Package 'sfaR'

July 4, 2023

Title Stochastic Frontier Analysis Routines

Version 1.0.0

Description Maximum likelihood estimation for stochastic frontier analysis (SFA) of production (profit) and cost functions. The package includes the basic stochastic frontier for cross-sectional or pooled data with several distributions for the one-sided error term (i.e., Rayleigh, gamma, Weibull, lognormal, uniform, generalized exponential and truncated skewed Laplace), the latent class stochastic frontier model (LCM) as described in Dakpo et al. (2021) <doi:10.1111/1477-9552.12422>, for cross-sectional and pooled data, and the sample selection model as described in Greene (2010) <doi:10.1007/s11123-009-0159-1>, and applied in Dakpo et al. (2021) <doi:10.1111/agec.12683>. Several possibilities in terms of optimization algorithms are proposed.

License GPL (>= 3)

URL https://github.com/hdakpo/sfaR

BugReports https://github.com/hdakpo/sfaR/issues

Depends R (>= 3.5.0)

Imports cubature, fastGHQuad, Formula, marqLevAlg, maxLik, methods, mnorm, nleqslv, plm, qrng, randtoolbox, sandwich, stats, texreg, trustOptim, ucminf

Suggests lmtest

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.2.3

NeedsCompilation no

Author K Hervé Dakpo [aut, cre],
Yann Desjeux [aut],
Arne Henningsen [aut],
Laure Latruffe [aut]

Maintainer K Hervé Dakpo <k-herve.dakpo@inrae.fr>

Repository CRAN

Date/Publication 2023-07-04 11:30:02 UTC

R topics documented:

`sfaR-package` *sfaR: A package for estimating stochastic frontier models*

Description

The **sfaR** package provides a set of tools (maximum likelihood - ML and maximum simulated likelihood - MSL) for various specifications of stochastic frontier analysis (SFA).

Details

Three categories of functions are available: `sfacross`, `sfalcmcross`, `sfaselectioncross`, which estimate different types of frontiers and offer eleven alternative optimization algorithms (i.e., "bfgs", "bhhh", "nr", "nm", "cg", "sann", "ucminf", "mla", "sr1", "sparse", "nlminb").

sfacross

sfacross estimates the basic stochastic frontier analysis (SFA) for cross-sectional or pooled data and allows for ten different distributions for the one-sided error term. These distributions include the exponential, the gamma, the generalized exponential, the half normal, the lognormal, the truncated normal, the truncated skewed Laplace, the Rayleigh, the uniform, and the Weibull distributions. In the case of the gamma, lognormal, and Weibull distributions, maximum simulated likelihood (MSL) is used with the possibility of four specific distributions to construct the draws: halton, generalized halton, sobol and uniform. Heteroscedasticity in both error terms can be implemented, in addition to heterogeneity in the truncated mean parameter in the case of the truncated normal and lognormal distributions. In addition, in the case of the truncated normal distribution, the scaling property can be estimated.

sfalcmcross

sfalcmcross estimates latent class stochastic frontier models (LCM) for cross-sectional or pooled data. It accounts for technological heterogeneity by splitting the observations into a maximum number of five classes. The classification operates based on a logit functional form that can be specified using some covariates (namely, the separating variables allowing the separation of observations in several classes). Only the half normal distribution is available for the one-sided error term. Heteroscedasticity in both error terms is possible. The choice of the number of classes can be guided by several information criteria (i.e., AIC, BIC, or HQIC).

sfaselectioncross

sfaselectioncross estimates the frontier for cross-sectional or pooled data in the presence of sample selection. The model solves the selection bias due to the correlation between the two-sided error terms in both the selection and the frontier equations. The likelihood can be estimated using five different possibilities: gauss-kronrod quadrature, adaptive integration over hypercubes (hcubature and pcubature), gauss-hermite quadrature, and maximum simulated likelihood. Only the half normal distribution is available for the one-sided error term. Heteroscedasticity in both error terms is possible.

Bugreport

Any bug or suggestion can be reported using the sfaR tracker facilities at: https://github.com/hdakpo/sfaR/issues

Author(s)

K Hervé Dakpo, Yann Desjeux, Arne Henningsen and Laure Latruffe

---

coef *Extract coefficients of stochastic frontier models*

Description

From an object of class `'summary.sfacross'`, `'summary.sfalcmcross'`, or `'summary.sfaselectioncross'`, `coef` extracts the coefficients, their standard errors, z-values, and (asymptotic) P-values.

From on object of class `'sfacross'`, `'sfalcmcross'`, or `'sfaselectioncross'`, it extracts only the estimated coefficients.

Usage

```
## S3 method for class 'sfacross'
coef(object, extraPar = FALSE, ...)

## S3 method for class 'summary.sfacross'
coef(object, ...)

## S3 method for class 'sfalcmcross'
coef(object, extraPar = FALSE, ...)

## S3 method for class 'summary.sfalcmcross'
coef(object, ...)

## S3 method for class 'sfaselectioncross'
coef(object, extraPar = FALSE, ...)

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

Arguments

`object`	A stochastic frontier model returned by `sfacross`, `sfalcmcross`, or `sfaselectioncross`, or an object of class `'summary.sfacross'`, `'summary.sfalcmcross'`, or `'summary.sfaselectioncross'`.

`extraPar`	Logical (default = `FALSE`). If `TRUE`, additional parameters are returned:
	`sigmaSq` = `sigmauSq` + `sigmavSq`
	`lambdaSq` = `sigmauSq`/`sigmavSq`
	`sigmauSq` = $\exp(Wu) = \exp(\boldsymbol{\delta}'\mathbf{Z}_u)$
	`sigmavSq` = $\exp(Wv) = \exp(\boldsymbol{\phi}'\mathbf{Z}_v)$
	`sigma` = `sigmaSq`^0.5
	`lambda` = `lambdaSq`^0.5
	`sigmau` = `sigmauSq`^0.5
	`sigmav` = `sigmavSq`^0.5
	`gamma` = `sigmauSq`/(`sigmauSq` + `sigmavSq`)

`...`	Currently ignored.

Value

For objects of class 'summary.sfacross', 'summary.sfalcmcross', or 'summary.sfaselectioncross', coef returns a matrix with four columns. Namely, the estimated coefficients, their standard errors, z-values, and (asymptotic) P-values.

For objects of class 'sfacross', 'sfalcmcross', or 'sfaselectioncross', coef returns a numeric vector of the estimated coefficients. If extraPar = TRUE, additional parameters, detailed in the section 'Arguments', are also returned. In the case of object of class 'sfalcmcross', each additional parameter ends with '#' that represents the class number.

See Also

sfacross, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfalcmcross, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfaselectioncross for sample selection in stochastic frontier model fitting function using cross-sectional or pooled data.

Examples

```
## Not run:
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog SFA (cost function) truncated normal with scaling property
tl_u_ts <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu, uhet = ~ regu, data = utility, S = -1,
scaling = TRUE, method = 'mla')
coef(tl_u_ts, extraPar = TRUE)
coef(summary(tl_u_ts))

## End(Not run)
```

---

dairynorway *Data on Norwegian dairy farms*

Description

This dataset contains nine years (1998-2006) of information on Norwegian dairy farms.

Format

A data frame with 2,727 observations on the following 23 variables.

farmid Farm identification.

year Year identification.

y1 Milk sold (1000 liters).

y2 Meat (1000 NOK).

y3 Support payments (1000 NOK).

y4 Other outputs (1000 NOK).

p1 Milk price (NOK/liter).

p2 Meat price (cattle index).

p3 Support payments price (CP index).

p4 Other outputs price index.

x1 Land (decare (daa) = 0.1 ha).

x2 Labour (1000 hours).

x3 Purchase feed (1000 NOK).

x4 Other variable costs (1000 NOK).

x5 Cattle capital (1000 NOK).

x6 Other capital (1000 NOK).

w1 Land price (NOK/daa).

w2 Labour price (NOK/hour).

w3 Feed price index.

w4 Other variable cost index.

w5 Cattle capital rent.

w6 Other capital rent and depreciation.

tc Total cost.

Source

https://sites.google.com/site/sfbook2014/home/for-stata-v12-v13-v14

References

Kumbhakar, S.C., H.J. Wang, and A. Horncastle. 2014. *A Practitioner's Guide to Stochastic Frontier Analysis Using Stata*. Cambridge University Press.

Examples

```
str(dairynorway)
summary(dairynorway)
```

dairyspain *Data on Spanish dairy farms*

Description

This dataset contains six years of observations on 247 dairy farms in northern Spain, drawn from 1993-1998. The original data consist in the farm and year identifications, plus measurements on one output (i.e. milk), and four inputs (i.e. cows, land, labor and feed).

Format

A data frame with 1,482 observations on the following 29 variables.

FARM Farm identification.

AGEF Age of the farmer.

YEAR Year identification.

COWS Number of milking cows.

LAND Agricultural area.

MILK Milk production.

LABOR Labor.

FEED Feed.

YIT Log of `MILK`.

X1 Log of `COWS`.

X2 Log of `LAND`.

X3 Log of `LABOR`.

X4 Log of `FEED`.

X11 1/2 * `X1^2`.

X22 1/2 * `X2^2`.

X33 1/2 * `X3^2`.

X44 1/2 * `X4^2`.

X12 `X1 * X2`.

X13 `X1 * X3`.

X14 `X1 * X4`.

X23 `X2 * X3`.

X24 `X2 * X4`.

X34 `X3 * X4`.

YEAR93 Dummy for `YEAR = 1993`.

YEAR94 Dummy for `YEAR = 1994`.

YEAR95 Dummy for `YEAR = 1995`.

YEAR96 Dummy for `YEAR = 1996`.

YEAR97 Dummy for `YEAR = 1997`.

YEAR98 Dummy for `YEAR = 1998`.

Details

This dataset has been used in Alvarez *et al.* (2004). The data have been normalized so that the logs of the inputs sum to zero over the 1,482 observations.

Source

http://pages.stern.nyu.edu/~wgreene/Econometrics/oldPanelDataSets.htm

References

Alvarez, A., C. Arias, and W. Greene. 2004. Accounting for unobservables in production models: management and inefficiency. *Econometric Society*, **341**:1–20.

Examples

```
str(dairyspain)
summary(dairyspain)
```

`efficiencies` *Compute conditional (in-)efficiency estimates of stochastic frontier models*

Description

`efficiencies` returns (in-)efficiency estimates of models estimated with `sfacross`, `sfalcmcross`, or `sfaselectioncross`.

Usage

```
## S3 method for class 'sfacross'
efficiencies(object, level = 0.95, newData = NULL, ...)

## S3 method for class 'sfalcmcross'
efficiencies(object, level = 0.95, newData = NULL, ...)

## S3 method for class 'sfaselectioncross'
efficiencies(object, level = 0.95, newData = NULL, ...)
```

Arguments

`object`	A stochastic frontier model returned by `sfacross`, `sfalcmcross`, or `sfaselectioncross`.
`level`	A number between between 0 and 0.9999 used for the computation of (in-)efficiency confidence intervals (defaut = `0.95`). Only used when `udist = 'hnormal'`, `'exponential'`, `'tnormal'` or `'uniform'` in `sfacross`.
`newData`	Optional data frame that is used to calculate the efficiency estimates. If NULL (the default), the efficiency estimates are calculated for the observations that were used in the estimation. In the case of object of class `sfaselectioncross`
`...`	Currently ignored.

Details

In general, the conditional inefficiency is obtained following Jondrow *et al.* (1982) and the conditional efficiency is computed following Battese and Coelli (1988). In some cases the conditional mode is also returned (Jondrow *et al.* 1982). The confidence interval is computed following Horrace and Schmidt (1996), Hjalmarsson *et al.* (1996), or Berra and Sharma (1999) (see 'Value' section).

In the case of the half normal distribution for the one-sided error term, the formulae are as follows (for notations, see the 'Details' section of `sfacross` or `sfalcmcross`):

- The conditional inefficiency is:

$$E\left[u_i|\epsilon_i\right] = \mu_{i*} + \sigma_* \frac{\phi\left(\frac{\mu_{i*}}{\sigma_*}\right)}{\Phi\left(\frac{\mu_{i*}}{\sigma_*}\right)}$$

where

$$\mu_{i*} = \frac{-S\epsilon_i\sigma_u^2}{\sigma_u^2 + \sigma_v^2}$$

and

$$\sigma_*^2 = \frac{\sigma_u^2\sigma_v^2}{\sigma_u^2 + \sigma_v^2}$$

- The Battese and Coelli (1988) conditional efficiency is obtained with:

$$E\left[\exp\left(-u_i\right)|\epsilon_i\right] = \exp\left(-\mu_{i*} + \frac{1}{2}\sigma_*^2\right)\frac{\Phi\left(\frac{\mu_{i*}}{\sigma_*} - \sigma_*\right)}{\Phi\left(\frac{\mu_{i*}}{\sigma_*}\right)}$$

- The reciprocal of the Battese and Coelli (1988) conditional efficiency is obtained with:

$$E\left[\exp\left(u_i\right)|\epsilon_i\right] = \exp\left(\mu_{i*} + \frac{1}{2}\sigma_*^2\right)\frac{\Phi\left(\frac{\mu_{i*}}{\sigma_*} + \sigma_*\right)}{\Phi\left(\frac{\mu_{i*}}{\sigma_*}\right)}$$

- The conditional mode is computed using:

$$M\left[u_i|\epsilon_i\right] = \mu_{i*} \quad \text{For} \quad \mu_{i*} > 0$$

and

$$M\left[u_i|\epsilon_i\right] = 0 \quad \text{For} \quad \mu_{i*} \leq 0$$

- The confidence intervals are obtained with:

$$\mu_{i*} + I_L\sigma_* \leq E\left[u_i|\epsilon_i\right] \leq \mu_{i*} + I_U\sigma_*$$

with $LB_i = \mu_{i*} + I_L\sigma_*$ and $UB_i = \mu_{i*} + I_U\sigma_*$

and

$$I_L = \Phi^{-1}\left\{1 - \left(1 - \frac{\alpha}{2}\right)\left[1 - \Phi\left(-\frac{\mu_{i*}}{\sigma_*}\right)\right]\right\}$$

and

$$I_U = \Phi^{-1}\left\{1 - \frac{\alpha}{2}\left[1 - \Phi\left(-\frac{\mu_{i*}}{\sigma_*}\right)\right]\right\}$$

Thus

$$\exp\left(-UB_i\right) \leq E\left[\exp\left(-u_i\right)|\epsilon_i\right] \leq \exp\left(-LB_i\right)$$

In the case of the sample selection, as underlined in Greene (2010), the conditional inefficiency could be computed using Jondrow *et al.* (1982). However, here the conditionanl (in)efficiency is obtained using the properties of the closed skew-normal (CSN) distribution (Lai, 2015). The conditional efficiency can be obtained using the moment generating functions of a CSN distribution (see Gonzalez-Farias *et al.* (2004)). We have:

$$E\left[\exp\left(tu_i\right)|\epsilon_i\right] = M_{u|\epsilon}(t) = \frac{\Phi_2\left(\tilde{\mathbf{D}}\tilde{\mathbf{\Sigma}}t; \tilde{\boldsymbol{\kappa}}, \tilde{\mathbf{\Delta}} + \tilde{\mathbf{D}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{D}}'\right)}{\Phi_2\left(\mathbf{0}; \tilde{\boldsymbol{\kappa}}, \tilde{\mathbf{\Delta}} + \tilde{\mathbf{D}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{D}}'\right)} \exp\left(t\tilde{\boldsymbol{\pi}} + \frac{1}{2}t^2\tilde{\mathbf{\Sigma}}\right)$$

where $\tilde{\boldsymbol{\pi}} = \frac{-S\epsilon_i\sigma_u^2}{\sigma_v^2+\sigma_u^2}$, $\tilde{\mathbf{\Sigma}} = \frac{\sigma_v^2\sigma_u^2}{\sigma_v^2+\sigma_u^2}$, $\tilde{\mathbf{D}} = \begin{pmatrix}\frac{S\rho}{\sigma_v}\\ 1\end{pmatrix}$, $\tilde{\boldsymbol{\kappa}} = \begin{pmatrix}-\mathbf{Z}'_{si}\boldsymbol{\gamma} - \frac{\rho\sigma_v\epsilon_i}{\sigma_v^2+\sigma_u^2}\\ \frac{S\sigma_u^2\epsilon_i}{\sigma_v^2+\sigma_u^2}\end{pmatrix}$, $\tilde{\mathbf{\Delta}} = \begin{pmatrix}1-\rho^2 & 0\\ 0 & 0\end{pmatrix}$.

The derivation of the efficiency and the reciprocal efficiency is obtained by replacing $t = -1$ and $t = 1$, respectively. To obtain the inefficiency as $E\left[u_i|\epsilon_i\right]$ is more complicated as it requires the derivation of a multivariate normal cdf. We have:

$$E\left[u_i|\epsilon_i\right] = \left.\frac{\partial M_{u|\epsilon}(t)}{\partial t}\right|_{t=0}$$

Then

$$E\left[u_i|\epsilon_i\right] = \tilde{\boldsymbol{\pi}} + \left(\tilde{\mathbf{D}}\tilde{\mathbf{\Sigma}}\right)' \frac{\Phi_2^*\left(\mathbf{0}; \tilde{\boldsymbol{\kappa}}, \ddot{\mathbf{\Delta}}\right)}{\Phi_2\left(\mathbf{0}; \tilde{\boldsymbol{\kappa}}, \ddot{\mathbf{\Delta}}\right)}$$

where $\Phi_2^*\left(\mathbf{s}; \tilde{\boldsymbol{\kappa}}, \ddot{\mathbf{\Delta}}\right) = \frac{\partial\Phi_2\left(\mathbf{s}; \tilde{\boldsymbol{\kappa}}, \ddot{\mathbf{\Delta}}\right)}{\partial\mathbf{s}}$

Value

A data frame that contains individual (in-)efficiency estimates. These are ordered in the same way as the corresponding observations in the dataset used for the estimation.

- For object of class `'sfacross'` the following elements are returned:

`u`	Conditional inefficiency. In the case argument `udist` of sfacross is set to `'uniform'`, two conditional inefficiency estimates are returned: `u1` for the classic conditional inefficiency following Jondrow *et al.* (1982), and `u2` which is obtained when $\theta/\sigma_v \longrightarrow \infty$ (see Nguyen, 2010).
`uLB`	Lower bound for conditional inefficiency. Only when the argument `udist` of sfacross is set to `'hnormal'`, `'exponential'`, `'tnormal'` or `'uniform'`.
`uUB`	Upper bound for conditional inefficiency. Only when the argument `udist` of sfacross is set to `'hnormal'`, `'exponential'`, `'tnormal'` or `'uniform'`.
`teJLMS`	$\exp(-E[u\|\epsilon])$. When the argument `udist` of sfacross is set to `'uniform'`, `teJLMS1` = $\exp(-E[u_1\|\epsilon])$ and `teJLMS2` = $\exp(-E[u_2\|\epsilon])$. Only when `logDepVar = TRUE`.
`m`	Conditional model. Only when the argument `udist` of sfacross is set to `'hnormal'`, `'exponential'`, `'tnormal'`, or `'rayleigh'`.
`teMO`	$\exp(-m)$. Only when, in the function sfacross, `logDepVar = TRUE` and `udist = 'hnormal'`, `'exponential'`, `'tnormal'`, `'uniform'`, or `'rayleigh'`.
`teBC`	Battese and Coelli (1988) conditional efficiency. Only when, in the function sfacross, `logDepVar = TRUE`. In the case `udist = 'uniform'`, two conditional efficiency estimates are returned: `teBC1` which is the classic conditional efficiency following Battese and Coelli (1988) and `teBC2` when $\theta/\sigma_v \longrightarrow \infty$ (see Nguyen, 2010).
`teBC_reciprocal`	Reciprocal of Battese and Coelli (1988) conditional efficiency. Similar to `teBC` except that it is computed as $E[\exp(u)\|\epsilon]$.
`teBCLB`	Lower bound for Battese and Coelli (1988) conditional efficiency. Only when, in the function sfacross, `logDepVar = TRUE` and `udist = 'hnormal'`, `'exponential'`, `'tnormal'`, or `'uniform'`.
`teBCUB`	Upper bound for Battese and Coelli (1988) conditional efficiency. Only when, in the function sfacross, `logDepVar = TRUE` and `udist = 'hnormal'`, `'exponential'`, `'tnormal'`, or `'uniform'`.
`theta`	In the case `udist = 'uniform'`. $u \in [0, \theta]$.

- For object of class `'sfalcmcross'` the following elements are returned:

`Group_c`	Most probable class for each observation.
`PosteriorProb_c`	Highest posterior probability.
`u_c`	Conditional inefficiency of the most probable class given the posterior probability.
`teJLMS_c`	$\exp(-E[u_c\|\epsilon_c])$. Only when, in the function sfalcmcross `logDepVar = TRUE`.

`teBC_c` $E\left[\exp\left(-u_c\right)|\epsilon_c\right]$. Only when, in the function sfalcmcross `logDepVar = TRUE`.

`teBC_reciprocal_c`
$E\left[\exp\left(u_c\right)|\epsilon_c\right]$. Only when, in the function sfalcmcross `logDepVar = TRUE`.

`PosteriorProb_c#`
Posterior probability of class #.

`PriorProb_c#` Prior probability of class #.

`u_c#` Conditional inefficiency associated to class #, regardless of `Group_c`.

`teBC_c#` Conditional efficiency ($E\left[\exp\left(-u_c\right)|\epsilon_c\right]$) associated to class #, regardless of `Group_c`. Only when, in the function sfalcmcross `logDepVar = TRUE`.

`teBC_reciprocal_c#`
Reciprocal conditional efficiency ($E\left[\exp\left(u_c\right)|\epsilon_c\right]$) associated to class #, regardless of `Group_c`. Only when, in the function sfalcmcross `logDepVar = TRUE`.

`ineff_c#` Conditional inefficiency (`u_c`) for observations in class # only.

`effBC_c#` Conditional efficiency (`teBC_c`) for observations in class # only.

`ReffBC_c#` Reciprocal conditional efficiency (`teBC_reciprocal_c`) for observations in class # only.

`theta_c#` In the case `udist = 'uniform'`. $u \in [0, \theta_{c\#}]$.

- For object of class `'sfaselectioncross'` the following elements are returned:

`u` Conditional inefficiency.

`teJLMS` $\exp\left(-E[u|\epsilon]\right)$. Only when `logDepVar = TRUE`.

`teBC` Battese and Coelli (1988) conditional efficiency. Only when, in the function sfaselectioncross, `logDepVar = TRUE`.

`teBC_reciprocal`
Reciprocal of Battese and Coelli (1988) conditional efficiency. Similar to `teBC` except that it is computed as $E\left[\exp\left(u\right)|\epsilon\right]$.

References

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

Bera, A.K., and S.C. Sharma. 1999. Estimating production uncertainty in stochastic frontier production function models. *Journal of Productivity Analysis*, **12**:187-210.

Gonzalez-Farias, G., Dominguez-Molina, A., Gupta, A. K., 2004. Additive properties of skew normal random vectors. *Journal of Statistical Planning and Inference*. **126**: 521-534.

Greene, W., 2010. A stochastic frontier model with correction for sample selection. *Journal of Productivity Analysis*. **34**, 15–24.

Hjalmarsson, L., S.C. Kumbhakar, and A. Heshmati. 1996. DEA, DFA and SFA: A comparison. *Journal of Productivity Analysis*, **7**:303-327.

Horrace, W.C., and P. Schmidt. 1996. Confidence statements for efficiency estimates from stochastic frontier models. *Journal of Productivity Analysis*, **7**:257-282.

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

Lai, H. P., 2015. Maximum likelihood estimation of the stochastic frontier model with endogenous switching or sample selection. *Journal of Productivity Analysis*, **43**: 105-117.

Nguyen, N.B. 2010. Estimation of technical efficiency in stochastic frontier analysis. PhD Dissertation, Bowling Green State University, August.

See Also

`sfalcmcross`, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

`sfacross`, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

`sfaselectioncross` for sample selection in stochastic frontier model fitting function using cross-sectional or pooled data.

Examples

```
## Not run:
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog SFA (cost function) truncated normal with scaling property
tl_u_ts <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) + log(wl/wf) +
log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) + I(log(wl/wf) *
log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)), udist = 'tnormal',
muhet = ~ regu, uhet = ~ regu, data = utility, S = -1, scaling = TRUE, method = 'mla')
eff.tl_u_ts <- efficiencies(tl_u_ts)
head(eff.tl_u_ts)
summary(eff.tl_u_ts)

## End(Not run)
```

`electricity` *Data on U.S. electric power generation*

Description

This dataset is on electric power generation in the United States.

Format

A data frame with 123 observations on the following 9 variables.

firm Firm identification.

cost Total cost in 1970, MM USD.

output Output in million KwH.

lprice Labor price.

lshare Labor's cost share.

cprice Capital price.

cshare Capital's cost share.

fprice Fuel price.

fshare Fuel's cost share.

Details

The dataset is from Christensen and Greene (1976) and has also been used in Greene (1990).

Source

http://pages.stern.nyu.edu/~wgreene/Text/tables/tablelist5.htm

References

Christensen, L.R., and W.H. Greene. 1976. Economies of scale in US electric power generation. *The Journal of Political Economy*, **84**:655–676.

Greene, W.H. 1990. A Gamma-distributed stochastic frontier model. *Journal of Econometrics*, **46**:141–163.

Examples

```
str(electricity)
summary(electricity)
```

`extract` *Extract frontier information to be used with* ***texreg*** *package*

Description

Extract coefficients and additional information for stochastic frontier models returned by `sfacross`, `sfalcmcross`, or `sfaselectioncross`.

Usage

```
extract.sfacross(model, ...)

extract.sfalcmcross(model, ...)

extract.sfaselectioncross(model, ...)
```

Arguments

model	objects of class 'sfacross', 'sfalcmcross', or 'sfaselectioncross'
...	Currently ignored

Value

A texreg object representing the statistical model.

See Also

sfacross, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfalcmcross, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfaselectioncross for sample selection in stochastic frontier model fitting function using cross-sectional data.

Examples

```
hlf <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'hnormal', uhet = ~ regu, data = utility, S = -1, method = 'bfgs')
trnorm <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu, data = utility, S = -1, method = 'bfgs')

tscal <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu, uhet = ~ regu, data = utility,
S = -1, method = 'bfgs', scaling = TRUE)

expo <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'exponential', uhet = ~ regu, data = utility, S = -1, method = 'bfgs')

texreg::screenreg(list(hlf, trnorm, tscal, expo))
```

`fitted` *Extract fitted values of stochastic frontier models*

Description

`fitted` returns the fitted frontier values from stochastic frontier models estimated with `sfacross`, `sfalcmcross`, or `sfaselectioncross`.

Usage

```
## S3 method for class 'sfacross'
fitted(object, ...)

## S3 method for class 'sfalcmcross'
fitted(object, ...)

## S3 method for class 'sfaselectioncross'
fitted(object, ...)
```

Arguments

`object`	A stochastic frontier model returned by `sfacross`, `sfalcmcross`, or `sfaselectioncross`.
`...`	Currently ignored.

Value

In the case of an object of class `'sfacross'`, or `'sfaselectioncross'`, a vector of fitted values is returned.

In the case of an object of class `'sfalcmcross'`, a data frame containing the fitted values for each class is returned where each variable ends with `'_c#'`, `'#'` being the class number.

Note

The fitted values are ordered in the same way as the corresponding observations in the dataset used for the estimation.

See Also

`sfacross`, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

`sfalcmcross`, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

`sfaselectioncross` for sample selection in stochastic frontier model fitting function using cross-sectional or pooled data.

Examples

```
## Not run:
## Using data on eighty-two countries production (GDP)
# LCM Cobb Douglas (production function) half normal distribution
cb_2c_h <- sfalcmcross(formula = ly ~ lk + ll + yr, udist = 'hnormal',
data = worldprod)
fit.cb_2c_h <- fitted(cb_2c_h)
head(fit.cb_2c_h)

## End(Not run)
```

---

ic *Extract information criteria of stochastic frontier models*

Description

ic returns information criterion from stochastic frontier models estimated with sfacross, sfalcmcross, or sfaselectioncross.

Usage

```
## S3 method for class 'sfacross'
ic(object, IC = "AIC", ...)

## S3 method for class 'sfalcmcross'
ic(object, IC = "AIC", ...)

## S3 method for class 'sfaselectioncross'
ic(object, IC = "AIC", ...)
```

Arguments

object	A stochastic frontier model returned by sfacross, sfalcmcross, or sfaselectioncross.
IC	Character string. Information criterion measure. Three criteria are available: • 'AIC' for Akaike information criterion (default) • 'BIC' for Bayesian information criterion • 'HQIC' for Hannan-Quinn information criterion .
...	Currently ignored.

Details

The different information criteria are computed as follows:

- AIC: $-2 \log LL + 2 * K$
- BIC: $-2 \log LL + \log N * K$
- HQIC: $-2 \log LL + 2 \log [\log N] * K$

where LL is the maximum likelihood value, K the number of parameters estimated and N the number of observations.

Value

`ic` returns the value of the information criterion (AIC, BIC or HQIC) of the maximum likelihood coefficients.

See Also

`sfacross`, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

`sfalcmcross`, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

`sfaselectioncross` for sample selection in stochastic frontier model fitting function using cross-sectional or pooled data.

Examples

```
## Not run:
## Using data on Swiss railway
# LCM (cost function) half normal distribution
cb_2c_u <- sfalcmcross(formula = LNCT ~ LNQ2 + LNQ3 + LNNET + LNPK + LNPL,
udist = 'hnormal', uhet = ~ 1, data = swissrailways, S = -1, method='ucminf')
ic(cb_2c_u)
ic(cb_2c_u, IC = 'BIC')
ic(cb_2c_u, IC = 'HQIC')

## End(Not run)
```

---

logLik *Extract log-likelihood value of stochastic frontier models*

Description

`logLik` extracts the log-likelihood value(s) from stochastic frontier models estimated with `sfacross`, `sfalcmcross`, or `sfaselectioncross`.

Usage

```
## S3 method for class 'sfacross'
logLik(object, individual = FALSE, ...)

## S3 method for class 'sfalcmcross'
logLik(object, individual = FALSE, ...)

## S3 method for class 'sfaselectioncross'
logLik(object, individual = FALSE, ...)
```

Arguments

object	A stochastic frontier model returned by sfacross, sfalcmcross, or sfaselectioncross.
individual	Logical. If FALSE (default), the sum of all observations' log-likelihood values is returned. If TRUE, a vector of each observation's log-likelihood value is returned.
...	Currently ignored.

Value

logLik returns either an object of class 'logLik', which is the log-likelihood value with the total number of observations (nobs) and the number of free parameters (df) as attributes, when individual = FALSE, or a list of elements, containing the log-likelihood of each observation (logLik), the total number of observations (Nobs) and the number of free parameters (df), when individual = TRUE.

See Also

sfacross, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfalcmcross, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfaselectioncross for sample selection in stochastic frontier model fitting function using cross-sectional or pooled data.

Examples

```
## Not run:
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog SFA (cost function) truncated normal with scaling property
tl_u_ts <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu, uhet = ~ regu, data = utility, S = -1,
scaling = TRUE, method = 'mla')
logLik(tl_u_ts)

## Using data on eighty-two countries production (GDP)
# LCM Cobb Douglas (production function) half normal distribution
```

```
cb_2c_h <- sfalcmcross(formula = ly ~ lk + ll + yr, udist = 'hnormal',
data = worldprod, S = 1)
logLik(cb_2c_h, individual = TRUE)

## End(Not run)
```

`marginal` *Marginal effects of the inefficiency drivers in stochastic frontier models*

Description

This function returns marginal effects of the inefficiency drivers from stochastic frontier models estimated with `sfacross`, `sfalcmcross`, or `sfaselectioncross`.

Usage

```
## S3 method for class 'sfacross'
marginal(object, newData = NULL, ...)

## S3 method for class 'sfalcmcross'
marginal(object, newData = NULL, ...)

## S3 method for class 'sfaselectioncross'
marginal(object, newData = NULL, ...)
```

Arguments

`object`	A stochastic frontier model returned by `sfacross`, `sfalcmcross`, or `sfaselectioncross`.
`newData`	Optional data frame that is used to calculate the marginal effect of Z variables on inefficiency. If NULL (the default), the marginal estimates are calculated for the observations that were used in the estimation.
`...`	Currently ignored.

Details

`marginal` operates in the presence of exogenous variables that explain inefficiency, namely the inefficiency drivers ($uhet = Z_u$ or $muhet = Z_{mu}$).

Two components are computed for each variable: the marginal effects on the expected inefficiency ($\frac{\partial E[u]}{\partial Z_{mu}}$) and the marginal effects on the variance of inefficiency ($\frac{\partial V[u]}{\partial Z_{mu}}$).

The model also allows the Wang (2002) parametrization of μ and σ_u^2 by the same vector of exogenous variables. This double parameterization accounts for non-monotonic relationships between the inefficiency and its drivers.

Value

marginal returns a data frame containing the marginal effects of the Z_u variables on the expected inefficiency (each variable has the prefix 'Eu_') and on the variance of the inefficiency (each variable has the prefix 'Vu_').

In the case of the latent class stochastic frontier (LCM), each variable ends with '_c#' where '#' is the class number.

References

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

See Also

sfacross, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfalcmcross, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfaselectioncross for sample selection in stochastic frontier model fitting function using cross-sectional or pooled data.

Examples

```
## Not run:
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog SFA (cost function) truncated normal with scaling property
tl_u_ts <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu + wl, uhet = ~ regu + wl, data = utility,
S = -1, scaling = TRUE, method = 'mla')
marg.tl_u_ts <- marginal(tl_u_ts)
summary(marg.tl_u_ts)

## Using data on eighty-two countries production (GDP)
# LCM Cobb Douglas (production function) half normal distribution
cb_2c_h <- sfalcmcross(formula = ly ~ lk + ll + yr, udist = 'hnormal',
    data = worldprod, uhet = ~ initStat + h, S = 1, method = 'mla')
  marg.cb_2c_h <- marginal(cb_2c_h)
  summary(marg.cb_2c_h)

## End(Not run)
```

nobs *Extract total number of observations used in frontier models*

Description

This function extracts the total number of 'observations' from a fitted frontier model.

Usage

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

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

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

Arguments

object	a sfacross, sfalcmcross, or sfaselectioncross object for which the number of total observations is to be extracted.
...	Currently ignored.

Details

nobs gives the number of observations actually used by the estimation procedure. It is not necessarily the number of observations of the model frame (number of rows in the model frame), because sometimes the model frame is further reduced by the estimation procedure especially in the presence of NA. In the case of sfaselectioncross, nobs returns the number of observations used in the frontier equation.

Value

A single number, normally an integer.

See Also

sfacross, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfalcmcross, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfaselectioncross for sample selection in stochastic frontier model fitting function using cross-sectional or pooled data.

Examples

```
## Not run:
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog (cost function) half normal with heteroscedasticity
tl_u_h <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'hnormal', uhet = ~ regu, data = utility, S = -1, method = 'bfgs')
nobs(tl_u_h)

## End(Not run)
```

---

residuals *Extract residuals of stochastic frontier models*

Description

This function returns the residuals' values from stochastic frontier models estimated with sfacross, sfalcmcross, or sfaselectioncross.

Usage

```
## S3 method for class 'sfacross'
residuals(object, ...)

## S3 method for class 'sfalcmcross'
residuals(object, ...)

## S3 method for class 'sfaselectioncross'
residuals(object, ...)
```

Arguments

object	A stochastic frontier model returned by sfacross, sfalcmcross, or sfaselectioncross.
...	Currently ignored.

Value

When the object is of class 'sfacross', or 'sfaselectioncross', residuals returns a vector of residuals values.

When the object is of 'sfalcmcross', residuals returns a data frame containing the residuals values for each latent class, where each variable ends with '_c#', '#' being the class number.

Note

The residuals values are ordered in the same way as the corresponding observations in the dataset used for the estimation.

See Also

sfacross, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfalcmcross, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfaselectioncross for sample selection in stochastic frontier model fitting function using cross-sectional or pooled data.

Examples

```
## Not run:
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog SFA (cost function) truncated normal with scaling property
tl_u_ts <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu, uhet = ~ regu, data = utility, S = -1,
scaling = TRUE, method = 'mla')
resid.tl_u_ts <- residuals(tl_u_ts)
head(resid.tl_u_ts)

## Using data on eighty-two countries production (GDP)
# LCM Cobb Douglas (production function) half normal distribution
cb_2c_h <- sfalcmcross(formula = ly ~ lk + ll + yr, udist = 'hnormal',
data = worldprod, S = 1)
resid.cb_2c_h <- residuals(cb_2c_h)
head(resid.cb_2c_h)

## End(Not run)
```

---

ricephil *Data on rice production in the Philippines*

Description

This dataset contains annual data collected from 43 smallholder rice producers in the Tarlac region of the Philippines between 1990 and 1997.

Format

A data frame with 344 observations on the following 17 variables.

YEARDUM Time period (1= 1990, ..., 8 = 1997).

FARMERCODE Farmer code (1, ..., 43).

PROD Output (tonnes of freshly threshed rice).

AREA Area planted (hectares).

LABOR Labor used (man-days of family and hired labor).

NPK Fertiliser used (kg of active ingredients).

OTHER Other inputs used (Laspeyres index = 100 for Farm 17 in 1991).

PRICE Output price (pesos per kg).

AREAP Rental price of land (pesos per hectare).

LABORP Labor price (pesos per hired man-day).

NPKP Fertiliser price (pesos per kg of active ingredient).

OTHERP Price of other inputs (implicit price index).

AGE Age of the household head (years).

EDYRS Education of the household head (years).

HHSIZE Household size.

NADULT Number of adults in the household.

BANRAT Percentage of area classified as bantog (upland) fields.

Details

This dataset is published as supplement to Coelli *et al.* (2005). While most variables of this dataset were supplied by the International Rice Research Institute (IRRI), some were calculated by Coelli *et al.* (2005, see p. 325–326). The survey is described in Pandey *et al.* (1999).

Source

Supplementary files for Coelli *et al.* (2005), http://www.uq.edu.au/economics/cepa/crob2005/software/CROB2005.zip.

References

Coelli, T. J., Rao, D. S. P., O'Donnell, C. J., and Battese, G. E. 2005. *An Introduction to Efficiency and Productivity Analysis*, Springer, New York.

Pandey, S., Masciat, P., Velasco, L, and Villano, R. 1999. Risk analysis of a rainfed rice production system system in Tarlac, Central Luzon, Philippines. *Experimental Agriculture*, **35**:225–237.

Examples

```
str(ricephil)
summary(ricephil)
```

sfacross *Stochastic frontier estimation using cross-sectional data*

Description

sfacross is a symbolic formula-based function for the estimation of stochastic frontier models in the case of cross-sectional or pooled cross-sectional data, using maximum (simulated) likelihood - M(S)L.

The function accounts for heteroscedasticity in both one-sided and two-sided error terms as in Reifschneider and Stevenson (1991), Caudill and Ford (1993), Caudill *et al.* (1995) and Hadri (1999), but also heterogeneity in the mean of the pre-truncated distribution as in Kumbhakar *et al.* (1991), Huang and Liu (1994) and Battese and Coelli (1995).

Ten distributions are possible for the one-sided error term and eleven optimization algorithms are available.

The truncated normal - normal distribution with scaling property as in Wang and Schmidt (2002) is also implemented.

Usage

```
sfacross(
  formula,
  muhet,
  uhet,
  vhet,
  logDepVar = TRUE,
  data,
  subset,
  weights,
  wscale = TRUE,
  S = 1L,
  udist = "hnormal",
  scaling = FALSE,
  start = NULL,
  method = "bfgs",
  hessianType = 1L,
  simType = "halton",
  Nsim = 100,
  prime = 2L,
  burn = 10,
  antithetics = FALSE,
  seed = 12345,
  itermax = 2000,
  printInfo = FALSE,
  tol = 1e-12,
  gradtol = 1e-06,
  stepmax = 0.1,
```

```
  qac = "marquardt"
)

## S3 method for class 'sfacross'
print(x, ...)

## S3 method for class 'sfacross'
bread(x, ...)

## S3 method for class 'sfacross'
estfun(x, ...)
```

Arguments

formula	A symbolic description of the model to be estimated based on the generic function formula (see section 'Details').
muhet	A one-part formula to consider heterogeneity in the mean of the pre-truncated distribution (see section 'Details').
uhet	A one-part formula to consider heteroscedasticity in the one-sided error variance (see section 'Details').
vhet	A one-part formula to consider heteroscedasticity in the two-sided error variance (see section 'Details').
logDepVar	Logical. Informs whether the dependent variable is logged (TRUE) or not (FALSE). Default = TRUE.
data	The data frame containing the data.
subset	An optional vector specifying a subset of observations to be used in the optimization process.
weights	An optional vector of weights to be used for weighted log-likelihood. Should be NULL or numeric vector with positive values. When NULL, a numeric vector of 1 is used.
wscale	Logical. When weights is not NULL, a scaling transformation is used such that the weights sum to the sample size. Default TRUE. When FALSE no scaling is used.
S	If S = 1 (default), a production (profit) frontier is estimated: $\epsilon_i = v_i - u_i$. If S = -1, a cost frontier is estimated: $\epsilon_i = v_i + u_i$.
udist	Character string. Default = 'hnormal'. Distribution specification for the one-sided error term. 10 different distributions are available: • 'hnormal', for the half normal distribution (Aigner *et al.* 1977, Meeusen and Vandenbroeck 1977) • 'exponential', for the exponential distribution • 'tnormal' for the truncated normal distribution (Stevenson 1980) • 'rayleigh', for the Rayleigh distribution (Hajargasht 2015) • 'uniform', for the uniform distribution (Li 1996, Nguyen 2010) • 'gamma', for the Gamma distribution (Greene 2003)

- 'lognormal', for the log normal distribution (Migon and Medici 2001, Wang and Ye 2020)
- 'weibull', for the Weibull distribution (Tsionas 2007)
- 'genexponential', for the generalized exponential distribution (Papadopoulos 2020)
- 'tslaplace', for the truncated skewed Laplace distribution (Wang 2012).

scaling — Logical. Only when udist = 'tnormal' and scaling = TRUE, the scaling property model (Wang and Schmidt 2002) is estimated. Default = FALSE. (see section 'Details').

start — Numeric vector. Optional starting values for the maximum likelihood (ML) estimation.

method — Optimization algorithm used for the estimation. Default = 'bfgs'. 11 algorithms are available:

- 'bfgs', for Broyden-Fletcher-Goldfarb-Shanno (see maxBFGS)
- 'bhhh', for Berndt-Hall-Hall-Hausman (see maxBHHH)
- 'nr', for Newton-Raphson (see maxNR)
- 'nm', for Nelder-Mead (see maxNM)
- 'cg', for Conjugate Gradient (see maxCG)
- 'sann', for Simulated Annealing (see maxSANN)
- 'ucminf', for a quasi-Newton type optimisation with BFGS updating of the inverse Hessian and soft line search with a trust region type monitoring of the input to the line search algorithm (see ucminf)
- 'mla', for general-purpose optimization based on Marquardt-Levenberg algorithm (see mla)
- 'sr1', for Symmetric Rank 1 (see trust.optim)
- 'sparse', for trust regions and sparse Hessian (see trust.optim)
- 'nlminb', for optimization using PORT routines (see nlminb)

hessianType — Integer. If 1 (Default), analytic Hessian is returned for all the distributions. If 2, bhhh Hessian is estimated ($g'g$).

simType — Character string. If simType = 'halton' (Default), Halton draws are used for maximum simulated likelihood (MSL). If simType = 'ghalton', Generalized-Halton draws are used for MSL. If simType = 'sobol', Sobol draws are used for MSL. If simType = 'uniform', uniform draws are used for MSL. (see section 'Details').

Nsim — Number of draws for MSL. Default 100.

prime — Prime number considered for Halton and Generalized-Halton draws. Default = 2.

burn — Number of the first observations discarded in the case of Halton draws. Default = 10.

antithetics — Logical. Default = FALSE. If TRUE, antithetics counterpart of the uniform draws is computed. (see section 'Details').

seed — Numeric. Seed for the random draws.

itermax — Maximum number of iterations allowed for optimization. Default = 2000.

printInfo	Logical. Print information during optimization. Default = FALSE.
tol	Numeric. Convergence tolerance. Default = 1e-12.
gradtol	Numeric. Convergence tolerance for gradient. Default = 1e-06.
stepmax	Numeric. Step max for ucminf algorithm. Default = 0.1.
qac	Character. Quadratic Approximation Correction for 'bhhh' and 'nr' algorithms. If 'stephalving', the step length is decreased but the direction is kept. If 'marquardt' (default), the step length is decreased while also moving closer to the pure gradient direction. See maxBHHH and maxNR.
x	an object of class sfacross (returned by the function sfacross).
...	additional arguments of frontier are passed to sfacross; additional arguments of the print, bread, estfun, nobs methods are currently ignored.

Details

The stochastic frontier model for the cross-sectional data is defined as:

$$y_i = \alpha + \mathbf{x_i'}\boldsymbol{\beta} + v_i - Su_i$$

with

$$\epsilon_i = v_i - Su_i$$

where i is the observation, y is the output (cost, revenue, profit), $\mathbf{x}$ is the vector of main explanatory variables (inputs and other control variables), u is the one-sided error term with variance σ_u^2, and v is the two-sided error term with variance σ_v^2.

S = 1 in the case of production (profit) frontier function and S = -1 in the case of cost frontier function.

The model is estimated using maximum likelihood (ML) for most distributions except the Gamma, Weibull and log-normal distributions for which maximum simulated likelihood (MSL) is used. For this latter, several draws can be implemented namely Halton, Generalized Halton, Sobol and uniform. In the case of uniform draws, antithetics can also be computed: first Nsim/2 draws are obtained, then the Nsim/2 other draws are obtained as counterpart of one (1-draw).

To account for heteroscedasticity in the variance parameters of the error terms, a single part (right) formula can also be specified. To impose the positivity to these parameters, the variances are modelled as: $\sigma_u^2 = \exp(\boldsymbol{\delta}'\mathbf{Z}_u)$ or $\sigma_v^2 = \exp(\boldsymbol{\phi}'\mathbf{Z}_v)$, where $\mathbf{Z}_u$ and $\mathbf{Z}_v$ are the heteroscedasticity variables (inefficiency drivers in the case of $\mathbf{Z}_u$) and $\boldsymbol{\delta}$ and $\boldsymbol{\phi}$ the coefficients. In the case of heterogeneity in the truncated mean μ, it is modelled as $\mu = \boldsymbol{\omega}'\mathbf{Z}_\mu$. The scaling property can be applied for the truncated normal distribution: $u \sim h(\mathbf{Z}_u, \delta)u$ where u follows a truncated normal distribution $N^+(\tau, \exp(cu))$.

In the case of the truncated normal distribution, the convolution of u_i and v_i is:

$$f(\epsilon_i) = \frac{1}{\sqrt{\sigma_u^2 + \sigma_v^2}}\phi\left(\frac{S\epsilon_i + \mu}{\sqrt{\sigma_u^2 + \sigma_v^2}}\right)\Phi\left(\frac{\mu_{i*}}{\sigma_*}\right)\Big/\Phi\left(\frac{\mu}{\sigma_u}\right)$$

where

$$\mu_{i*} = \frac{\mu\sigma_v^2 - S\epsilon_i\sigma_u^2}{\sigma_u^2 + \sigma_v^2}$$

and

$$\sigma_*^2 = \frac{\sigma_u^2\sigma_v^2}{\sigma_u^2 + \sigma_v^2}$$

In the case of the half normal distribution the convolution is obtained by setting $\mu = 0$.

`sfacross` allows for the maximization of weighted log-likelihood. When option `weights` is specified and `wscale = TRUE`, the weights are scaled as:

$$new_{weights} = sample_{size} \times \frac{old_{weights}}{\sum(old_{weights})}$$

For complex problems, non-gradient methods (e.g. `nm` or `sann`) can be used to warm start the optimization and zoom in the neighborhood of the solution. Then a gradient-based methods is recommended in the second step. In the case of `sann`, we recommend to significantly increase the iteration limit (e.g. `itermax = 20000`). The Conjugate Gradient (`cg`) can also be used in the first stage.

A set of extractor functions for fitted model objects is available for objects of class `'sfacross'` including methods to the generic functions `print`, `summary`, `coef`, `fitted`, `logLik`, `residuals`, `vcov`, `efficiencies`, `ic`, `marginal`, `skewnessTest`, `estfun` and `bread` (from the **sandwich** package), `lmtest::coeftest()` (from the **lmtest** package).

Value

`sfacross` returns a list of class `'sfacross'` containing the following elements:

`call`	The matched call.
`formula`	The estimated model.
`S`	The argument `'S'`. See the section 'Arguments'.
`typeSfa`	Character string. 'Stochastic Production/Profit Frontier, e = v - u' when `S = 1` and 'Stochastic Cost Frontier, e = v + u' when `S = -1`.
`Nobs`	Number of observations used for optimization.
`nXvar`	Number of explanatory variables in the production or cost frontier.
`nmuZUvar`	Number of variables explaining heterogeneity in the truncated mean, only if `udist = 'tnormal'` or `'lognormal'`.
`scaling`	The argument `'scaling'`. See the section 'Arguments'.
`logDepVar`	The argument `'logDepVar'`. See the section 'Arguments'.
`nuZUvar`	Number of variables explaining heteroscedasticity in the one-sided error term.
`nvZVvar`	Number of variables explaining heteroscedasticity in the two-sided error term.
`nParm`	Total number of parameters estimated.
`udist`	The argument `'udist'`. See the section 'Arguments'.

startVal	Numeric vector. Starting value for M(S)L estimation.
dataTable	A data frame (tibble format) containing information on data used for optimization along with residuals and fitted values of the OLS and M(S)L estimations, and the individual observation log-likelihood. When weights is specified an additional variable is also provided in dataTable.
olsParam	Numeric vector. OLS estimates.
olsStder	Numeric vector. Standard errors of OLS estimates.
olsSigmasq	Numeric. Estimated variance of OLS random error.
olsLoglik	Numeric. Log-likelihood value of OLS estimation.
olsSkew	Numeric. Skewness of the residuals of the OLS estimation.
olsM3Okay	Logical. Indicating whether the residuals of the OLS estimation have the expected skewness.
CoelliM3Test	Coelli's test for OLS residuals skewness. (See Coelli, 1995).
AgostinoTest	D'Agostino's test for OLS residuals skewness. (See D'Agostino and Pearson, 1973).
isWeights	Logical. If TRUE weighted log-likelihood is maximized.
optType	Optimization algorithm used.
nIter	Number of iterations of the ML estimation.
optStatus	Optimization algorithm termination message.
startLoglik	Log-likelihood at the starting values.
mlLoglik	Log-likelihood value of the M(S)L estimation.
mlParam	Parameters obtained from M(S)L estimation.
gradient	Each variable gradient of the M(S)L estimation.
gradL_OBS	Matrix. Each variable individual observation gradient of the M(S)L estimation.
gradientNorm	Gradient norm of the M(S)L estimation.
invHessian	Covariance matrix of the parameters obtained from the M(S)L estimation.
hessianType	The argument 'hessianType'. See the section 'Arguments'.
mlDate	Date and time of the estimated model.
simDist	The argument 'simDist', only if udist = 'gamma', 'lognormal' or , 'weibull'. See the section 'Arguments'.
Nsim	The argument 'Nsim', only if udist = 'gamma', 'lognormal' or , 'weibull'. See the section 'Arguments'.
FiMat	Matrix of random draws used for MSL, only if udist = 'gamma', 'lognormal' or , 'weibull'.

Note

For the Halton draws, the code is adapted from the **mlogit** package.

References

Aigner, D., Lovell, C. A. K., and Schmidt, P. 1977. Formulation and estimation of stochastic frontier production function models. *Journal of Econometrics*, **6**(1), 21–37.

Battese, G. E., and Coelli, T. J. 1995. A model for technical inefficiency effects in a stochastic frontier production function for panel data. *Empirical Economics*, **20**(2), 325–332.

Caudill, S. B., and Ford, J. M. 1993. Biases in frontier estimation due to heteroscedasticity. *Economics Letters*, **41**(1), 17–20.

Caudill, S. B., Ford, J. M., and Gropper, D. M. 1995. Frontier estimation and firm-specific inefficiency measures in the presence of heteroscedasticity. *Journal of Business & Economic Statistics*, **13**(1), 105–111.

Coelli, T. 1995. Estimators and hypothesis tests for a stochastic frontier function - a Monte-Carlo analysis. *Journal of Productivity Analysis*, **6**:247–268.

D'Agostino, R., and E.S. Pearson. 1973. Tests for departure from normality. Empirical results for the distributions of b_2 and $\sqrt{b_1}$. *Biometrika*, **60**:613–622.

Greene, W. H. 2003. Simulated likelihood estimation of the normal-Gamma stochastic frontier function. *Journal of Productivity Analysis*, **19**(2-3), 179–190.

Hadri, K. 1999. Estimation of a doubly heteroscedastic stochastic frontier cost function. *Journal of Business & Economic Statistics*, **17**(3), 359–363.

Hajargasht, G. 2015. Stochastic frontiers with a Rayleigh distribution. *Journal of Productivity Analysis*, **44**(2), 199–208.

Huang, C. J., and Liu, J.-T. 1994. Estimation of a non-neutral stochastic frontier production function. *Journal of Productivity Analysis*, **5**(2), 171–180.

Kumbhakar, S. C., Ghosh, S., and McGuckin, J. T. 1991) A generalized production frontier approach for estimating determinants of inefficiency in U.S. dairy farms. *Journal of Business & Economic Statistics*, **9**(3), 279–286.

Li, Q. 1996. Estimating a stochastic production frontier when the adjusted error is symmetric. *Economics Letters*, **52**(3), 221–228.

Meeusen, W., and Vandenbroeck, J. 1977. Efficiency estimation from Cobb-Douglas production functions with composed error. *International Economic Review*, **18**(2), 435–445.

Migon, H. S., and Medici, E. V. 2001. Bayesian hierarchical models for stochastic production frontier. Lacea, Montevideo, Uruguay.

Nguyen, N. B. 2010. Estimation of technical efficiency in stochastic frontier analysis. PhD dissertation, Bowling Green State University, August.

Papadopoulos, A. 2021. Stochastic frontier models using the generalized exponential distribution. *Journal of Productivity Analysis*, **55**:15–29.

Reifschneider, D., and Stevenson, R. 1991. Systematic departures from the frontier: A framework for the analysis of firm inefficiency. *International Economic Review*, **32**(3), 715–723.

Stevenson, R. E. 1980. Likelihood Functions for Generalized Stochastic Frontier Estimation. *Journal of Econometrics*, **13**(1), 57–66.

Tsionas, E. G. 2007. Efficiency measurement with the Weibull stochastic frontier. *Oxford Bulletin of Economics and Statistics*, **69**(5), 693–706.

Wang, K., and Ye, X. 2020. Development of alternative stochastic frontier models for estimating time-space prism vertices. *Transportation*.

Wang, H.J., and Schmidt, P. 2002. One-step and two-step estimation of the effects of exogenous variables on technical efficiency levels. *Journal of Productivity Analysis*, **18**:129–144.

Wang, J. 2012. A normal truncated skewed-Laplace model in stochastic frontier analysis. Master thesis, Western Kentucky University, May.

See Also

`print` for printing `sfacross` object.

`summary` for creating and printing summary results.

`coef` for extracting coefficients of the estimation.

`efficiencies` for computing (in-)efficiency estimates.

`fitted` for extracting the fitted frontier values.

`ic` for extracting information criteria.

`logLik` for extracting log-likelihood value(s) of the estimation.

`marginal` for computing marginal effects of inefficiency drivers.

`residuals` for extracting residuals of the estimation.

`skewnessTest` for conducting residuals skewness test.

`vcov` for computing the variance-covariance matrix of the coefficients.

`bread` for bread for sandwich estimator.

`estfun` for gradient extraction for each observation.

`skewnessTest` for implementing skewness test.

Examples

```
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog (cost function) half normal with heteroscedasticity
tl_u_h <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'hnormal', uhet = ~ regu, data = utility, S = -1, method = 'bfgs')
summary(tl_u_h)

# Translog (cost function) truncated normal with heteroscedasticity
tl_u_t <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu, data = utility, S = -1, method = 'bhhh')
summary(tl_u_t)

# Translog (cost function) truncated normal with scaling property
tl_u_ts <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
```

```
udist = 'tnormal', muhet = ~ regu, uhet = ~ regu, data = utility, S = -1,
scaling = TRUE, method = 'mla')
summary(tl_u_ts)

## Using data on Philippine rice producers
# Cobb Douglas (production function) generalized exponential, and Weibull
# distributions

cb_p_ge <- sfacross(formula = log(PROD) ~ log(AREA) + log(LABOR) + log(NPK) +
log(OTHER), udist = 'genexponential', data = ricephil, S = 1, method = 'bfgs')
summary(cb_p_ge)

## Using data on U.S. electric utility industry
# Cost frontier Gamma distribution
tl_u_g <- sfacross(formula = log(cost/fprice) ~ log(output) + I(log(output)^2) +
I(log(lprice/fprice)) + I(log(cprice/fprice)), udist = 'gamma', uhet = ~ 1,
data = electricity, S = -1, method = 'bfgs', simType = 'halton', Nsim = 200,
hessianType = 2)
summary(tl_u_g)
```

`sfalcmcross` *Latent class stochastic frontier using cross-sectional data*

Description

`sfalcmcross` is a symbolic formula based function for the estimation of the latent class stochastic frontier model (LCM) in the case of cross-sectional or pooled cross-sectional data. The model is estimated using maximum likelihood (ML). See Orea and Kumbhakar (2004), Parmeter and Kumbhakar (2014, p282).

Only the half-normal distribution is possible for the one-sided error term. Eleven optimization algorithms are available.

The function also accounts for heteroscedasticity in both one-sided and two-sided error terms, as in Reifschneider and Stevenson (1991), Caudill and Ford (1993), Caudill *et al.* (1995) and Hadri (1999).

The model can estimate up to five classes.

Usage

```
sfalcmcross(
  formula,
  uhet,
  vhet,
  thet,
  logDepVar = TRUE,
  data,
  subset,
  weights,
```

```
  wscale = TRUE,
  S = 1L,
  udist = "hnormal",
  start = NULL,
  whichStart = 2L,
  initAlg = "nm",
  initIter = 100,
  lcmClasses = 2,
  method = "bfgs",
  hessianType = 1,
  itermax = 2000L,
  printInfo = FALSE,
  tol = 1e-12,
  gradtol = 1e-06,
  stepmax = 0.1,
  qac = "marquardt"
)

## S3 method for class 'sfalcmcross'
print(x, ...)

## S3 method for class 'sfalcmcross'
bread(x, ...)

## S3 method for class 'sfalcmcross'
estfun(x, ...)
```

Arguments

formula	A symbolic description of the model to be estimated based on the generic function formula (see section 'Details').
uhet	A one-part formula to account for heteroscedasticity in the one-sided error variance (see section 'Details').
vhet	A one-part formula to account for heteroscedasticity in the two-sided error variance (see section 'Details').
thet	A one-part formula to account for technological heterogeneity in the construction of the classes.
logDepVar	Logical. Informs whether the dependent variable is logged (TRUE) or not (FALSE). Default = TRUE.
data	The data frame containing the data.
subset	An optional vector specifying a subset of observations to be used in the optimization process.
weights	An optional vector of weights to be used for weighted log-likelihood. Should be NULL or numeric vector with positive values. When NULL, a numeric vector of 1 is used.

wscale — Logical. When weights is not NULL, a scaling transformation is used such that the weights sums to the sample size. Default TRUE. When FALSE no scaling is used.

S — If S = 1 (default), a production (profit) frontier is estimated: $\epsilon_i = v_i - u_i$. If S = -1, a cost frontier is estimated: $\epsilon_i = v_i + u_i$.

udist — Character string. Distribution specification for the one-sided error term. Only the half normal distribution 'hnormal' (Aigner *et al.*, 1977, Meeusen and Vandenbroeck, 1977) is currently implemented.

start — Numeric vector. Optional starting values for the maximum likelihood (ML) estimation.

whichStart — Integer. If 'whichStart = 1', the starting values are obtained from the method of moments. When 'whichStart = 2' (Default), the model is initialized by solving the homoscedastic pooled cross section SFA model.

initAlg — Character string specifying the algorithm used for initialization and obtain the starting values (when 'whichStart = 2'). Only **maxLik** package algorithms are available:

- 'bfgs', for Broyden-Fletcher-Goldfarb-Shanno (see maxBFGS)
- 'bhhh', for Berndt-Hall-Hall-Hausman (see maxBHHH)
- 'nr', for Newton-Raphson (see maxNR)
- 'nm', for Nelder-Mead - Default - (see maxNM)
- 'cg', for Conjugate Gradient (see maxCG)
- 'sann', for Simulated Annealing (see maxSANN)

initIter — Maximum number of iterations for initialization algorithm. Default 100.

lcmClasses — Number of classes to be estimated (default = 2). A maximum of five classes can be estimated.

method — Optimization algorithm used for the estimation. Default = 'bfgs'. 11 algorithms are available:

- 'bfgs', for Broyden-Fletcher-Goldfarb-Shanno (see maxBFGS)
- 'bhhh', for Berndt-Hall-Hall-Hausman (see maxBHHH)
- 'nr', for Newton-Raphson (see maxNR)
- 'nm', for Nelder-Mead (see maxNM)
- 'cg', for Conjugate Gradient (see maxCG)
- 'sann', for Simulated Annealing (see maxSANN)
- 'ucminf', for a quasi-Newton type optimization with BFGS updating of the inverse Hessian and soft line search with a trust region type monitoring of the input to the line search algorithm (see ucminf)
- 'mla', for general-purpose optimization based on Marquardt-Levenberg algorithm (see mla)
- 'sr1', for Symmetric Rank 1 (see trust.optim)
- 'sparse', for trust regions and sparse Hessian (see trust.optim)
- 'nlminb', for optimization using PORT routines (see nlminb)

hessianType — Integer. If 1 (default), analytic Hessian is returned. If 2, bhhh Hessian is estimated ($g'g$).

itermax	Maximum number of iterations allowed for optimization. Default = 2000.
printInfo	Logical. Print information during optimization. Default = FALSE.
tol	Numeric. Convergence tolerance. Default = 1e-12.
gradtol	Numeric. Convergence tolerance for gradient. Default = 1e-06.
stepmax	Numeric. Step max for ucminf algorithm. Default = 0.1.
qac	Character. Quadratic Approximation Correction for 'bhhh' and 'nr' algorithms. If 'qac = stephalving', the step length is decreased but the direction is kept. If 'qac = marquardt' (default), the step length is decreased while also moving closer to the pure gradient direction. See maxBHHH and maxNR.
x	an object of class sfalcmcross (returned by the function sfalcmcross).
...	additional arguments of frontier are passed to sfalcmcross; additional arguments of the print, bread, estfun, nobs methods are currently ignored.

Details

LCM is an estimation of a finite mixture of production functions:

$$y_i = \alpha_j + \mathbf{x_i'}\boldsymbol{\beta_j} + v_{i|j} - Su_{i|j}$$

$$\epsilon_{i|j} = v_{i|j} - Su_{i|j}$$

where i is the observation, j is the class, y is the output (cost, revenue, profit), x is the vector of main explanatory variables (inputs and other control variables), u is the one-sided error term with variance σ_u^2, and v is the two-sided error term with variance σ_v^2.

S = 1 in the case of production (profit) frontier function and S = -1 in the case of cost frontier function.

The contribution of observation i to the likelihood conditional on class j is defined as:

$$P(i|j) = \frac{2}{\sqrt{\sigma_{u|j}^2 + \sigma_{v|j}^2}} \phi\left(\frac{S\epsilon_{i|j}}{\sqrt{\sigma_{u|j}^2 + \sigma_{v|j}^2}}\right) \Phi\left(\frac{\mu_{i*|j}}{\sigma_{*|j}}\right)$$

where

$$\mu_{i*|j} = \frac{-S\epsilon_{i|j}\sigma_{u|j}^2}{\sigma_{u|j}^2 + \sigma_{v|j}^2}$$

and

$$\sigma_*^2 = \frac{\sigma_{u|j}^2\sigma_{v|j}^2}{\sigma_{u|j}^2 + \sigma_{v|j}^2}$$

The prior probability of using a particular technology can depend on some covariates (namely the variables separating the observations into classes) using a logit specification:

$$\pi(i,j) = \frac{\exp\left(\boldsymbol{\theta}_j' \mathbf{Z}_{hi}\right)}{\sum_{m=1}^{J} \exp\left(\boldsymbol{\theta}_m' \mathbf{Z}_{hi}\right)}$$

with $\mathbf{Z}_h$ the covariates, $\boldsymbol{\theta}$ the coefficients estimated for the covariates, and $\exp(\boldsymbol{\theta}_J' \mathbf{Z}_h) = 1$.

The unconditional likelihood of observation i is simply the average over the J classes:

$$P(i) = \sum_{m=1}^{J} \pi(i,m) P(i|m)$$

The number of classes to retain can be based on information criterion (see for instance `ic`).

Class assignment is based on the largest posterior probability. This probability is obtained using Bayes' rule, as follows for class j:

$$w\left(j|i\right) = \frac{P\left(i|j\right)\pi\left(i,j\right)}{\sum_{m=1}^{J} P\left(i|m\right)\pi\left(i,m\right)}$$

To accommodate heteroscedasticity in the variance parameters of the error terms, a single part (right) formula can also be specified. To impose the positivity on these parameters, the variances are modelled respectively as: $\sigma^2_{u|j} = \exp\left(\boldsymbol{\delta}_j' \mathbf{Z}_u\right)$ and $\sigma^2_{v|j} = \exp\left(\boldsymbol{\phi}_j' \mathbf{Z}_v\right)$, where Z_u and Z_v are the heteroscedasticity variables (inefficiency drivers in the case of $\mathbf{Z}_u$) and $\boldsymbol{\delta}$ and $\boldsymbol{\phi}$ the coefficients. `'sfalcmcross'` only supports the half-normal distribution for the one-sided error term.

`sfalcmcross` allows for the maximization of weighted log-likelihood. When option `weights` is specified and `wscale = TRUE`, the weights are scaled as:

$$new_{weights} = sample_{size} \times \frac{old_{weights}}{\sum(old_{weights})}$$

For complex problems, non-gradient methods (e.g. `nm` or `sann`) can be used to warm start the optimization and zoom in the neighborhood of the solution. Then a gradient-based methods is recommended in the second step. In the case of `sann`, we recommend to significantly increase the iteration limit (e.g. `itermax = 20000`). The Conjugate Gradient (`cg`) can also be used in the first stage.

A set of extractor functions for fitted model objects is available for objects of class `'sfalcmcross'` including methods to the generic functions `print`, `summary`, `coef`, `fitted`, `logLik`, `residuals`, `vcov`, `efficiencies`, `ic`, `marginal`, `estfun` and `bread` (from the **sandwich** package), `lmtest::coeftest()` (from the **lmtest** package).

Value

`sfalcmcross` returns a list of class `'sfalcmcross'` containing the following elements:

`call`	The matched call.
`formula`	Multi parts formula describing the estimated model.
`S`	The argument `'S'`. See the section 'Arguments'.
`typeSfa`	Character string. 'Latent Class Production/Profit Frontier, e = v - u' when `S = 1` and 'Latent Class Cost Frontier, e = v + u' when `S = -1`.

`Nobs`	Number of observations used for optimization.
`nXvar`	Number of main explanatory variables.
`nZHvar`	Number of variables in the logit specification of the finite mixture model (i.e. number of covariates).
`logDepVar`	The argument `'logDepVar'`. See the section 'Arguments'.
`nuZUvar`	Number of variables explaining heteroscedasticity in the one-sided error term.
`nvZVvar`	Number of variables explaining heteroscedasticity in the two-sided error term.
`nParm`	Total number of parameters estimated.
`udist`	The argument `'udist'`. See the section 'Arguments'.
`startVal`	Numeric vector. Starting value for ML estimation.
`dataTable`	A data frame (tibble format) containing information on data used for optimization along with residuals and fitted values of the OLS and ML estimations, and the individual observation log-likelihood. When `weights` is specified an additional variable is also provided in `dataTable`.
`initHalf`	When `start = NULL` and `whichStart == 2L`. Initial ML estimation with half normal distribution for the one-sided error term. Model to construct the starting values for the latent class estimation. Object of class `'maxLik'` and `'maxim'` returned.
`isWeights`	Logical. If `TRUE` weighted log-likelihood is maximized.
`optType`	The optimization algorithm used.
`nIter`	Number of iterations of the ML estimation.
`optStatus`	An optimization algorithm termination message.
`startLoglik`	Log-likelihood at the starting values.
`nClasses`	The number of classes estimated.
`mlLoglik`	Log-likelihood value of the ML estimation.
`mlParam`	Numeric vector. Parameters obtained from ML estimation.
`mlParamMatrix`	Double. Matrix of ML parameters by class.
`gradient`	Numeric vector. Each variable gradient of the ML estimation.
`gradL_OBS`	Matrix. Each variable individual observation gradient of the ML estimation.
`gradientNorm`	Numeric. Gradient norm of the ML estimation.
`invHessian`	The covariance matrix of the parameters obtained from the ML estimation.
`hessianType`	The argument `'hessianType'`. See the section 'Arguments'.
`mlDate`	Date and time of the estimated model.

Note

In the case of panel data, `sfalcmcross` estimates a pooled cross-section where the probability of belonging to a class a priori is not permanent (not fixed over time).

References

Aigner, D., Lovell, C. A. K., and P. Schmidt. 1977. Formulation and estimation of stochastic frontier production function models. *Journal of Econometrics*, **6**(1), 21–37.

Caudill, S. B., and J. M. Ford. 1993. Biases in frontier estimation due to heteroscedasticity. *Economics Letters*, **41**(1), 17–20.

Caudill, S. B., Ford, J. M., and D. M. Gropper. 1995. Frontier estimation and firm-specific inefficiency measures in the presence of heteroscedasticity. *Journal of Business & Economic Statistics*, **13**(1), 105–111.

Hadri, K. 1999. Estimation of a doubly heteroscedastic stochastic frontier cost function. *Journal of Business & Economic Statistics*, **17**(3), 359–363.

Meeusen, W., and J. Vandenbroeck. 1977. Efficiency estimation from Cobb-Douglas production functions with composed error. *International Economic Review*, **18**(2), 435–445.

Orea, L., and S.C. Kumbhakar. 2004. Efficiency measurement using a latent class stochastic frontier model. *Empirical Economics*, **29**, 169–183.

Parmeter, C.F., and S.C. Kumbhakar. 2014. Efficiency analysis: A primer on recent advances. *Foundations and Trends in Econometrics*, **7**, 191–385.

Reifschneider, D., and R. Stevenson. 1991. Systematic departures from the frontier: A framework for the analysis of firm inefficiency. *International Economic Review*, **32**(3), 715–723.

See Also

`print` for printing `sfalcmcross` object.

`summary` for creating and printing summary results.

`coef` for extracting coefficients of the estimation.

`efficiencies` for computing (in-)efficiency estimates.

`fitted` for extracting the fitted frontier values.

`ic` for extracting information criteria.

`logLik` for extracting log-likelihood value(s) of the estimation.

`marginal` for computing marginal effects of inefficiency drivers.

`residuals` for extracting residuals of the estimation.

`vcov` for computing the variance-covariance matrix of the coefficients.

`bread` for bread for sandwich estimator.

`estfun` for gradient extraction for each observation.

Examples

```
## Using data on eighty-two countries production (GDP)
# LCM Cobb Douglas (production function) half normal distribution
# Intercept and initStat used as separating variables
cb_2c_h1 <- sfalcmcross(formula = ly ~ lk + ll + yr, thet = ~initStat,
data = worldprod)
summary(cb_2c_h1)
```

```
# summary of the initial ML model
summary(cb_2c_h1$InitHalf)

# Only the intercept is used as the separating variable
# and only variable initStat is used as inefficiency driver
cb_2c_h3 <- sfalcmcross(formula = ly ~ lk + ll + yr, uhet = ~initStat,
data = worldprod)
summary(cb_2c_h3)
```

---

sfaR-deprecated *Deprecated functions of sfaR*

Description

These functions are provided for compatibility with older versions of 'sfaR' only, and could be defunct at a future release.

Usage

```
lcmcross(
  formula,
  uhet,
  vhet,
  thet,
  logDepVar = TRUE,
  data,
  subset,
  weights,
  wscale = TRUE,
  S = 1L,
  udist = "hnormal",
  start = NULL,
  whichStart = 2L,
  initAlg = "nm",
  initIter = 100,
  lcmClasses = 2,
  method = "bfgs",
  hessianType = 1,
  itermax = 2000L,
  printInfo = FALSE,
  tol = 1e-12,
  gradtol = 1e-06,
  stepmax = 0.1,
  qac = "marquardt"
)
```

```
## S3 method for class 'lcmcross'
print(x, ...)

## S3 method for class 'lcmcross'
bread(x, ...)

## S3 method for class 'lcmcross'
estfun(x, ...)

## S3 method for class 'lcmcross'
coef(object, extraPar = FALSE, ...)

## S3 method for class 'summary.lcmcross'
coef(object, ...)

## S3 method for class 'lcmcross'
fitted(object, ...)

## S3 method for class 'lcmcross'
ic(object, IC = "AIC", ...)

## S3 method for class 'lcmcross'
logLik(object, individual = FALSE, ...)

## S3 method for class 'lcmcross'
marginal(object, newData = NULL, ...)

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

## S3 method for class 'lcmcross'
residuals(object, ...)

## S3 method for class 'lcmcross'
summary(object, grad = FALSE, ci = FALSE, ...)

## S3 method for class 'summary.lcmcross'
print(x, digits = max(3, getOption("digits") - 2), ...)

## S3 method for class 'lcmcross'
efficiencies(object, level = 0.95, newData = NULL, ...)

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

Arguments

formula	A symbolic description of the model to be estimated based on the generic function formula (see section 'Details').

`uhet` A one-part formula to account for heteroscedasticity in the one-sided error variance (see section 'Details').

`vhet` A one-part formula to account for heteroscedasticity in the two-sided error variance (see section 'Details').

`thet` A one-part formula to account for technological heterogeneity in the construction of the classes.

`logDepVar` Logical. Informs whether the dependent variable is logged (`TRUE`) or not (`FALSE`). Default = `TRUE`.

`data` The data frame containing the data.

`subset` An optional vector specifying a subset of observations to be used in the optimization process.

`weights` An optional vector of weights to be used for weighted log-likelihood. Should be `NULL` or numeric vector with positive values. When `NULL`, a numeric vector of 1 is used.

`wscale` Logical. When `weights` is not `NULL`, a scaling transformation is used such that the `weights` sums to the sample size. Default `TRUE`. When `FALSE` no scaling is used.

`S` If `S = 1` (default), a production (profit) frontier is estimated: $\epsilon_i = v_i - u_i$. If `S = -1`, a cost frontier is estimated: $\epsilon_i = v_i + u_i$.

`udist` Character string. Distribution specification for the one-sided error term. Only the half normal distribution `'hnormal'` (Aigner *et al.*, 1977, Meeusen and Vandenbroeck, 1977) is currently implemented.

`start` Numeric vector. Optional starting values for the maximum likelihood (ML) estimation.

`whichStart` Integer. If `'whichStart = 1'`, the starting values are obtained from the method of moments. When `'whichStart = 2'` (Default), the model is initialized by solving the homoscedastic pooled cross section SFA model. `'whichStart = 1'` can be fast.

`initAlg` Character string specifying the algorithm used for initialization and obtain the starting values (when `'whichStart = 2'`). Only **maxLik** package algorithms are available:

- `'bfgs'`, for Broyden-Fletcher-Goldfarb-Shanno (see `maxBFGS`)
- `'bhhh'`, for Berndt-Hall-Hall-Hausman (see `maxBHHH`)
- `'nr'`, for Newton-Raphson (see `maxNR`)
- `'nm'`, for Nelder-Mead - Default - (see `maxNM`)
- `'cg'`, for Conjugate Gradient (see `maxCG`)
- `'sann'`, for Simulated Annealing (see `maxSANN`)

`initIter` Maximum number of iterations for initialization algorithm. Default `100`.

`lcmClasses` Number of classes to be estimated (default = `2`). A maximum of five classes can be estimated.

`method` Optimization algorithm used for the estimation. Default = `'bfgs'`. 11 algorithms are available:

- `'bfgs'`, for Broyden-Fletcher-Goldfarb-Shanno (see `maxBFGS`)

- 'bhhh', for Berndt-Hall-Hall-Hausman (see maxBHHH)
- 'nr', for Newton-Raphson (see maxNR)
- 'nm', for Nelder-Mead (see maxNM)
- 'cg', for Conjugate Gradient (see maxCG)
- 'sann', for Simulated Annealing (see maxSANN)
- 'ucminf', for a quasi-Newton type optimization with BFGS updating of the inverse Hessian and soft line search with a trust region type monitoring of the input to the line search algorithm (see ucminf)
- 'mla', for general-purpose optimization based on Marquardt-Levenberg algorithm (see mla)
- 'sr1', for Symmetric Rank 1 (see trust.optim)
- 'sparse', for trust regions and sparse Hessian (see trust.optim)
- 'nlminb', for optimization using PORT routines (see nlminb)

hessianType — Integer. If 1 (default), analytic Hessian is returned. If 2, bhhh Hessian is estimated ($g'g$).

itermax — Maximum number of iterations allowed for optimization. Default = 2000.

printInfo — Logical. Print information during optimization. Default = FALSE.

tol — Numeric. Convergence tolerance. Default = 1e-12.

gradtol — Numeric. Convergence tolerance for gradient. Default = 1e-06.

stepmax — Numeric. Step max for ucminf algorithm. Default = 0.1.

qac — Character. Quadratic Approximation Correction for 'bhhh' and 'nr' algorithms. If 'qac = stephalving', the step length is decreased but the direction is kept. If 'qac = marquardt' (default), the step length is decreased while also moving closer to the pure gradient direction. See maxBHHH and maxNR.

x — an object of class lcmcross (returned by the function lcmcross).

... — additional arguments of frontier are passed to lcmcross; additional arguments of the print, bread, estfun, nobs methods are currently ignored.

object — an object of class lcmcross (returned by the function lcmcross).

extraPar — Logical (default = FALSE). If TRUE, additional parameters are returned (see coef or vcov).

IC — Character string. Information criterion measure. Three criteria are available:

- 'AIC' for Akaike information criterion (default)
- 'BIC' for Bayesian information criterion
- 'HQIC' for Hannan-Quinn information criterion

.

individual — Logical. If FALSE (default), the sum of all observations' log-likelihood values is returned. If TRUE, a vector of each observation's log-likelihood value is returned.

newData — Optional data frame that is used to calculate the efficiency estimates. If NULL (the default), the efficiency estimates are calculated for the observations that were used in the estimation.

grad — Logical. Default = FALSE. If TRUE, the gradient for the maximum likelihood (ML) estimates of the different parameters is returned.

ci	Logical. Default = FALSE. If TRUE, the 95% confidence interval for the different parameters (OLS or/and ML estimates) is returned.
digits	Numeric. Number of digits displayed in values.
level	A number between between 0 and 0.9999 used for the computation of (in-)efficiency confidence intervals (defaut = 0.95). Not used in the case of lcmcross.

Details

The following functions are deprecated and could be removed from **sfaR** in a near future. Use the replacement indicated below:

- lcmcross: sfalcmcross
- bread.lcmcross: bread.sfalcmcross
- coef.lcmcross: coef.sfalcmcross
- coef.summary.lcmcross: coef.summary.sfalcmcross
- efficiencies.lcmcross: efficiencies.sfalcmcross
- estfun.lcmcross: estfun.sfalcmcross
- fitted.lcmcross: fitted.sfalcmcross
- ic.lcmcross: ic.sfalcmcross
- logLik.lcmcross: logLik.sfalcmcross
- marginal.lcmcross: marginal.sfalcmcross
- nobs.lcmcross: nobs.sfalcmcross
- print.lcmcross: print.sfalcmcross
- print.summary.lcmcross: print.summary.sfalcmcross
- residuals.lcmcross: residuals.sfalcmcross
- summary.lcmcross: summary.sfalcmcross
- vcov.lcmcross: vcov.sfalcmcross

---

sfaselectioncross *Sample selection in stochastic frontier estimation using cross-section data*

Description

sfaselectioncross is a symbolic formula based function for the estimation of the stochastic frontier model in the presence of sample selection. The model accommodates cross-sectional or pooled cross-sectional data. The model can be estimated using different quadrature approaches or maximum simulated likelihood (MSL). See Greene (2010).

Only the half-normal distribution is possible for the one-sided error term. Eleven optimization algorithms are available.

The function also accounts for heteroscedasticity in both one-sided and two-sided error terms, as in Reifschneider and Stevenson (1991), Caudill and Ford (1993), Caudill *et al.* (1995) and Hadri (1999).

Usage

```
sfaselectioncross(
  selectionF,
  frontierF,
  uhet,
  vhet,
  modelType = "greene10",
  logDepVar = TRUE,
  data,
  subset,
  weights,
  wscale = TRUE,
  S = 1L,
  udist = "hnormal",
  start = NULL,
  method = "bfgs",
  hessianType = 2L,
  lType = "ghermite",
  Nsub = 100,
  uBound = Inf,
  simType = "halton",
  Nsim = 100,
  prime = 2L,
  burn = 10,
  antithetics = FALSE,
  seed = 12345,
  itermax = 2000,
  printInfo = FALSE,
  intol = 1e-06,
  tol = 1e-12,
  gradtol = 1e-06,
  stepmax = 0.1,
  qac = "marquardt"
)

## S3 method for class 'sfaselectioncross'
print(x, ...)

## S3 method for class 'sfaselectioncross'
bread(x, ...)

## S3 method for class 'sfaselectioncross'
estfun(x, ...)
```

Arguments

selectionF	A symbolic (formula) description of the selection equation.
frontierF	A symbolic (formula) description of the outcome (frontier) equation.

uhet — A one-part formula to consider heteroscedasticity in the one-sided error variance (see section 'Details').

vhet — A one-part formula to consider heteroscedasticity in the two-sided error variance (see section 'Details').

modelType — Character string. Model used to solve the selection bias. Only the model discussed in Greene (2010) is currently available.

logDepVar — Logical. Informs whether the dependent variable is logged (TRUE) or not (FALSE). Default = TRUE.

data — The data frame containing the data.

subset — An optional vector specifying a subset of observations to be used in the optimization process.

weights — An optional vector of weights to be used for weighted log-likelihood. Should be NULL or numeric vector with positive values. When NULL, a numeric vector of 1 is used.

wscale — Logical. When weights is not NULL, a scaling transformation is used such that the weights sum to the sample size. Default TRUE. When FALSE no scaling is used.

S — If S = 1 (default), a production (profit) frontier is estimated: $\epsilon_i = v_i - u_i$. If S = -1, a cost frontier is estimated: $\epsilon_i = v_i + u_i$.

udist — Character string. Distribution specification for the one-sided error term. Only the half normal distribution 'hnormal' is currently implemented.

start — Numeric vector. Optional starting values for the maximum likelihood (ML) estimation.

method — Optimization algorithm used for the estimation. Default = 'bfgs'. 11 algorithms are available:

- 'bfgs', for Broyden-Fletcher-Goldfarb-Shanno (see maxBFGS)
- 'bhhh', for Berndt-Hall-Hall-Hausman (see maxBHHH)
- 'nr', for Newton-Raphson (see maxNR)
- 'nm', for Nelder-Mead (see maxNM)
- 'cg', for Conjugate Gradient (see maxCG)
- 'sann', for Simulated Annealing (see maxSANN)
- 'ucminf', for a quasi-Newton type optimization with BFGS updating of the inverse Hessian and soft line search with a trust region type monitoring of the input to the line search algorithm (see ucminf)
- 'mla', for general-purpose optimization based on Marquardt-Levenberg algorithm (see mla)
- 'sr1', for Symmetric Rank 1 (see trust.optim)
- 'sparse', for trust regions and sparse Hessian (see trust.optim)
- 'nlminb', for optimization using PORT routines (see nlminb)

hessianType — Integer. If 1, analytic Hessian is returned. If 2, bhhh Hessian is estimated ($g'g$). bhhh hessian is estimated by default as the estimation is conducted in two steps.

lType	Specifies the way the likelihood is estimated. Five possibilities are available: kronrod for Gauss-Kronrod quadrature (see integrate), hcubature and pcubature for adaptive integration over hypercubes (see hcubature and pcubature), ghermite for Gauss-Hermite quadrature (see gaussHermiteData), and msl for maximum simulated likelihood. Default ghermite.
Nsub	Integer. Number of subdivisions/nodes used for quadrature approaches. Default Nsub = 100.
uBound	Numeric. Upper bound for the inefficiency component when solving integrals using quadrature approaches except Gauss-Hermite for which the upper bound is automatically infinite (Inf). Default uBound = Inf.
simType	Character string. If simType = 'halton' (Default), Halton draws are used for maximum simulated likelihood (MSL). If simType = 'ghalton', Generalized-Halton draws are used for MSL. If simType = 'sobol', Sobol draws are used for MSL. If simType = 'uniform', uniform draws are used for MSL. (see section 'Details').
Nsim	Number of draws for MSL (default 100).
prime	Prime number considered for Halton and Generalized-Halton draws. Default = 2.
burn	Number of the first observations discarded in the case of Halton draws. Default = 10.
antithetics	Logical. Default = FALSE. If TRUE, antithetics counterpart of the uniform draws is computed. (see section 'Details').
seed	Numeric. Seed for the random draws.
itermax	Maximum number of iterations allowed for optimization. Default = 2000.
printInfo	Logical. Print information during optimization. Default = FALSE.
intol	Numeric. Integration tolerance for quadrature approaches (kronrod, hcubature, pcubature).
tol	Numeric. Convergence tolerance. Default = 1e-12.
gradtol	Numeric. Convergence tolerance for gradient. Default = 1e-06.
stepmax	Numeric. Step max for ucminf algorithm. Default = 0.1.
qac	Character. Quadratic Approximation Correction for 'bhhh' and 'nr' algorithms. If 'stephalving', the step length is decreased but the direction is kept. If 'marquardt' (default), the step length is decreased while also moving closer to the pure gradient direction. See maxBHHH and maxNR.
x	an object of class sfaselectioncross (returned by the function sfaselectioncross).
...	additional arguments of frontier are passed to sfaselectioncross; additional arguments of the print, bread, estfun, nobs methods are currently ignored.

Details

The current model is an extension of Heckman (1976, 1979) sample selection model to nonlinear models particularly stochastic frontier model. The model has first been discussed in Greene (2010), and an application can be found in Dakpo et al. (2021). Practically, we have:

$$y_{1i} = \begin{cases} 1 & \text{if} \quad y_{1i}^* > 0 \\ 0 & \text{if} \quad y_{1i}^* \leq 0 \end{cases}$$

where

$$y_{1i}^* = \mathbf{Z}_{si}'\gamma + w_i, \quad w_i \sim \mathcal{N}(0,1)$$

and

$$y_{2i} = \begin{cases} y_{2i}^* & \text{if} \quad y_{1i}^* > 0 \\ NA & \text{if} \quad y_{1i}^* \leq 0 \end{cases}$$

where

$$y_{2i}^* = \mathbf{x_i'}\beta + v_i - Su_i, \quad v_i = \sigma_v V_i \quad \wedge \quad V_i \sim \mathcal{N}(0,1), \quad u_i = \sigma_u |U_i| \quad \wedge \quad U_i \sim \mathcal{N}(0,1)$$

y_{1i} describes the selection equation while y_{2i} represents the frontier equation. The selection bias arises from the correlation between the two symmetric random components v_i and w_i:

$$(v_i, w_i) \sim \mathcal{N}_2\left[(0,0),(1,\rho\sigma_v,\sigma_v^2)\right]$$

Conditionaly on $|U_i|$, the probability associated to each observation is:

$$Pr\left[y_{1i}^* \leq 0\right]^{1-y_{1i}} \cdot \left\{f(y_{2i}|y_{1i}^* > 0) \times Pr\left[y_{1i}^* > 0\right]\right\}^{y_{1i}}$$

Using the conditional probability formula:

$$P\left(A \cap B\right) = P(A) \cdot P(B|A) = P(B) \cdot P(A|B)$$

Therefore:

$$f(y_{2i}|y_{1i}^* \geq 0) \cdot Pr\left[y_{1i}^* \geq 0\right] = f(y_{2i}) \cdot Pr(y_{1i}^* \geq 0|y_{2i})$$

Using the properties of a bivariate normal distribution, we have:

$$y_{i1}^*|y_{i2} \sim N\left(\mathbf{Z_{si}'}\boldsymbol{\gamma} + \frac{\rho}{\sigma_v}v_i, 1-\rho^2\right)$$

Hence conditionally on $|U_i|$, we have:

$$f(y_{2i}|y_{1i}^* \geq 0) \cdot Pr\left[y_{1i}^* \geq 0\right] = \frac{1}{\sigma_v}\phi\left(\frac{v_i}{\sigma_v}\right)\Phi\left(\frac{\mathbf{Z_{si}'}\boldsymbol{\gamma} + \frac{\rho}{\sigma_v}v_i}{\sqrt{1-\rho^2}}\right)$$

The conditional likelihood is equal to:

$$L_i\big||U_i| = \Phi(-\mathbf{Z}'_{\mathbf{si}}\boldsymbol{\gamma})^{1-y_{1i}} \times \left\{ \frac{1}{\sigma_v}\phi\left(\frac{y_{2i} - \mathbf{x}'_{\mathbf{i}}\boldsymbol{\beta} + S\sigma_u|U_i|}{\sigma_v}\right) \Phi\left(\frac{\mathbf{Z}'_{\mathbf{si}}\boldsymbol{\gamma} + \frac{\rho}{\sigma_v}\left(y_{2i} - \mathbf{x}'_{\mathbf{i}}\boldsymbol{\beta} + S\sigma_u|U_i|\right)}{\sqrt{1-\rho^2}}\right)\right\}^{y_{1i}}$$

Since the non-selected observations bring no additional information, the conditional likelihood to be considered is:

$$L_i\big||U_i| = \frac{1}{\sigma_v}\phi\left(\frac{y_{2i} - \mathbf{x}'_{\mathbf{i}}\boldsymbol{\beta} + S\sigma_u|U_i|}{\sigma_v}\right) \Phi\left(\frac{\mathbf{Z}'_{\mathbf{si}}\boldsymbol{\gamma} + \frac{\rho}{\sigma_v}\left(y_{2i} - \mathbf{x}'_{\mathbf{i}}\boldsymbol{\beta} + S\sigma_u|U_i|\right)}{\sqrt{1-\rho^2}}\right)$$

The unconditional likelihood is obtained by integrating $|U_i|$ out of the conditional likelihood. Thus

$$L_i = \int_{|U_i|} \frac{1}{\sigma_v}\phi\left(\frac{y_{2i} - \mathbf{x}'_{\mathbf{i}}\boldsymbol{\beta} + S\sigma_u|U_i|}{\sigma_v}\right) \Phi\left(\frac{\mathbf{Z}'_{\mathbf{si}}\boldsymbol{\gamma} + \frac{\rho}{\sigma_v}\left(y_{2i} - \mathbf{x}'_{\mathbf{i}}\boldsymbol{\beta} + S\sigma_u|U_i|\right)}{\sqrt{1-\rho^2}}\right) p\left(|U_i|\right) d|U_i|$$

To simplifiy the estimation, the likelihood can be estimated using a two-step approach. In the first step, the probit model can be run and estimate of γ can be obtained. Then, in the second step, the following model is estimated:

$$L_i = \int_{|U_i|} \frac{1}{\sigma_v}\phi\left(\frac{y_{2i} - \mathbf{x}'_{\mathbf{i}}\boldsymbol{\beta} + S\sigma_u|U_i|}{\sigma_v}\right) \Phi\left(\frac{a_i + \frac{\rho}{\sigma_v}\left(y_{2i} - \mathbf{x}'_{\mathbf{i}}\boldsymbol{\beta} + S\sigma_u|U_i|\right)}{\sqrt{1-\rho^2}}\right) p\left(|U_i|\right) d|U_i|$$

where $a_i = \mathbf{Z}'_{\mathbf{si}}\hat{\gamma}$. This likelihood can be estimated using five different approaches: Gauss-Kronrod quadrature, adaptive integration over hypercubes (hcubature and pcubature), Gauss-Hermite quadrature, and maximum simulated likelihood. We also use the BHHH estimator to obtain the asymptotic standard errors for the parameter estimators.

`sfaselectioncross` allows for the maximization of weighted log-likelihood. When option `weights` is specified and `wscale = TRUE`, the weights are scaled as:

$$new_{weights} = sample_{size} \times \frac{old_{weights}}{\sum(old_{weights})}$$

For complex problems, non-gradient methods (e.g. `nm` or `sann`) can be used to warm start the optimization and zoom in the neighborhood of the solution. Then a gradient-based methods is recommended in the second step. In the case of `sann`, we recommend to significantly increase the iteration limit (e.g. `itermax = 20000`). The Conjugate Gradient (`cg`) can also be used in the first stage.

A set of extractor functions for fitted model objects is available for objects of class `'sfaselectioncross'` including methods to the generic functions `print`, `summary`, `coef`, `fitted`, `logLik`, `residuals`, `vcov`, `efficiencies`, `ic`, `marginal`, `estfun` and `bread` (from the **sandwich** package), `lmtest::coeftest()` (from the **lmtest** package).

Value

sfaselectioncross returns a list of class 'sfaselectioncross' containing the following elements:

call	The matched call.
selectionF	The selection equation formula.
frontierF	The frontier equation formula.
S	The argument 'S'. See the section 'Arguments'.
typeSfa	Character string. 'Stochastic Production/Profit Frontier, e = v - u' when S = 1 and 'Stochastic Cost Frontier, e = v + u' when S = -1.
Ninit	Number of initial observations in all samples.
Nobs	Number of observations used for optimization.
nXvar	Number of explanatory variables in the production or cost frontier.
logDepVar	The argument 'logDepVar'. See the section 'Arguments'.
nuZUvar	Number of variables explaining heteroscedasticity in the one-sided error term.
nvZVvar	Number of variables explaining heteroscedasticity in the two-sided error term.
nParm	Total number of parameters estimated.
udist	The argument 'udist'. See the section 'Arguments'.
startVal	Numeric vector. Starting value for M(S)L estimation.
dataTable	A data frame (tibble format) containing information on data used for optimization along with residuals and fitted values of the OLS and M(S)L estimations, and the individual observation log-likelihood. When argument weights is specified, an additional variable is provided in dataTable.
lpmObj	Linear probability model used for initializing the first step probit model.
probitObj	Probit model. Object of class 'maxLik' and 'maxim'.
ols2stepParam	Numeric vector. OLS second step estimates for selection correction. Inverse Mills Ratio is introduced as an additional explanatory variable.
ols2stepStder	Numeric vector. Standard errors of OLS second step estimates.
ols2stepSigmasq	Numeric. Estimated variance of OLS second step random error.
ols2stepLoglik	Numeric. Log-likelihood value of OLS second step estimation.
ols2stepSkew	Numeric. Skewness of the residuals of the OLS second step estimation.
ols2stepM3Okay	Logical. Indicating whether the residuals of the OLS second step estimation have the expected skewness.
CoelliM3Test	Coelli's test for OLS residuals skewness. (See Coelli, 1995).
AgostinoTest	D'Agostino's test for OLS residuals skewness. (See D'Agostino and Pearson, 1973).
isWeights	Logical. If TRUE weighted log-likelihood is maximized.
lType	Type of likelihood estimated. See the section 'Arguments'.
optType	Optimization algorithm used.

nIter	Number of iterations of the ML estimation.
optStatus	Optimization algorithm termination message.
startLoglik	Log-likelihood at the starting values.
mlLoglik	Log-likelihood value of the M(S)L estimation.
mlParam	Parameters obtained from M(S)L estimation.
gradient	Each variable gradient of the M(S)L estimation.
gradL_OBS	Matrix. Each variable individual observation gradient of the M(S)L estimation.
gradientNorm	Gradient norm of the M(S)L estimation.
invHessian	Covariance matrix of the parameters obtained from the M(S)L estimation.
hessianType	The argument 'hessianType'. See the section 'Arguments'.
mlDate	Date and time of the estimated model.
simDist	The argument 'simDist', only if lType = 'msl'. See the section 'Arguments'.
Nsim	The argument 'Nsim', only if lType = 'msl'. See the section 'Arguments'.
FiMat	Matrix of random draws used for MSL, only if lType = 'msl'.
gHermiteData	List. Gauss-Hermite quadrature rule as provided by gaussHermiteData. Only if lType = 'ghermite'.
Nsub	Number of subdivisions used for quadrature approaches.
uBound	Upper bound for the inefficiency component when solving integrals using quadrature approaches except Gauss-Hermite for which the upper bound is automatically infinite (Inf).
intol	Integration tolerance for quadrature approaches except Gauss-Hermite.

Note

For the Halton draws, the code is adapted from the **mlogit** package.

References

Caudill, S. B., and Ford, J. M. 1993. Biases in frontier estimation due to heteroscedasticity. *Economics Letters*, **41**(1), 17–20.

Caudill, S. B., Ford, J. M., and Gropper, D. M. 1995. Frontier estimation and firm-specific inefficiency measures in the presence of heteroscedasticity. *Journal of Business & Economic Statistics*, **13**(1), 105–111.

Coelli, T. 1995. Estimators and hypothesis tests for a stochastic frontier function - a Monte-Carlo analysis. *Journal of Productivity Analysis*, **6**:247–268.

D'Agostino, R., and E.S. Pearson. 1973. Tests for departure from normality. Empirical results for the distributions of b_2 and $\sqrt{b_1}$. *Biometrika*, **60**:613–622.

Dakpo, K. H., Latruffe, L., Desjeux, Y., Jeanneaux, P., 2022. Modeling heterogeneous technologies in the presence of sample selection: The case of dairy farms and the adoption of agri-environmental schemes in France. *Agricultural Economics*, **53**(3), 422-438.

Greene, W., 2010. A stochastic frontier model with correction for sample selection. *Journal of Productivity Analysis*. **34**, 15–24.

Hadri, K. 1999. Estimation of a doubly heteroscedastic stochastic frontier cost function. *Journal of Business & Economic Statistics*, **17**(3), 359–363.

Heckman, J., 1976. Discrete, qualitative and limited dependent variables. *Ann Econ Soc Meas.* **4**, 475–492.

Heckman, J., 1979. Sample Selection Bias as a Specification Error. *Econometrica*. **47**, 153–161.

Reifschneider, D., and Stevenson, R. 1991. Systematic departures from the frontier: A framework for the analysis of firm inefficiency. *International Economic Review*, **32**(3), 715–723.

See Also

`print` for printing `sfaselectioncross` object.

`summary` for creating and printing summary results.

`coef` for extracting coefficients of the estimation.

`efficiencies` for computing (in-)efficiency estimates.

`fitted` for extracting the fitted frontier values.

`ic` for extracting information criteria.

`logLik` for extracting log-likelihood value(s) of the estimation.

`marginal` for computing marginal effects of inefficiency drivers.

`residuals` for extracting residuals of the estimation.

`vcov` for computing the variance-covariance matrix of the coefficients.

`bread` for bread for sandwich estimator.

`estfun` for gradient extraction for each observation.

Examples

```
## Not run:

## Simulated example

N <- 2000  # sample size
set.seed(12345)
z1 <- rnorm(N)
z2 <- rnorm(N)
v1 <- rnorm(N)
v2 <- rnorm(N)
e1 <- v1
e2 <- 0.7071 * (v1 + v2)
ds <- z1 + z2 + e1
d <- ifelse(ds > 0, 1, 0)
u <- abs(rnorm(N))
x1 <- rnorm(N)
x2 <- rnorm(N)
y <- x1 + x2 + e2 - u
data <- cbind(y = y, x1 = x1, x2 = x2, z1 = z1, z2 = z2, d = d)
```

```
## Estimation using quadrature (Gauss-Kronrod)

selecRes1 <- sfaselectioncross(selectionF = d ~ z1 + z2, frontierF = y ~ x1 + x2,
modelType = 'greene10', method = 'bfgs',
logDepVar = TRUE, data = as.data.frame(data),
S = 1L, udist = 'hnormal', lType = 'kronrod', Nsub = 100, uBound = Inf,
simType = 'halton', Nsim = 300, prime = 2L, burn = 10, antithetics = FALSE,
seed = 12345, itermax = 2000, printInfo = FALSE)

summary(selecRes1)

## Estimation using maximum simulated likelihood

selecRes2 <- sfaselectioncross(selectionF = d ~ z1 + z2, frontierF = y ~ x1 + x2,
modelType = 'greene10', method = 'bfgs',
logDepVar = TRUE, data = as.data.frame(data),
S = 1L, udist = 'hnormal', lType = 'msl', Nsub = 100, uBound = Inf,
simType = 'halton', Nsim = 300, prime = 2L, burn = 10, antithetics = FALSE,
seed = 12345, itermax = 2000, printInfo = FALSE)

summary(selecRes2)


## End(Not run)
```

---

skewnessTest *Skewness test for stochastic frontier models*

Description

skewnessTest computes skewness test for stochastic frontier models (i.e. objects of class 'sfacross').

Usage

```
skewnessTest(object, test = "agostino")
```

Arguments

object	An object of class 'sfacross', returned by sfacross.
test	A character string specifying the test to implement. If 'agostino' (default), D'Agostino skewness test is implemented (D'Agostino and Pearson, 1973). If 'coelli', Coelli skewness test is implemented (Coelli, 1995).

Value

skewnessTest returns the results of either the D'Agostino's or the Coelli's skewness test.

Note

skewnessTest is currently only available for object of class 'sfacross'.

References

Coelli, T. 1995. Estimators and hypothesis tests for a stochastic frontier function - a Monte-Carlo analysis. *Journal of Productivity Analysis*, **6**:247–268.

D'Agostino, R., and E.S. Pearson. 1973. Tests for departure from normality. Empirical results for the distributions of b_2 and $\sqrt{b_1}$. *Biometrika*, **60**:613–622.

Examples

```
## Not run:
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog SFA (cost function) truncated normal with scaling property
tl_u_ts <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu, uhet = ~ regu, data = utility, S = -1,
scaling = TRUE, method = 'mla')
skewnessTest(tl_u_ts)
skewnessTest(tl_u_ts, test = 'coelli')

## End(Not run)
```

---

summary *Summary of results for stochastic frontier models*

Description

Create and print summary results for stochastic frontier models returned by sfacross, sfalcmcross, or sfaselectioncross.

Usage

```
## S3 method for class 'sfacross'
summary(object, grad = FALSE, ci = FALSE, ...)

## S3 method for class 'summary.sfacross'
print(x, digits = max(3, getOption("digits") - 2), ...)

## S3 method for class 'sfalcmcross'
summary(object, grad = FALSE, ci = FALSE, ...)

## S3 method for class 'summary.sfalcmcross'
print(x, digits = max(3, getOption("digits") - 2), ...)
```

```
## S3 method for class 'sfaselectioncross'
summary(object, grad = FALSE, ci = FALSE, ...)

## S3 method for class 'summary.sfaselectioncross'
print(x, digits = max(3, getOption("digits") - 2), ...)
```

Arguments

object	An object of either class 'sfacross' returned by the function sfacross, or 'sfalcmcross' returned by the function sfalcmcross, or class 'sfaselectioncross' returned by the function sfaselectioncross.
grad	Logical. Default = FALSE. If TRUE, the gradient for the maximum likelihood (ML) estimates of the different parameters is returned.
ci	Logical. Default = FALSE. If TRUE, the 95% confidence interval for the different parameters (OLS or/and ML estimates) is returned.
...	Currently ignored.
x	An object of either class 'summary.sfacross', 'summary.sfalcmcross', or 'summary.sfaselectioncross'.
digits	Numeric. Number of digits displayed in values.

Value

The summary method returns a list of class 'summary.sfacross', 'summary.sfalcmcross', or 'summary.sfaselectioncross' that contains the same elements as an object returned by sfacross, sfalcmcross, or sfaselectioncross with the following additional elements:

AIC	Akaike information criterion.
BIC	Bayesian information criterion.
HQIC	Hannan-Quinn information criterion.
sigmavSq	For object of class 'sfacross' or 'sfaselectioncross'. Variance of the two-sided error term (σ_v^2).
sigmauSq	For object of class 'sfacross' or 'sfaselectioncross'. Parametrization of the variance of the one-sided error term (σ_u^2).
Varu	For object of class 'sfacross' or 'sfaselectioncross'. Variance of the one-sided error term.
theta	For object of class 'sfacross' with 'udist = uniform'. Θ value in the case the uniform distribution is defined as: $u_i \in [0, \Theta]$.
Eu	For object of class 'sfacross' or 'sfaselectioncross'. Expected unconditional inefficiency ($E[u]$).
Expu	For object of class 'sfacross' or 'sfaselectioncross'. Expected unconditional efficiency ($E[\exp(u)]$).
olsRes	For object of class 'sfacross'. Matrix of OLS estimates, their standard errors, t-values, P-values, and when ci = TRUE their confidence intervals.

ols2StepRes	For object of class 'sfaselectioncross'. Matrix of OLS 2 step estimates, their standard errors, t-values, P-values, and when ci = TRUE their confidence intervals.
mlRes	Matrix of ML estimates, their standard errors, z-values, asymptotic P-values, and when grad = TRUE their gradient, ci = TRUE their confidence intervals.
chisq	For object of class 'sfacross'. Chi-square statistics of the difference between the stochastic frontier and the OLS.
df	Degree of freedom for the inefficiency model.

See Also

sfacross, for the stochastic frontier analysis model fitting function for cross-sectional or pooled data.

sfalcmcross, for the latent class stochastic frontier analysis model fitting function for cross-sectional or pooled data.

sfaselectioncross for sample selection in stochastic frontier model fitting function for cross-sectional or pooled data.

print for printing sfacross object.

coef for extracting coefficients of the estimation.

efficiencies for computing (in-)efficiency estimates.

fitted for extracting the fitted frontier values.

ic for extracting information criteria.

logLik for extracting log-likelihood value(s) of the estimation.

marginal for computing marginal effects of inefficiency drivers.

residuals for extracting residuals of the estimation.

vcov for computing the variance-covariance matrix of the coefficients.

bread for bread for sandwich estimator.

estfun for gradient extraction for each observation.

skewnessTest for implementing skewness test.

Examples

```
## Using data on fossil fuel fired steam electric power generation plants in the U.S.
# Translog SFA (cost function) truncated normal with scaling property
tl_u_ts <- sfacross(formula = log(tc/wf) ~ log(y) + I(1/2 * (log(y))^2) +
log(wl/wf) + log(wk/wf) + I(1/2 * (log(wl/wf))^2) + I(1/2 * (log(wk/wf))^2) +
I(log(wl/wf) * log(wk/wf)) + I(log(y) * log(wl/wf)) + I(log(y) * log(wk/wf)),
udist = 'tnormal', muhet = ~ regu, uhet = ~ regu, data = utility, S = -1,
scaling = TRUE, method = 'mla')
summary(tl_u_ts, grad = TRUE, ci = TRUE)
```

`swissrailways` *Data on Swiss railway companies*

Description

This dataset is an unbalanced panel of 50 Swiss railway companies over the period 1985-1997.

Format

A data frame with 605 observations on the following 42 variables.

ID Firm identification.

YEAR Year identification.

NI Number of years observed.

STOPS Number of stops in network.

NETWORK Network length (in meters).

NARROW_T Dummy variable for railroads with narrow track.

RACK Dummy variable for 'rack rail' in network.

TUNNEL Dummy variable for network with tunnels over 300 meters on average.

T Time indicator, first year = 0.

Q2 Passenger output – passenger km.

Q3 Freight output – ton km.

CT Total cost (1,000 Swiss franc).

PL Labor price.

PE Electricity price.

PK Capital price.

VIRAGE 1 for railroads with curvy tracks.

LNCT Log of `CT/PE`.

LNQ2 Log of `Q2`.

LNQ3 Log of `Q3`.

LNNET Log of `NETWORK/1000`.

LNPL Log of `PL/PE`.

LNPE Log of `PE`.

LNPK Log of `PK/PE`.

LNSTOP Log of `STOPS`.

MLNQ2 Mean of `LNQ2`.

MLNQ3 Mean of `LNQ3`.

MLNNET Mean of `LNNET`.

MLNPL Mean of `LNPL`.

MLNPK Mean of `LNPK`.

MLNSTOP Mean of `LNSTOP`.

Details

The dataset is extracted from the annual reports of the Swiss Federal Office of Statistics on public transport companies and has been used in Farsi *et al.* (2005).

Source

http://pages.stern.nyu.edu/~wgreene/Text/Edition7/tablelist8new.htm

http://people.stern.nyu.edu/wgreene/Microeconometrics.htm

References

Farsi, M., M. Filippini, and W. Greene. 2005. Efficiency measurement in network industries: Application to the Swiss railway companies. *Journal of Regulatory Economics*, **28**:69–90.

Examples

```
str(swissrailways)
```

---

utility *Data on U.S. electricity generating plants*

Description

This dataset contains data on fossil fuel fired steam electric power generation plants in the United States between 1986 and 1996.

Format

A data frame with 791 observations on the following 11 variables.

firm Plant identification.

year Year identification.

y Net-steam electric power generation in megawatt-hours.

regu Dummy variable which takes a value equal to 1 if the power plant is in a state which enacted legislation or issued a regulatory order to implement retail access during the sample period, and 0 otherwise.

k Capital stock.

labor Labor and maintenance.

fuel Fuel.

wl Labor price.

wf Fuel price.

wk Capital price.

tc Total cost.

Details

The dataset has been used in Kumbhakar *et al.* (2014).

Source

https://sites.google.com/site/sfbook2014/home/for-stata-v12-v13-v14

References

Kumbhakar, S.C., H.J. Wang, and A. Horncastle. 2014. *A Practitioner's Guide to Stochastic Frontier Analysis Using Stata*. Cambridge University Press.

Examples

```
str(utility)
summary(utility)
```

---

vcov *Compute variance-covariance matrix of stochastic frontier models*

Description

vcov computes the variance-covariance matrix of the maximum likelihood (ML) coefficients from stochastic frontier models estimated with sfacross, sfalcmcross, or sfaselectioncross.

Usage

```
## S3 method for class 'sfacross'
vcov(object, extraPar = FALSE, ...)

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

## S3 method for class 'sfaselectioncross'
vcov(object, extraPar = FALSE, ...)
```

Arguments

object	A stochastic frontier model returned by sfacross, sfalcmcross, or sfaselectioncross.
extraPar	Logical. Only available for non heteroscedastic models returned by sfacross and sfaselectioncross. Default = FALSE. If TRUE, variances and covariances of additional parameters are returned: sigmaSq = sigmauSq + sigmavSq lambdaSq = sigmauSq/sigmavSq sigmauSq = $\exp(Wu) = \exp(\boldsymbol{\delta}\mathbf{Z}_u)$ sigmavSq = $\exp(Wv) = \exp(\boldsymbol{\phi}\mathbf{Z}_v)$

sigma = sigmaSq^0.5
lambda = lambdaSq^0.5
sigmau = sigmauSq^0.5
sigmav = sigmavSq^0.5
gamma = sigmauSq/(sigmauSq + sigmavSq)

... Currently ignored

Details

The variance-covariance matrix is obtained by the inversion of the negative Hessian matrix. Depending on the distribution and the 'hessianType' option, the analytical/numeric Hessian or the bhhh Hessian is evaluated.

The argument extraPar, is currently available only for objects of class 'sfacross' and 'sfaselectioncross'. When 'extraPar = TRUE', the variance-covariance of the additional parameters is obtained using the delta method.

Value

The variance-covariance matrix of the maximum likelihood coefficients is returned.

See Also

sfacross, for the stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfalcmcross, for the latent class stochastic frontier analysis model fitting function using cross-sectional or pooled data.

sfaselectioncross for sample selection in stochastic frontier model fitting function using cross-sectional data.

Examples

```
## Using data on Spanish dairy farms
# Cobb Douglas (production function) half normal distribution
cb_s_h <- sfacross(formula = YIT ~ X1 + X2 + X3 + X4, udist = 'hnormal',
data = dairyspain, S = 1, method = 'bfgs')
vcov(cb_s_h)
vcov(cb_s_h, extraPar = TRUE)

# Other variance-covariance matrices can be obtained using the sandwich package

# Robust variance-covariance matrix

requireNamespace('sandwich', quietly = TRUE)

sandwich::vcovCL(cb_s_h)

# Coefficients and standard errors can be obtained using lmtest package
```

```
requireNamespace('lmtest', quietly = TRUE)

lmtest::coeftest(cb_s_h, vcov. = sandwich::vcovCL)

# Clustered standard errors

lmtest::coeftest(cb_s_h, vcov. = sandwich::vcovCL, cluster = ~ FARM)

# Doubly clustered standard errors

lmtest::coeftest(cb_s_h, vcov. = sandwich::vcovCL, cluster = ~ FARM + YEAR)

# BHHH standard errors

lmtest::coeftest(cb_s_h, vcov. = sandwich::vcovOPG)

# Adjusted BHHH standard errors

lmtest::coeftest(cb_s_h, vcov. = sandwich::vcovOPG, adjust = TRUE)

## Using data on eighty-two countries production (GDP)
# LCM Cobb Douglas (production function) half normal distribution
cb_2c_h <- sfalcmcross(formula = ly ~ lk + ll + yr, udist = 'hnormal',
data = worldprod, uhet = ~ initStat, S = 1)
vcov(cb_2c_h)
```

---

worldprod *Data on world production*

Description

This dataset provides information on production related variables for eighty-two countries over the period 1960–1987.

Format

A data frame with 2,296 observations on the following 12 variables.

country Country name.

code Country identification.

yr Year identification.

y GDP in 1987 U.S. dollars.

k Physical capital stock in 1987 U.S. dollars.

l Labor (number of individuals in the workforce between the age of 15 and 64).

h Human capital-adjusted labor.

ly Log of y.

lk Log of k.

ll Log of l.

lh Log of h.

initStat Log of the initial capital to labor ratio of each country, lk - ll, measured at the beginning of the sample period.

Details

The dataset is from the World Bank STARS database and has been used in Kumbhakar *et al.* (2014).

Source

https://sites.google.com/site/sfbook2014/home/for-stata-v12-v13-v14

References

Kumbhakar, S.C., H.J. Wang, and A. Horncastle. 2014. *A Practitioner's Guide to Stochastic Frontier Analysis Using Stata*. Cambridge University Press.

Examples

```
str(worldprod)
summary(worldprod)
```

Index