pwt56,
         subset = year == 1965, distribution = "e")

# Test CRS: 'car' package in required for that
## Not run: linearHypothesis(m1, "log(L) + log(K) = 1")

# Write efficiencies to the data frame using 'esample':

pwt56$BC[ m1$esample ] <- m1$efficiencies$BC
## Not run: View(pwt56)

# Computation using matrices

Y1 <- as.matrix(log(pwt56[pwt56$year == 1965,
                          c("Y"), drop = FALSE]))
X1 <- as.matrix(log(pwt56[pwt56$year == 1965,
                          c("K", "L"), drop = FALSE]))

X1 [51, 2] <- NA # create missing
X1 [49, 1] <- NA # create missing

m2 <- sf(Y1 ~ X1, distribution = "h")

# Load U.S. commercial banks dataset

data(banks05)
head(banks05)

# Doubly heteroskedastic stochastic cost frontier
# model (truncated normal)
```

```
# Print summaries of cost efficiencies' estimates

m3 <- sf(lnC ~ lnw1 + lnw2 + lny1 + lny2, ln.var.u.0i = ~ ER,
         ln.var.v.0i = ~ LA, data = banks05, distribution = "t",
         prod = FALSE, print.level = 3)

m3 <- sf(lnC ~ lnw1 + lnw2 + lny1 + lny2, ln.var.u.0i = ~ ER,
         ln.var.v.0i = ~ LA, data = banks05, distribution = "e",
         prod = FALSE, print.level = 3)

# Non-monotonic marginal effects of equity ratio on
# the mean of distribution of inefficiency term

m4 <- sf(lnC ~ lnw1 + lnw2 + lny1 + lny2, ln.var.u.0i = ~ ER,
         mean.u.0i = ~ ER, data = banks05, distribution = "t",
         prod = FALSE, marg.eff = TRUE)

summary(m4$marg.effects)


# Cross-sectional examples end --------------------------------------------

## Not run:

# Panel data examples begin -----------------------------------------------

# The only way to differentiate between cross-sectional and panel-data
# models is by specifying "it".
# If "it" is not specified, cross-sectional model will be estimated.
# Example is below.

# Read data ----------------------------------------------------------------

# Load U.S. commercial banks dataset

data(banks00_07)
head(banks00_07, 5)

banks00_07$trend <- banks00_07$year - min(banks00_07$year) + 1

# Model specification ------------------------------------------------------

my.prod     <- FALSE
my.it       <- c("id","year")

# my.model = "BC1992"
# my.model = "K1990modified"
# my.model = "K1990"

# translog -----------------------------------------------------------------
formu <- log(TC) ~ (log(Y1) + log(Y2) + log(W1) + log(W2) + trend)^2 +
 I(0.5*log(Y1)^2) + I(0.5*log(Y2)^2) + I(0.5*log(W1)^2) + I(0.5*log(W2)^2) +
```

```
 trend + I(0.5*trend^2)

# Cobb-Douglas ------------------------------------------------------------
# formu <- log(TC) ~ log(Y1) + log(Y2) + log(W1) + log(W2) + trend + I(trend^2)

ols <- lm(formu, data = banks00_07)
# just to mark the results of the OLS model
summary(ols)

# Components specifications ----------------------------------------------

ln.var.v.it <- ~ log(TA)
ln.var.u.0i <- ~ ER_ave
mean.u.0i_1 <- ~ LLP_ave + LA_ave
mean.u.0i_2 <- ~ LLP_ave + LA_ave - 1

# Suppose "it" is not specified
# Then it is a cross-sectional model

t0_1st_0 <- sf(formu, data = banks00_07, subset = year > 2000,
               prod = my.prod,
               ln.var.v.it = ln.var.v.it,
               ln.var.u.0i = ln.var.u.0i,
               eff.time.invariant = TRUE,
               mean.u.0i.zero = TRUE)

# 1st generation models ---------------------------------------------------

# normal-half normal ------------------------------------------------------

# the same as above but "it" is specified

t0_1st_0 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
               prod = my.prod,
               ln.var.v.it = ln.var.v.it,
               ln.var.u.0i = ln.var.u.0i,
               eff.time.invariant = TRUE,
               mean.u.0i.zero = TRUE)

# Note efficiencies are time-invariant

# confidence intervals for efficiencies -----------------------------------

head(t0_1st_0$efficiencies, 20)


# normal-truncated normal -------------------------------------------------

# truncation point is constant (for all ids) ------------------------------

t0_1st_1 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
               prod = my.prod,
               eff.time.invariant = TRUE,
```

```
               mean.u.0i.zero = FALSE,
               ln.var.v.it = ln.var.v.it,
               ln.var.u.0i = ln.var.u.0i,
               mean.u.0i = NULL,
               cost.eff.less.one = TRUE)



# truncation point is determined by variables -----------------------------

t0_1st_2 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
               prod = my.prod,
               eff.time.invariant = TRUE,
               mean.u.0i.zero = FALSE,
               mean.u.0i = mean.u.0i_1,
               ln.var.v.it = ln.var.v.it,
               ln.var.u.0i = ln.var.u.0i,
               cost.eff.less.one = TRUE)



# the same, but without intercept in mean.u.0i

t0_1st_3 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
               prod = my.prod,
               eff.time.invariant = TRUE,
               mean.u.0i.zero = FALSE,
               mean.u.0i = mean.u.0i_2,
               ln.var.v.it = ln.var.v.it,
               ln.var.u.0i = ln.var.u.0i,
               cost.eff.less.one = TRUE)

# 2nd generation models ---------------------------------------------------

# normal-half normal ------------------------------------------------------

# Kumbhakar (1990) model --------------------------------------------------

t_nhn_K1990 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
                  prod = my.prod,
                  eff.time.invariant = FALSE,
                  mean.u.0i.zero = TRUE,
                  ln.var.v.it = ln.var.v.it,
                  ln.var.u.0i = ln.var.u.0i,
                  cost.eff.less.one = FALSE)


# Kumbhakar (1990) modified model -----------------------------------------

t_nhn_K1990modified <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
                          prod = my.prod, model = "K1990modified",
                          eff.time.invariant = FALSE,
                          mean.u.0i.zero = TRUE,
```

```
                          ln.var.v.it = ln.var.v.it,
                          ln.var.u.0i = ln.var.u.0i,
                          cost.eff.less.one = FALSE)


# Battese and Coelli (1992) model -----------------------------------------

t_nhn_BC1992 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
                   prod = my.prod, model = "BC1992",
                   eff.time.invariant = FALSE,
                   mean.u.0i.zero = TRUE,
                   ln.var.v.it = ln.var.v.it,
                   ln.var.u.0i = ln.var.u.0i,
                   cost.eff.less.one = FALSE)

# normal-truncated normal -------------------------------------------------

# Kumbhakar (1990) model --------------------------------------------------

# truncation point is constant (for all ids) ------------------------------

t_ntn_K1990_0 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
                    prod = my.prod,
                    eff.time.invariant = FALSE,
                    mean.u.0i.zero = FALSE,
                    ln.var.v.it = ln.var.v.it,
                    ln.var.u.0i = ln.var.u.0i,
                    cost.eff.less.one = FALSE)


# truncation point is determined by variables -----------------------------

t_ntn_K1990_1 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
                    prod = my.prod,
                    eff.time.invariant = FALSE,
                    mean.u.0i.zero = FALSE,
                    mean.u.0i = mean.u.0i_1,
                    ln.var.v.it = ln.var.v.it,
                    ln.var.u.0i = ln.var.u.0i,
                    cost.eff.less.one = FALSE)

# Efficiencies are tiny, since empirically truncation points are quite big.
# Try withouth an intercept in conditional mean f-n

t_ntn_K1990_2 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
                    prod = my.prod,
                    eff.time.invariant = FALSE,
                    mean.u.0i.zero = FALSE,
                    mean.u.0i = mean.u.0i_2,
                    ln.var.v.it = ln.var.v.it,
                    ln.var.u.0i = ln.var.u.0i,
                    cost.eff.less.one = FALSE)
```

```
# Kumbhakar (1990) modified model -----------------------------------------

t_ntn_K1990modified <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
                          prod = my.prod, model = "K1990modified",
                          eff.time.invariant = FALSE,
                          mean.u.0i.zero = FALSE,
                          mean.u.0i = mean.u.0i_1,
                          ln.var.v.it = ln.var.v.it,
                          ln.var.u.0i = ln.var.u.0i,
                          cost.eff.less.one = FALSE)

# Battese and Coelli (1992) model -----------------------------------------


t_ntn_BC1992 <- sf(formu, data = banks00_07, it = my.it, subset = year > 2000,
                   prod = my.prod, model = "BC1992",
                   eff.time.invariant = FALSE,
                   mean.u.0i.zero = FALSE,
                   mean.u.0i = mean.u.0i_1,
                   ln.var.v.it = ln.var.v.it,
                   ln.var.u.0i = ln.var.u.0i,
                   cost.eff.less.one = FALSE)

# The next one (without "subset = year > 2000" option) converges OK

t_ntn_BC1992 <- sf(formu, data = banks00_07, it = my.it,
                   prod = my.prod, model = "BC1992",
                   eff.time.invariant = FALSE,
                   mean.u.0i.zero = FALSE,
                   mean.u.0i = mean.u.0i_1,
                   ln.var.v.it = ln.var.v.it,
                   ln.var.u.0i = ln.var.u.0i,
                   cost.eff.less.one = FALSE)

# 4 component model ------------------------------------------------------

# Note, R should better be more than 200, this is just for illustration.
# This is the model that takes long to be estimated.
# For the following example, 'mlmaximize' required 357 iterations and
# took 8 minutes.
# The time will increase with more data and more parameters.

formu <- log(TC) ~ log(Y1) + log(Y2) + log(W1) + log(W2) + trend

t_4comp <- sf(formu, data = banks00_07, it = my.it,
              subset = year >= 2001 & year < 2006,
              prod = my.prod, model = "4comp",
              R = 500, initialize.halton = TRUE,
              lmtol = 1e-5, maxit = 500, print.level = 4)

# With R = 500, 'mlmaximize' required 124 iterations and
# took 7 minutes.
# The time will increase with more data and more parameters.
```

```
formu <- log(TC) ~ log(Y1) + log(Y2) + log(W1) + log(W2) + trend

t_4comp_500 <- sf(formu, data = banks00_07, it = my.it,
              subset = year >= 2001 & year < 2006,
              prod = my.prod, model = "4comp",
              R = 500, initialize.halton = TRUE,
              lmtol = 1e-5, maxit = 500, print.level = 4)

# @e_i0, @e_it, and @e_over give efficiencies,
# where @ is either 'c' or 't' for cost or production function.
# e.g., t_ntn_4comp$ce_i0 from last model, gives persistent cost efficiencies

# Panel data examples end ------------------------------------------------


## End(Not run)
```

---

summary.npsf *'summary' method for class 'npsf'*

Description

Prints summary of SF or DEA model estimated by `sf`, `teradial`, `tenonradial`, and `teradialbc`, or testing procedures `nptestrts` and `nptestind`.

Usage

```
## S3 method for class 'npsf'
summary( object, ... )
## S3 method for class 'summary.npsf'
print( x, digits = NULL, print.level = NULL, ... )
```

Arguments

`object`	an object of class `npsf` returned by one of the functions `sf`, `teradial`, `tenonradial`, `teradialbc`, `nptestrts` or `nptestind`.
`x`	an object of class `npsf` returned by one of the functions `sf`, `teradial`, `tenonradial`, `teradialbc`, `nptestrts` or `nptestind`.
`digits`	numeric. Number of digits to be displayed in estimation results and for efficiency estimates. Default is 4.
`print.level`	numeric. 0 - nothing is printed; 1 - print summary of the model and data. 2 - print summary of technical efficiency measures. 3 - print estimation results observation by observation (for DEA models). Default is 1.
`...`	currently unused.

Details

The summary depends on the model or testing procedure that is being estimated

Value

Currently no value is returned

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

See Also

sf, teradial, tenonradial, teradialbc, tenonradialbc, nptestrts, and nptestind

Examples

```
require( npsf )

# Load Penn World Tables 5.6 dataset

data( pwt56 )

# Stochastic production frontier model with
# homoskedastic error components (half-normal)

# Use subset of observations - for year 1965

# DEA

t1 <- teradialbc ( Y ~ K + L, data = pwt56, subset = Nu < 10,
 reps = 199, print.level = 0)
summary(t1)

# SFA

m1 <- sf(log(Y) ~ log(L) + log(K), data = pwt56,
 subset = year == 1965, distribution = "h",
 print.level = 0)
summary( m1 )

# Load U.S. commercial banks dataset

data(banks05)

m3 <- sf(lnC ~ lnw1 + lnw2 + lny1 + lny2, ln.var.u.0i = ~ ER,
         ln.var.v.0i = ~ LA, data = banks05, distribution = "t",
         prod = FALSE, print.level = 3)

summary(m3)
```

tenonradial *Nonradial Measure of Technical Efficiency, the Russell Measure*

Description

Routine tenonradial uses reduced linear programming to compute the nonradial output- or input-based measure of technical efficiency, which is known as the Russell measure. In input-based nonradial efficiency measurement, this measure allows for non-proportional/different reductions in each positive input, and this is what permits it to shrink an input vector all the way back to the efficient subset. In output-based nonradial efficiency measurement, the Russell measure allows for non-proportional/different expansions of each positive output.

Usage

```
tenonradial(formula, data, subset,
 rts = c("C", "NI", "V"),
 base = c("output", "input"),
 ref = NULL, data.ref = NULL, subset.ref = NULL,
 full.solution = TRUE,
 print.level = 1)
```

Arguments

formula	an object of class “formula” (or one that can be coerced to that class): a symbolic description of the model. The details of model specification are given under ‘Details’.
data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment (formula), typically the environment from which tenonradial is called.
subset	an optional vector specifying a subset of observations for which technical efficiency is to be computed.
rts	character or numeric. string: first letter of the word “c” for constant, “n” for non-increasing, or “v” for variable returns to scale assumption. numeric: 3 for constant, 2 for non-increasing, or 1 for variable returns to scale assumption.
base	character or numeric. string: first letter of the word “o” for computing output-based or “i” for computing input-based technical efficiency measure. string: 2 for computing output-based or 1 for computing input-based technical efficiency measure
ref	an object of class “formula” (or one that can be coerced to that class): a symbolic description of inputs and outputs that are used to define the technology reference set. The details of technology reference set specification are given under ‘Details’. If reference is not provided, the technical efficiency measures for data points are computed relative to technology based on data points themselves.
data.ref	an optional data frame containing the variables in the technology reference set. If not found in data.ref, the variables are taken from environment(ref), typically the environment from which tenonradial is called.

`subset.ref`	an optional vector specifying a subset of observations to define the technology reference set.
`full.solution`	logical. The detailed solution is returned. See `value` section.
`print.level`	numeric. 0 - nothing is printed; 1 - print summary of the model and data. 2 - print summary of technical efficiency measures. 3 - print estimation results observation by observation. Default is 1.

Details

Routine `tenonradial` computes the nonradial output- or input-based measure of technical efficiency under assumption of constant, non-increasing, or variable returns to scale technology. The details of the estimator can be found e.g., in Färe, Grosskopf, and Lovell (1994) or Badunenko and Mozharovskyi (2020).

Models for `tenonradial` are specified symbolically. A typical model has the form `outputs ~ inputs`, where `outputs` (`inputs`) is a series of (numeric) terms which specifies outputs (inputs). The same goes for reference set. Refer to the examples.

Results can be summarized using `summary.npsf`.

Value

`tenonradial` returns a list of class `npsf` containing the following elements:

`model`	string: model name.
`K`	numeric: number of data points for which efficiency is estimated.
`M`	numeric: number of outputs.
`N`	numeric: number of inputs.
`Kref`	numeric: number of data points in the reference.
`rts`	string: RTS assumption.
`base`	string: base for efficiency measurement.
`te`	numeric: nonradial measure (Russell) of technical efficiency.
`te.detail`	numeric: `K` x `ncol` matrix containing thetas or lambdas for `ncol` outputs (output-based) or inputs (input-based). `ncol` is `M` for output- and `N` for input-based efficiency measurement.
`intensity`	numeric: `K` x `Kref` matrix containing the intensity variables `z`. These can be used to identify peers.
`esample`	logical: returns TRUE if the observation in user supplied data is in the estimation subsample and FALSE otherwise.
`esample.ref`	logical: returns TRUE if the observation in the user supplied reference is in the reference subsample and FALSE otherwise.

Note

In case of one input (output), the input (output)-based Russell measure is equal to Debrue-Farrell (`teradial`) measure of technical efficiency.

Results can be summarized using `summary.npsf`.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>, Pavlo Mozharovskyi <pavlo.mozharovskyi@telecom-paris.fr>

References

Badunenko, O. and Mozharovskyi, P. (2016), Nonparametric Frontier Analysis using Stata, *Stata Journal*, **16**3, 550–89, doi: 10.1177/1536867X1601600302

Badunenko, O. and Mozharovskyi, P. (2020), Statistical inference for the Russell measure of technical efficiency, *Journal of the Operational Research Society*, **71**3, 517–527, doi: 10.1080/01605682.2019.1599778

Färe, R. and Lovell, C. A. K. (1978), Measuring the technical efficiency of production, *Journal of Economic Theory*, **19**, 150–162, doi: 10.1016/00220531(78)900601

Färe, R., Grosskopf, S. and Lovell, C. A. K. (1994), *Production Frontiers*, Cambridge U.K.: Cambridge University Press, doi: 10.1017/CBO9780511551710

See Also

`teradial`, `teradialbc`, `tenonradialbc`, `nptestrts`, `nptestind`, `sf`, `summary.npsf` for printing summary results

Examples

```
require( npsf )

# Prepare data and matrices

data( pwt56 )
head( pwt56 )

# Create some missing values

pwt56 [49, "K"] <- NA # create missing

Y1 <- as.matrix ( pwt56[ pwt56$year == 1965, c("Y"), drop = FALSE] )
X1 <- as.matrix ( pwt56[ pwt56$year == 1965, c("K", "L"), drop = FALSE] )

X1 [51, 2] <- NA # create missing
X1 [49, 1] <- NA # create missing

data( ccr81 )
head( ccr81 )

# Create some missing values

ccr81 [64, "x4"] <- NA # create missing
ccr81 [68, "y2"] <- NA # create missing

Y2 <- as.matrix( ccr81[ , c("y1", "y2", "y3"), drop = FALSE] )
X2 <- as.matrix( ccr81[ , c("x1", "x2", "x3", "x4", "x5"), drop = FALSE] )
```

```
# Computing without reference set

# Using formula

# Country is a categorical variable, so nonradial gives error message

# t1 <- tenonradial ( Country ~ K + L, data = pwt56 )

# for computing the efficiencies of countries in 1965
# with technology reference set is defined by observations in 1965
# (that same sample of countries)

t2 <- tenonradial ( Y ~ K + L, data = pwt56, rts = "v",
base = "in", print.level = 2)

# Using a subset

t3 <- tenonradial ( Y ~ K + L, data = pwt56, subset = year == 1965,
rts = "VRS", base = "in", print.level = 3 )

t4 <- tenonradial ( Y ~ K + L, data = pwt56, subset = Nu < 10,
rts = "vrs", base = "I" )

t5 <- tenonradial ( Y ~ L, data = pwt56, subset = Nu < 10, rts = "v" )

# Multiple outputs

t8 <- tenonradial ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5, data = ccr81,
rts = "v", base = "i" )

# Using a subset

t7 <- tenonradial ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5, data = ccr81,
subset = x5 != 22, rts = "n", base = "o" )

# Computation using matrices

t9 <- tenonradial ( Y1 ~ X1, rts = "v", base = "i" )

# Define subsets on a fly

t10 <- tenonradial ( Y1[-1,] ~ X1[-2,1] )
t11 <- tenonradial ( Y1[-3,] ~ X1[-1,], rts = "v", base = "o" )

# Multiple outputs

t12 <- tenonradial ( Y2 ~ X2 )
t13 <- tenonradial ( Y2[-66,] ~ X2[-1, -c(1,3)] )


# Computing with reference set

# Using formula
```

```
# For computing the efficiencies of countries with order number
# less than 10 with technology reference set defined by countries
# with order number larger than 10 and smaller than 11 (in effect
# no reference set, hence warning) type

t14 <- tenonradial ( Y ~ K + L, data = pwt56, subset = Nu < 10,
ref = Y ~ K + L, data.ref = pwt56,
subset.ref = Nu > 10 & Nu < 11 ) # warning

# For computing the efficiencies of countries with order number
# less than 10 with technology reference set defined by countries
# with order number larger than 10 and smaller than 15 type

t15 <- tenonradial ( Y ~ K + L, data = pwt56, subset = Nu < 10, ref = Y ~ K + L,
data.ref = pwt56, subset.ref = Nu > 10 & Nu < 15 )

# For computing the efficiencies of countries in 1965
# with technology reference set is defined by observations in both
# 1965 and 1990 (all) type

t16 <- tenonradial ( Y ~ K + L, data = pwt56, subset = year == 1965,
rts = "v", base = "i",
ref = Y ~ K + L, data.ref = pwt56 )

# For computing the efficiencies of countries in 1990
# with technology reference set is defined by observations in 1965
# type

t17 <- tenonradial ( Y ~ K + L, data = pwt56, subset = year == 1990,
ref = Y ~ K + L, data.ref = pwt56, subset.ref = year == 1965 )

# Using matrices

t18 <- tenonradial ( Y1[-1,] ~ X1[-2,], ref = Y1[-2,] ~ X1[-1,] )

# error: not equal number of observations in outputs and inputs

# t19 <- tenonradial ( Y1[-1,] ~ X1[-(1:2),],
# ref = Y1[-2,] ~ X1[-1,1] )

# Combined formula and matrix

# error: not equal number of inputs in data and reference set

# t20 <- tenonradial ( Y ~ K + L, data = pwt56, subset = Nu < 10,
# ref = Y1[-2,] ~ X1[-1,1] )

t21 <- tenonradial ( Y ~ K + L, data = pwt56, subset = Nu < 10,
ref = Y1[-2,] ~ X1[-1,] )

## Not run:
```

```
# Really large data-set

data(usmanuf)
head(usmanuf)

nrow(usmanuf)
table(usmanuf$year)

# This will take some time depending on computer power

t22 <- tenonradial ( Y ~ K + L + M, data = usmanuf,
subset = year >= 1995 & year <= 2000 )

# Summary

summary ( t22$te )

# Write efficiencies to the data frame:

usmanuf$te_nonrad_crs_out[ t22$esample ] <- t22$te

head(usmanuf, 17)


## End(Not run)
```

`tenonradialbc` *Statistical Inference Regarding the Russell Measure of Technical Efficiency*

Description

Routine `tenonradialbc` performs bias correction of the nonradial Russell input- or output-based measure of technical efficiency, computes bias and constructs confidence intervals via bootstrapping techniques.

Usage

```
tenonradialbc(formula, data, subset,
          ref = NULL, data.ref = NULL, subset.ref = NULL,
          rts = c("C", "NI", "V"), base = c("output", "input"),
          homogeneous = TRUE, smoothed = TRUE, kappa = NULL,
          reps = 999, level = 95,
          print.level = 1, show.progress = TRUE, seed = NULL)
```

Arguments

`formula`	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model. The details of model specification are given under 'Details'.
`data`	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment (`formula`), typically the environment from which `teradial` is called.
`subset`	an optional vector specifying a subset of observations for which technical efficiency is to be computed.
`rts`	character or numeric. string: first letter of the word "c" for constant, "n" for non-increasing, or "v" for variable returns to scale assumption. numeric: 3 for constant, 2 for non-increasing, or 1 for variable returns to scale assumption.
`base`	character or numeric. string: first letter of the word "o" for computing output-based or "i" for computing input-based technical efficiency measure. string: 2 for computing output-based or 1 for computing input-based technical efficiency measure
`ref`	an object of class "formula" (or one that can be coerced to that class): a symbolic description of inputs and outputs that are used to define the technology reference set. The details of technology reference set specification are given under 'Details'. If reference is not provided, the technical efficiency measures for data points are computed relative to technology based on data points themselves.
`data.ref`	an optional data frame containing the variables in the technology reference set. If not found in `data.ref`, the variables are taken from environment(ref), typically the environment from which `teradial` is called.
`subset.ref`	an optional vector specifying a subset of observations to define the technology reference set.
`smoothed`	logical. If TRUE, the reference set is bootstrapped with smoothing; if FALSE, the reference set is bootstrapped with subsampling.
`homogeneous`	logical. Relevant if `smoothed=TRUE`. If TRUE, the reference set is bootstrapped with homogeneous smoothing; if FALSE, the reference set is bootstrapped with heterogeneous subsampling.
`kappa`	relevant if `smoothed=TRUE`. 'kappa' sets the size of the subsample as K^kappa, where K is the number of data points in the original reference set. The default value is 0.7. 'kappa' may be between 0.5 and 1.
`reps`	specifies the number of bootstrap replications to be performed. The default is 999. The minimum is 100. Adequate estimates of confidence intervals using bias-corrected methods typically require 1,000 or more replications.
`level`	sets confidence level for confidence intervals; default is `level=95`.
`show.progress`	logical. Relevant if `print.level>=1`. If TRUE, progress of the bootstrap is displayed; if FALSE, display of the bootstrap progress is suppressed.
`print.level`	numeric. 0 - nothing is printed; 1 - print summary of the model and data. 2 - print summary of technical efficiency measures. 3 - print estimation results observation by observation. Default is 1.
`seed`	numeric. The seed (for replication purposes).

Details

Routine `tenonradialbc` performs bias correction of the nonradial Russell input- or output-based measure of technical efficiency, computes bias and constructs confidence intervals via bootstrapping techniques (see Badunenko and Mozharovskyi (2020), doi: 10.1080/01605682.2019.1599778).

Models for `tenonradialbc` are specified symbolically. A typical model has the form `outputs ~ inputs`, where `outputs` (`inputs`) is a series of (numeric) terms which specifies outputs (inputs). The same goes for reference set. Refer to the examples.

Results can be summarized using `summary.npsf`.

Value

`tenonradialbc` returns a list of class `npsf` containing the following elements:

`K`	numeric: number of data points.
`M`	numeric: number of outputs.
`N`	numeric: number of inputs.
`rts`	string: RTS assumption.
`base`	string: base for efficiency measurement.
`reps`	numeric: number of bootstrap replications.
`level`	numeric: confidence level for confidence intervals.
`te`	numeric: radial measure (Russell) of technical efficiency.
`tebc`	numeric: bias-corrected radial measures of technical efficiency.
`biasboot`	numeric: bootstrap bias estimate for the original radial measures of technical efficiency.
`varboot`	numeric: bootstrap variance estimate for the radial measures of technical efficiency.
`biassqvar`	numeric: one-third of the ratio of bias squared to variance for radial measures of technical efficiency.
`realreps`	numeric: actual number of replications used for statistical inference.
`telow`	numeric: lower bound estimate for radial measures of technical efficiency.
`teupp`	numeric: upper bound estimate for radial measures of technical efficiency.
`teboot`	numeric: `reps` x `K` matrix containing bootstrapped measures of technical efficiency from each of `reps` bootstrap replications.
`esample`	logical: returns TRUE if the observation in user supplied data is in the estimation subsample and FALSE otherwise.

Note

Before specifying option `homogeneous` it is advised to preform the test of independence (see `nptestind`). Routine `nptestrts` may help deciding regarding option `rts`.

Results can be summarized using `summary.npsf`.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>, Pavlo Mozharovskyi <pavlo.mozharovskyi@telecom-paris.fr>

References

Badunenko, O. and Mozharovskyi, P. (2020), Statistical inference for the Russell measure of technical efficiency, *Journal of the Operational Research Society*, **71**3, 517–527, doi: 10.1080/01605682.2019.1599778

Färe, R., Grosskopf, S. and Lovell, C. A. K. (1994), *Production Frontiers*, Cambridge U.K.: Cambridge University Press, doi: 10.1017/CBO9780511551710

See Also

teradial, tenonradial, teradialbc, nptestrts, nptestind, sf

Examples

```
 ## Not run:

  data( ccr81 )
  head( ccr81 )

  # Subsampling

  t9 <- tenonradialbc(y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5, data = ccr81,
                          ref = NULL, data.ref = NULL, subset.ref = NULL,
                          rts = "v", base = "i",
                          homogeneous = FALSE, smoothed = TRUE, kappa = .6,
                          reps = 999, level = 95,
                          print.level = 1, show.progress = TRUE, seed = NULL)
  # display the results

  cbind(te = t9$te, telow = t9$telow, tebc = t9$tebc, teupp = t9$teupp,
        biasboot = t9$biasboot, varboot = t9$varboot, biassqvar = t9$biassqvar)

  # Smoothing

  t10 <- tenonradialbc(y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5, data = ccr81,
                          ref = NULL, data.ref = NULL, subset.ref = NULL,
                          rts = "v", base = "i",
                          homogeneous = TRUE, smoothed = TRUE, kappa = .6,
                          reps = 999, level = 95,
                          print.level = 1, show.progress = TRUE, seed = NULL)
  # display the results

  cbind(te = t10$te, telow = t10$telow, tebc = t10$tebc, teupp = t10$teupp,
        biasboot = t10$biasboot, varboot = t10$varboot, biassqvar = t10$biassqvar)


## End(Not run)
```

`teradial` *Radial Measure of Technical Efficiency, the Debrue-Farrell Measure*

Description

Routine `teradial` computes radial Debrue-Farrell input- or output-based measure of efficiency via reduced linear programing. In input-based radial efficiency measurement, this measure allows for proportional reductions in each positive input, and this is what permits it to shrink an input vector all the way back to the efficient subset. In output-based radial efficiency measurement, the Debrue-Farrell measure allows for proportional expansions of each positive output.

Usage

```
teradial(formula, data, subset,
 rts = c("C", "NI", "V"),
 base = c("output", "input"),
 ref = NULL, data.ref = NULL, subset.ref = NULL,
 intensity = FALSE,
 print.level = 1)
```

Arguments

`formula`	an object of class “formula” (or one that can be coerced to that class): a symbolic description of the model. The details of model specification are given under ‘Details’.
`data`	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment (`formula`), typically the environment from which `teradial` is called.
`subset`	an optional vector specifying a subset of observations for which technical efficiency is to be computed.
`rts`	character or numeric. string: first letter of the word “c” for constant, “n” for non-increasing, or “v” for variable returns to scale assumption. numeric: 3 for constant, 2 for non-increasing, or 1 for variable returns to scale assumption.
`base`	character or numeric. string: first letter of the word “o” for computing output-based or “i” for computing input-based technical efficiency measure. string: 2 for computing output-based or 1 for computing input-based technical efficiency measure
`ref`	an object of class “formula” (or one that can be coerced to that class): a symbolic description of inputs and outputs that are used to define the technology reference set. The details of technology reference set specification are given under ‘Details’. If reference is not provided, the technical efficiency measures for data points are computed relative to technology based on data points themselves.
`data.ref`	an optional data frame containing the variables in the technology reference set. If not found in `data.ref`, the variables are taken from environment(ref), typically the environment from which `teradial` is called.

`subset.ref` an optional vector specifying a subset of observations to define the technology reference set.

`intensity` logical. If set to TRUE, the value `intensity` will contain `K` x `Kref` matrix with intensity variables, which can be used to for example identify the peers. Default is `FALSE` as the matris is potentially large.

`print.level` numeric. 0 - nothing is printed; 1 - print summary of the model and data. 2 - print summary of technical efficiency measures. 3 - print estimation results observation by observation. Default is 1.

Details

Routine `teradial` computes the radial output- or input-based measure of technical efficiency under assumption of constant, non-increasing, or variable returns to scale technology. The details of the estimator can be found e.g., in Färe, Grosskopf, and Lovell (1994, especially section 3.1 on p.62 fot input-based and section 4.1 on p.96 for output-based efficiency measurement) or Badunenko and Mozharovskyi (2016).

Models for `teradial` are specified symbolically. A typical model has the form `outputs ~ inputs`, where `outputs` (`inputs`) is a series of (numeric) terms which specifies outputs (inputs). The same goes for reference set. Refer to the examples.

Results can be summarized using `summary.npsf`.

Value

`teradial` returns a list of class `npsf` containing the following elements:

`K` numeric: number of data points.

`M` numeric: number of outputs.

`N` numeric: number of inputs.

`rts` string: RTS assumption.

`base` string: base for efficiency measurement.

`te` numeric: radial measure (Debrue-Farrell) of technical efficiency.

`intensity` numeric: if the option `intensity` is set to TRUE, the value `intensity` will contain `K` x `Kref` matrix with intensity variables, which can be used to for example identify the peers (see example with `t3` in the example section). Is `NULL` if option `intensity` is set to `FALSE`, which is a default.

`esample` logical: returns TRUE if the observation in user supplied data is in the estimation subsample and FALSE otherwise.

`esample.ref` logical: returns TRUE if the observation in the user supplied reference is in the reference subsample and FALSE otherwise.

Note

In case of one input (output), the input (output)-based Debrue-Farrell measure is equal to Russell measure of technical efficiency (see `tenonradial`).

Results can be summarized using `summary.npsf`.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>, Pavlo Mozharovskyi <pavlo.mozharovskyi@telecom-paris.fr>

References

Badunenko, O. and Mozharovskyi, P. (2016), Nonparametric Frontier Analysis using Stata, *Stata Journal*, **16**3, 550–89, doi: 10.1177/1536867X1601600302

Färe, R. and Lovell, C. A. K. (1978), Measuring the technical efficiency of production, *Journal of Economic Theory*, **19**, 150–162, doi: 10.1016/00220531(78)900601

Färe, R., Grosskopf, S. and Lovell, C. A. K. (1994), *Production Frontiers*, Cambridge U.K.: Cambridge University Press, doi: 10.1017/CBO9780511551710

See Also

`tenonradial`, `teradialbc`, `tenonradialbc`, `nptestrts`, `nptestind`, `sf`

Examples

```
require( npsf )

# Prepare data and matrices

data( pwt56 )
head( pwt56 )

# Create some missing values

pwt56 [49, "K"] <- NA # create missing

Y1 <- as.matrix ( pwt56[ pwt56$year == 1965, c("Y"), drop = FALSE] )
X1 <- as.matrix ( pwt56[ pwt56$year == 1965, c("K", "L"), drop = FALSE] )

X1 [51, 2] <- NA # create missing
X1 [49, 1] <- NA # create missing

data( ccr81 )
head( ccr81 )

# Create some missing values

ccr81 [64, "x4"] <- NA # create missing
ccr81 [68, "y2"] <- NA # create missing

Y2 <- as.matrix( ccr81[ , c("y1", "y2", "y3"), drop = FALSE] )
X2 <- as.matrix( ccr81[ , c("x1", "x2", "x3", "x4", "x5"), drop = FALSE] )

# Computing without reference set

# Using formula
```

```
# Country is a categorical variable, so nonradial gives error message

# t1 <- teradial ( Country ~ K + L, data = pwt56 )

# for computing the efficiencies of countries in 1965
# with technology reference set is defined by observations in 1965
# (that same sample of countries)

t2 <- teradial ( Y ~ K + L, data = pwt56, rts = "v",
base = "in", print.level = 2)

# Using a subset

t3 <- teradial ( Y ~ K + L, data = pwt56, subset = year == 1965,
rts = "VRS", base = "in", print.level = 3, intensity = TRUE )

# VRS constraint is satisfied, which is easy to varify
# by checking the sums of intensity variables
rowSums(t3$intensity)

# to obtain peers create a list that will contain order numers of peers
t3.peers <- list()
# now fill this list
for(i in seq.int(sum(t3$esample))){
  t3.peers[[i]] <- which( t3$intensity[i,] != 0 )
}

t4 <- teradial ( Y ~ K + L, data = pwt56, subset = Nu < 10,
rts = "vrs", base = "I" )

t5 <- teradial ( Y ~ L, data = pwt56, subset = Nu < 10, rts = "v" )

# Multiple outputs

t8 <- teradial ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5, data = ccr81,
rts = "v", base = "i" )

# Using a subset

t7 <- teradial ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5, data = ccr81,
subset = x5 != 22, rts = "n", base = "o" )

# Computation using matrices

t9 <- teradial ( Y1 ~ X1, rts = "v", base = "i" )

# Define subsets on a fly

t10 <- teradial ( Y1[-1,] ~ X1[-2,1] )
t11 <- teradial ( Y1[-3,] ~ X1[-1,], rts = "v", base = "o" )

# Multiple outputs
```

```
t12 <- teradial ( Y2 ~ X2 )
t13 <- teradial ( Y2[-66,] ~ X2[-1, -c(1,3)] )


# Computing with reference set

# Using formula

# For computing the efficiencies of countries with order number
# less than 10 with technology reference set defined by countries
# with order number larger than 10 and smaller than 11 (in effect
# no reference set, hence warning) type

t14 <- teradial ( Y ~ K + L, data = pwt56, subset = Nu < 10,
ref = Y ~ K + L, data.ref = pwt56,
subset.ref = Nu > 10 & Nu < 11 ) # warning

# For computing the efficiencies of countries with order number
# less than 10 with technology reference set defined by countries
# with order number larger than 10 and smaller than 15 type

t15 <- teradial ( Y ~ K + L, data = pwt56, subset = Nu < 10, ref = Y ~ K + L,
data.ref = pwt56, subset.ref = Nu > 10 & Nu < 15 )

# For computing the efficiencies of countries in 1965
# with technology reference set is defined by observations in both
# 1965 and 1990 (all) type

t16 <- teradial ( Y ~ K + L, data = pwt56, subset = year == 1965,
rts = "v", base = "i",
ref = Y ~ K + L, data.ref = pwt56 )

# For computing the efficiencies of countries in 1990
# with technology reference set is defined by observations in 1965
# type

t17 <- teradial ( Y ~ K + L, data = pwt56, subset = year == 1990,
ref = Y ~ K + L, data.ref = pwt56, subset.ref = year == 1965 )

# Using matrices

t18 <- teradial ( Y1[-1,] ~ X1[-2,], ref = Y1[-2,] ~ X1[-1,] )

# error: not equal number of observations in outputs and inputs

# t19 <- teradial ( Y1[-1,] ~ X1[-(1:2),],
# ref = Y1[-2,] ~ X1[-1,1] )

# Combined formula and matrix

# error: not equal number of inputs in data and reference set

# t20 <- teradial ( Y ~ K + L, data = pwt56, subset = Nu < 10,
```

```
# ref = Y1[-2,] ~ X1[-1,1] )

t21 <- teradial ( Y ~ K + L, data = pwt56, subset = Nu < 10,
ref = Y1[-2,] ~ X1[-1,] )

## Not run:

# Really large data-set

data(usmanuf)
head(usmanuf)

nrow(usmanuf)
table(usmanuf$year)

# This will take some time depending on computer power

t22 <- teradial ( Y ~ K + L + M, data = usmanuf,
subset = year >= 1995 & year <= 2000 )

# Summary

summary ( t22$te )

# Write efficiencies to the data frame:

usmanuf$te_nonrad_crs_out[ t22$esample ] <- t22$te

head(usmanuf, 17)


## End(Not run)
```

`teradialbc` *Statistical Inference Regarding the Radial Measure of Technical Efficiency*

Description

Routine `teradialbc` performs bias correction of the radial Debrue-Farrell input- or output-based measure of technical efficiency, computes bias and constructs confidence intervals via bootstrapping techniques.

Usage

```
teradialbc(formula, data, subset,
 ref = NULL, data.ref = NULL, subset.ref = NULL,
 rts = c("C", "NI", "V"), base = c("output", "input"),
 homogeneous = TRUE, smoothed = TRUE, kappa = NULL,
```

```
reps = 999, level = 95,
core.count = 1, cl.type = c("SOCK", "MPI"),
print.level = 1, dots = TRUE)
```

Arguments

`formula`	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model. The details of model specification are given under 'Details'.
`data`	an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment (`formula`), typically the environment from which `teradial` is called.
`subset`	an optional vector specifying a subset of observations for which technical efficiency is to be computed.
`rts`	character or numeric. string: first letter of the word "c" for constant, "n" for non-increasing, or "v" for variable returns to scale assumption. numeric: 3 for constant, 2 for non-increasing, or 1 for variable returns to scale assumption.
`base`	character or numeric. string: first letter of the word "o" for computing output-based or "i" for computing input-based technical efficiency measure. string: 2 for computing output-based or 1 for computing input-based technical efficiency measure
`ref`	an object of class "formula" (or one that can be coerced to that class): a symbolic description of inputs and outputs that are used to define the technology reference set. The details of technology reference set specification are given under 'Details'. If reference is not provided, the technical efficiency measures for data points are computed relative to technology based on data points themselves.
`data.ref`	an optional data frame containing the variables in the technology reference set. If not found in `data.ref`, the variables are taken from environment(ref), typically the environment from which `teradial` is called.
`subset.ref`	an optional vector specifying a subset of observations to define the technology reference set.
`smoothed`	logical. If TRUE, the reference set is bootstrapped with smoothing; if FALSE, the reference set is bootstrapped with subsampling.
`homogeneous`	logical. Relevant if `smoothed=TRUE`. If TRUE, the reference set is bootstrapped with homogeneous smoothing; if FALSE, the reference set is bootstrapped with heterogeneous smoothing.
`kappa`	relevant if `smoothed=TRUE`. 'kappa' sets the size of the subsample as K^kappa, where K is the number of data points in the original reference set. The default value is 0.7. 'kappa' may be between 0.5 and 1.
`reps`	specifies the number of bootstrap replications to be performed. The default is 999. The minimum is 100. Adequate estimates of confidence intervals using bias-corrected methods typically require 1,000 or more replications.
`level`	sets confidence level for confidence intervals; default is `level = 95`.
`core.count`	positive integer. Number of cluster nodes. If `core.count=1`, the process runs sequentially. See `performParallel` in package `snowFT` for more details.

cl.type
: Character string that specifies cluster type (see makeClusterFT in package snowFT). Possible values are 'MPI' and 'SOCK' ('PVM' is currently not available). See performParallel in package snowFT for more details.

dots
: logical. Relevant if print.level>=1. If TRUE, one dot character is displayed for each successful replication; if FALSE, display of the replication dots is suppressed.

print.level
: numeric. 0 - nothing is printed; 1 - print summary of the model and data. 2 - print summary of technical efficiency measures. 3 - print estimation results observation by observation. Default is 1.

Details

Routine teradialbc performs bias correction of the radial Debrue-Farrell input- or output-based measure of technical efficiency, computes bias and constructs confidence intervals via bootstrapping techniques. See Simar and Wilson (1998) doi: 10.1287/mnsc.44.1.49, Simar and Wilson (2000) doi: 10.1080/02664760050081951, Kneip, Simar, and Wilson (2008) doi: 10.1017/S0266466608080651, and references with links below.

Models for teradialbc are specified symbolically. A typical model has the form outputs ~ inputs, where outputs (inputs) is a series of (numeric) terms which specifies outputs (inputs). The same goes for reference set. Refer to the examples.

If core.count>=1, teradialbc will perform bootstrap on multiple cores. Parallel computing requires package snowFT. By the default cluster type is defined by option cl.type="SOCK". Specifying cl.type="MPI" requires package Rmpi.

On some systems, specifying option cl.type="SOCK" results in much quicker execution than specifying option cl.type="MPI". Option cl.type="SOCK" might be problematic on Mac system.

Parallel computing make a difference for large data sets. Specifying option dots=TRUE will indicate at what speed the bootstrap actually proceeds. Specify reps=100 and compare two runs with option core.count=1 and core.count>1 to see if parallel computing speeds up the bootstrap. For small samples, parallel computing may actually slow down the teradialbc.

Results can be summarized using summary.npsf.

Value

teradialbc returns a list of class npsf containing the following elements:

K
: numeric: number of data points.

M
: numeric: number of outputs.

N
: numeric: number of inputs.

rts
: string: RTS assumption.

base
: string: base for efficiency measurement.

reps
: numeric: number of bootstrap replications.

level
: numeric: confidence level for confidence intervals.

te
: numeric: radial measure (Russell) of technical efficiency.

tebc
: numeric: bias-corrected radial measures of technical efficiency.

biasboot	numeric: bootstrap bias estimate for the original radial measures of technical efficiency.
varboot	numeric: bootstrap variance estimate for the radial measures of technical efficiency.
biassqvar	numeric: one-third of the ratio of bias squared to variance for radial measures of technical efficiency.
realreps	numeric: actual number of replications used for statistical inference.
telow	numeric: lower bound estimate for radial measures of technical efficiency.
teupp	numeric: upper bound estimate for radial measures of technical efficiency.
teboot	numeric: reps x K matrix containing bootstrapped measures of technical efficiency from each of reps bootstrap replications.
esample	logical: returns TRUE if the observation in user supplied data is in the estimation subsample and FALSE otherwise.

Note

Before specifying option homogeneous it is advised to preform the test of independence (see nptestind). Routine nptestrts may help deciding regarding option rts.

Results can be summarized using summary.npsf.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>, Pavlo Mozharovskyi <pavlo.mozharovskyi@telecom-paris.fr>

References

Badunenko, O. and Mozharovskyi, P. (2016), Nonparametric Frontier Analysis using Stata, *Stata Journal*, **16**3, 550–89, doi: 10.1177/1536867X1601600302

Färe, R. and Lovell, C. A. K. (1978), Measuring the technical efficiency of production, *Journal of Economic Theory*, **19**, 150–162, doi: 10.1016/00220531(78)900601

Färe, R., Grosskopf, S. and Lovell, C. A. K. (1994), *Production Frontiers*, Cambridge U.K.: Cambridge University Press, doi: 10.1017/CBO9780511551710

Kneip, A., Simar L., and P.W. Wilson (2008), Asymptotics and Consistent Bootstraps for DEA Estimators in Nonparametric Frontier Models, *Econometric Theory*, **24**, 1663–1697, doi: 10.1017/S0266466608080651

Simar, L. and P.W. Wilson (1998), Sensitivity Analysis of Efficiency Scores: How to Bootstrap in Nonparametric Frontier Models, *Management Science*, **44**, 49–61, doi: 10.1287/mnsc.44.1.49

Simar, L. and P.W. Wilson (2000), A General Methodology for Bootstrapping in Nonparametric Frontier Models, *Journal of Applied Statistics*, **27**, 779–802, doi: 10.1080/02664760050081951

See Also

teradial, tenonradial, tenonradialbc, nptestrts, nptestind, sf

Examples

```
## Not run:

require( npsf )

# Prepare data and matrices

data( pwt56 )
head( pwt56 )

# Create some missing values

pwt56 [49, "K"] <- NA # just to create missing

Y1 <- as.matrix ( pwt56[ pwt56$year == 1965, c("Y"), drop = FALSE] )
X1 <- as.matrix ( pwt56[ pwt56$year == 1965, c("K", "L"), drop = FALSE] )

X1 [51, 2] <- NA # just to create missing
X1 [49, 1] <- NA # just to create missing

data( ccr81 )
head( ccr81 )

# Create some missing values

ccr81 [64, "x4"] <- NA # just to create missing
ccr81 [68, "y2"] <- NA # just to create missing

Y2 <- as.matrix( ccr81[ , c("y1", "y2", "y3"), drop = FALSE] )
X2 <- as.matrix( ccr81[ , c("x1", "x2", "x3", "x4", "x5"), drop = FALSE] )

# Compute output-based measures of technical efficiency under
# the assumption of CRS (the default) and perform bias-correctiion
# using smoothed homogeneous bootstrap (the default) with 999
# replications (the default).

t1 <- teradialbc ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
data = ccr81)

# or just

t2 <- teradialbc ( Y2 ~ X2)

# Combined formula and matrix

t3 <- teradialbc ( Y ~ K + L, data = pwt56, subset = Nu < 10,
ref = Y1[-2,] ~ X1[-1,] )

# Compute input-based measures of technical efficiency under
# the assumption of VRS and perform bias-correctiion using
# subsampling heterogenous bootstrap with 1999 replications.
# Choose to report 99
```

```
# formed by data points where x5 is not equal 10.
# Suppress printing dots.

t4 <- teradialbc ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
data = ccr81, ref = y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
subset.ref = x5 != 10, data.ref = ccr81, reps = 1999,
smoothed = FALSE, kappa = 0.7, dots = FALSE,
base = "i", rts = "v", level = 99)

# Compute input-based measures of technical efficiency under
# the assumption of NRS and perform bias-correctiion using
# smoothed heterogenous bootstrap with 499 replications for
# all data points. The reference set formed by data points
# where x5 is not equal 10.

t5 <- teradialbc ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
data = ccr81, ref = y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
subset.ref = x5 != 10, data.ref = ccr81, homogeneous = FALSE,
reps = 999, smoothed = TRUE, dots = TRUE, base = "i", rts = "n")


# ==========================
# ===  Parallel computing ===
# ==========================

# Perform previous bias-correction but use 8 cores and
# cluster type SOCK

t51 <-  teradialbc ( y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
data = ccr81, ref = y1 + y2 + y3 ~ x1 + x2 + x3 + x4 + x5,
subset.ref = x5 != 10, data.ref = ccr81, homogeneous = FALSE,
reps = 999, smoothed = TRUE, dots = TRUE, base = "i", rts = "n",
core.count = 8, cl.type = "SOCK")


# Really large data-set

data(usmanuf)
head(usmanuf)

nrow(usmanuf)
table(usmanuf$year)

# This will take some time depending on computer power

data(usmanuf)
head(usmanuf)

t6 <- teradialbc ( Y ~ K + L + M, data = usmanuf,
subset = year >= 1999 & year <= 2000, homogeneous = FALSE,
base = "o", reps = 100,
core.count = 8, cl.type = "SOCK")
```

```
## End(Not run)
```

---

truncreg *Parametric truncated regression for cross-sectional data*

Description

truncreg performs maximum likelihood estimation of the parameters in cross-sectional truncated regression.

Usage

```
truncreg(formula, data, subset,
 ll = -Inf, ul = Inf,
 lmtol = .Machine$double.eps, maxiter = 150,
 marg.eff = FALSE,
 print.level = 1)
```

Arguments

formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model. The details of model specification are given under 'Details'.
data	an optional data frame containing variables in the model. If not found in data, the variables are taken from environment (formula), typically the environment from which sf is called.
subset	an optional vector specifying a subset of observations for which technical or cost efficiencies are to be computed.
ll	scalar or one-sided formula for lower limit for left-truncation; e.g. ll = ~ 1 or ll = ~ z1.
ul	scalar or one-sided formula for upper limit for right-truncation; e.g. ul = ~ 800 or ul = ~ z1.
lmtol	numeric. Tolerance for the scaled gradient in ML optimization. Default is .Machine$double.eps.
maxiter	numeric. maximum number of iteration for maximization of the log likelihood function.
marg.eff	logical. If TRUE, unit-specific marginal effects of exogenous variables on the mean of distribution of inefficiency term are returned.
print.level	numeric. 0 - nothing is printed. 1 - optimization steps and print estimation results. 2 - detailed optimization steps and print estimation results. Default is 1.

Details

truncreg performs a regression from a sample drawn from a restricted part of the population. Under the assumption that the error term of the whole population is normal, the error terms in the truncated regression model have a truncated normal distribution.

Both lower limit for left-truncation and upper limit for right-truncation can be specified simultaneously.

Models for truncreg are specified symbolically. A typical model has the form y ~ x1 + ..., where y represents the left hand side variable and {x1,...} right hand side variables.

If marg.eff = TRUE, the marginal effects are computed.

Value

truncreg returns a list of class npsf containing the following elements:

call	call. 'truncreg.cs'.
model	character. Unevaluated call to function truncreg.
coef	numeric. Named vector of ML parameter estimates.
table	matrix. Table with results.
vcov	matrix. Estimated covariance matrix of ML estimator.
ll	numeric. Value of log-likelihood at ML estimates.
lmtol	numeric. Convergence criterion: tolerance for the scaled gradient.
LM	numeric. Value of the scaled gradient.
esttime	numeric. Estimation time.
marg.effects	data frame. Contains unit-specific marginal effects of exogenous variables.
sigma	numeric. estimate of sigma.
LL	numeric. The lower limit for left-truncation
UL	numeric. The upper limit for left-truncation
n	numeric. Number of observations (used in regression).
n.full	numeric. Number of observations (used and not used in regression).
nontruncsample	logical. Returns TRUE if the observation in user supplied data is in the estimation subsample and in non-truncated part of the sample, and FALSE otherwise.
esample	logical. Returns TRUE if the observation in user supplied data is in the estimation subsample and FALSE otherwise.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

See Also

teradial, tenonradial, teradialbc, tenonradialbc, nptestrts, nptestind, sf

Examples

```
require( npsf )

# Female labor force participation dataset

data(mroz)
head(mroz)

t1 <- npsf::truncreg(hours ~ kidslt6 + kidsge6 + age*educ,
 data = mroz, ll = 0, lmtol = 1e-16, print.level = 2)

# matrices also can be used
myY <- mroz$hours
myX <- cbind(mroz$kidslt6, mroz$kidsge6, mroz$age, mroz$educ, mroz$age*mroz$educ)

t1.m <- truncreg(myY ~ myX, ll = 0)

# gives identical result to `t1':
# compare summary(t1) and summary(t1.m)

# using variable for limits is admissible
# we obtain the same result as before

mroz$myll <- 0
t11 <- npsf::truncreg(hours ~ kidslt6 + kidsge6 + age*educ,
 data = mroz, ll = ~ myll, lmtol = 1e-16, print.level = 0)
summary(t11)

# if you believe that the sample is additionally truncted from above at say 3500

t12 <- update(t1, ul = 3500, print.level = 1)

# for obtaining marginal effects

t13 <- update(t12, marg.eff = TRUE)

summary(t13$marg.effects)
```

---

usmanuf — *US Manufacturing Industry Data*

Description

This data come from the National Bureau of Economic Research Center for Economic Studies manufacturing industry database. This data set provides only selected variables.

Usage

```
data( usmanuf )
```

Format

This data frame contains the following variables (columns):

`naics` NAICS 6-digit Codes

`year` Year ranges from 1990 to 2009

`Y` Total value of shipments in $1m divided by the deflator for VSHIP 1997=1.000 (vship/piship)

`K` Total real capital stock in $1m (cap)

`L` Total employment in thousands (emp)

`M` Total cost of materials in $1m divided by the deflator for MATCOST 1997=1.000 plus oost of electric & fuels in $1m divided by the deflator for ENERGY 1997=1.000 (matcost/pimat + energy/pien)

Details

These data come from the National Bureau of Economic Research Center for Economic Studies manufacturing industry database, which was compiled by Randy A. Becker and Wayne B. Gray. This database is a joint effort between the National Bureau of Economic Research (NBER) and U.S. Census Bureau's Center for Economic Studies (CES), containing annual industry-level data from 1958-2009 on output, employment, payroll and other input costs, investment, capital stocks, TFP, and various industry-specific price indexes. Because of the change from SIC to NAICS industry definitions in 1997, the database is provided in two versions: one with 459 four-digit 1987 SIC industries and the other with 473 six-digit 1997 NAICS industries.

Source

`https://www.nber.org/nberces/`.

References

Bartelsman, E.J. and Gray, W. (1996), The NBER Manufacturing Productivity Database, *National Bureau of Economic Research*, Technical Working Paper Series, doi: 10.3386/t0205

`vcov.npsf` *'vcov' method for class 'npsf'*

Description

Extracts the variance-covariance matrix of the ML parameters of a stochastic frontier model estimated by `sf`.

Usage

```
 ## S3 method for class 'npsf'
vcov( object, ... )
```

Arguments

object	an object of class npsf returned by the function sf.
...	currently unused.

Details

The estimated variance-covariance matrix of the ML parameters is the inverse of the negative Hessian evaluated at the MLE.

Value

vcov.npsf returns the estimated variance-covariance matrix of the ML parameters.

Author(s)

Oleg Badunenko <oleg.badunenko@brunel.ac.uk>

See Also

coef.npsf, nobs.npsf, summary.npsf, and sf.

Examples

```
require( npsf )

# Load Penn World Tables 5.6 dataset

data( pwt56 )
head( pwt56 )

# Create some missing values

pwt56 [4, "K"] <- NA

# Stochastic production frontier model with
# homoskedastic error components (half-normal)

# Use subset of observations - for year 1965

m1 <- sf(log(Y) ~ log(L) + log(K), data = pwt56,
 subset = year == 1965, distribution = "h")
vcov( m1 )
```

Index