Package ‘smfa’

May 9, 2026

Title Stochastic Metafrontier Analysis

Version 1.0.0

Description Implements stochastic metafrontier analysis for productivity and performance benchmarking across firms operating under different technologies. Contains routines for the deterministic metafrontier envelope of O'Donnell et al. (2008) <doi:10.1007/s00181-007-0119-4> via linear and quadratic programming, and the stochastic metafrontier of Huang et al. (2014) <doi:10.1007/s11123-014-0402-2>. Also supports latent class stochastic metafrontier analysis and sample selection correction stochastic metafrontier models. Depends on the 'sfaR' package by Dakpo et al. (2023) <https://CRAN.R-project.org/package=sfaR>.

License GPL (>= 3)

URL https://github.com/SulmanOlieko/smfa,
https://SulmanOlieko.github.io/smfa/

BugReports https://github.com/SulmanOlieko/smfa/issues

Depends R (>= 3.5.0), sfaR

Imports stats

Suggests knitr, rmarkdown, pkgdown, lmtest

VignetteBuilder knitr

Config/Needs/website pkgdown

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

NeedsCompilation no

Author Sulman Olieko Owili [aut, cre] (ORCID: <https://orcid.org/0000-0001-7401-5326>)

Maintainer Sulman Olieko Owili <oliekosulman@gmail.com>

Repository CRAN

Date/Publication 2026-04-28 19:00:02 UTC

Contents

`coef` *Extract coefficients of stochastic metafrontier models*

Description

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

From on object of class `'smfa'`, it extracts only the estimated coefficients.

Usage

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

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

Arguments

`object`	A stochastic metafrontier model returned by `smfa`, or an object of class `'summary.smfa'`.
`...`	Currently ignored.

Value

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

For objects of class `'smfa'`, `coef` returns a numeric vector of the estimated coefficients.

See Also

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

`efficiencies` *Compute efficiency estimates and metatechnology ratios from stochastic metafrontier models*

Description

`efficiencies` returns all efficiency estimates and metatechnology ratio (MTR) measures for objects of class `"smfa"` returned by `smfa`. The function supports models estimated via linear programming (LP), quadratic programming (QP), and stochastic second-stage SFA (`"sfa"`), and for each observation it computes the group-specific technical efficiency, the metafrontier technical efficiency, and the metatechnology ratio (MTR), using both the Jondrow, Lovell, Materov, and Schmidt (1982) (JLMS) and the Battese and Coelli (1988) (BC) estimators. Additional model-specific columns are returned depending on `groupType`.

Usage

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

Arguments

`object`	An object of class `"smfa"` returned by `smfa`.
`level`	A number strictly between 0 and 0.9999 specifying the nominal coverage for (in-)efficiency confidence intervals. Default `0.95`. This argument is passed to the underlying `efficiencies` method of the group-level model (class `"sfacross"`, `"sfalcmcross"`, or `"sfaselectioncross"`).
`newData`	Optional data frame for out-of-sample prediction of efficiency estimates. When `NULL` (default), efficiencies are computed for the observations used in the estimation.
`...`	Further arguments (currently ignored).

Details

Group-specific efficiency estimates: For each group, the group-level frontier model is estimated by maximising the log-likelihood using the distribution specified by `udist` in `smfa`. Given the estimated composite error $\varepsilon_i = v_i - Su_i$, the conditional distribution of $u_i \mid \varepsilon_i$ is used to compute:

- the JLMS estimator (Jondrow, Lovell, Materov, and Schmidt, 1982): $\hat{u}_i = E[u_i \mid \varepsilon_i]$, and $TE^g_{JLMS,i} = \exp(-\hat{u}_i)$;
- the BC estimator (Battese and Coelli, 1988): $TE^g_{BC,i} = E[\exp(-u_i) \mid \varepsilon_i]$;
- the mode estimator: $m_i = \mathrm{mode}[u_i \mid \varepsilon_i]$, and $TE^g_{\mathrm{mode},i} = \exp(-m_i)$;
- confidence bounds on u_i and $TE^g_{BC,i}$ at the nominal level `level`.

For `groupType = "sfaselectioncross"`, all estimates are `NA` for observations not selected into the sample (binary selection indicator equal to 0). For `groupType = "sfalcmcross"`, the composite efficiencies `u_g`, `TE_group_JLMS`, and `TE_group_BC` are computed using the posterior-probability-weighted class assignments.

Metatechnology ratio and metafrontier efficiency: The MTR measures how far the group frontier lies below the metafrontier for each observation. Let $\ln \hat{y}_i^g$ be the group-specific fitted frontier value and $\ln \hat{y}_i^*$ the metafrontier fitted value.

- For metaMethod = "lp" or "qp" (Battese, Rao, and O'Donnell, 2004):

$$MTR_i = \exp\big(-\max\big\{S \cdot (\ln \hat{y}_i^* - \ln \hat{y}_i^g),\ 0\big\}\big)$$

  where $S = 1$ for production/profit frontiers and $S = -1$ for cost frontiers. The technology gap $U_i = \max\{S \cdot (\ln \hat{y}_i^* - \ln \hat{y}_i^g), 0\}$ is stored in u_meta.
- For metaMethod = "sfa" with sfaApproach = "huang" (Huang, Huang, and Liu, 2014):

$$MTR_i = TE_i^* = \exp(-U_i)$$

  where U_i is the one-sided error term from the second-stage SFA.
- For metaMethod = "sfa" with sfaApproach = "ordonnell" (O'Donnell, Rao, and Battese, 2008): $MTR_i = TE_i^{*,\mathrm{sfa}}/TE_i^g$, where $TE_i^{*,\mathrm{sfa}}$ is the technical efficiency from the second-stage SFA fitted on the LP envelope values.

The metafrontier technical efficiency is then:

$$TE_i^* = TE_i^g \times MTR_i$$

computed separately for the JLMS and BC group efficiency estimators. Both MTR_JLMS and MTR_BC are reported, distinguishing which group-level efficiency estimator was used as the basis.

Value

A data frame with one row per observation (in the original row order), containing the following columns. The exact set of columns depends on groupType:

Columns present for all model types:

id Observation identifier. Contains the row name of each observation as it appeared in the data supplied to smfa. When the data frame has no explicit row names, sequential integers ("1", "2", ...) are used. This column is always the first column of the returned data frame.

<group> **or** Group_c The technology group identifier for each observation. For groupType = "sfacross" and "sfaselectioncross", this column takes the name of the user-supplied group variable and contains the group label to which each observation belongs. For groupType = "sfalcmcross", it is named Group_c and contains the integer index of the latent class assigned by the maximum posterior probability criterion.

u_g Group-specific conditional mean of the inefficiency term, computed as $E[u_i \mid \varepsilon_i]$. This is the JLMS (Jondrow, Lovell, Materov, and Schmidt, 1982) point estimate of the inefficiency at the group-frontier level. For groupType = "sfaselectioncross", u_g is NA for observations not selected into the sample (selection indicator = 0).

TE_group_JLMS Group-specific technical efficiency using the Jondrow, Lovell, Materov, and Schmidt (1982) estimator: $TE_i^g = \exp(-E[u_i \mid \varepsilon_i])$. For groupType = "sfaselectioncross", NA for non-selected observations.

TE_group_BC Group-specific technical efficiency using the Battese and Coelli (1988) estimator: $TE_i^g = E[\exp(-u_i) \mid \varepsilon_i]$. For groupType = "sfaselectioncross", NA for non-selected observations.

TE_group_BC_reciprocal Reciprocal of the Battese and Coelli (1988) group technical efficiency: $1/TE_i^{g,BC}$. For production frontiers this equals the cost-efficiency index implied by the BC estimator. Present for all three model types. For groupType = "sfaselectioncross", NA for non-selected observations.

u_meta Metafrontier inefficiency, measuring the technology-gap component $U_i \geq 0$ that separates the group frontier from the global metafrontier. Computed from the second-stage SFA when metaMethod = "sfa", or derived from the LP/QP gap as $U_i = \max\{S \cdot (\ln \hat{y}_i^* - \ln \hat{y}_i^g), 0\}$ when metaMethod = "lp" or "qp".

TE_meta_JLMS Metafrontier technical efficiency based on the JLMS group efficiency: $TE^*_{JLMS,i} = TE^g_{JLMS,i} \times MTR_{JLMS,i}$.

TE_meta_BC Metafrontier technical efficiency based on the Battese and Coelli (1988) group efficiency: $TE^*_{BC,i} = TE^g_{BC,i} \times MTR_{BC,i}$.

MTR_JLMS Metatechnology ratio computed using the JLMS group efficiency: $MTR_{JLMS,i} = TE^*_{JLMS,i}/TE^g_{JLMS,i} = \exp(-U_i)$. Values range from 0 to 1. A value of 1 indicates that the group frontier for this observation coincides with the metafrontier.

MTR_BC Metatechnology ratio computed using the Battese and Coelli (1988) group efficiency: $MTR_{BC,i} = TE^*_{BC,i}/TE^g_{BC,i} = \exp(-U_i)$.

Additional columns for groupType = "sfacross" **only:**

uLB_g, uUB_g Lower and upper bounds of the level confidence interval for the conditional mean inefficiency u_g, constructed using the asymptotic distribution of the conditional estimator. Available for distributions with closed-form expressions for the confidence bounds, such as udist = "hnormal" and udist = "tnormal".

m_g Mode of the conditional distribution of the one-sided error term $u_i \mid \varepsilon_i$. This is an alternative point estimate of inefficiency. Available for distributions for which the conditional mode has a closed-form expression.

TE_group_mode Group-specific technical efficiency evaluated at the conditional mode: $TE^g_{\text{mode},i} = \exp(-m_i)$.

teBCLB_g, teBCUB_g Lower and upper bounds of the level confidence interval for the Battese and Coelli (1988) group technical efficiency TE_group_BC. Constructed from the corresponding bounds on the conditional distribution of $\exp(-u_i \mid \varepsilon_i)$.

Additional columns for groupType = "sfalcmcross" **only:**

PosteriorProb_c Posterior probability that observation i belongs to its assigned class (the one with the highest posterior probability). Computed via Bayes' rule as $P(j \mid y_i, x_i) \propto \pi(i,j)\,P(i \mid j)$, where $\pi(i,j)$ is the prior class probability and $P(i \mid j)$ is the class-conditional likelihood.

PosteriorProb_cJ **(per class,** $J = 1, 2, \ldots$**)** Posterior probability of belonging to latent class J, computed via Bayes' rule for each class separately. One column is produced for each estimated class.

PriorProb_cJ **(per class,** $J = 1, 2, \ldots$**)** Prior (unconditional) probability of belonging to latent class J, given by the logistic specification $\pi(i,J) = \exp(\boldsymbol{\theta}'_J \mathbf{Z}_{hi}) / \sum_m \exp(\boldsymbol{\theta}'_m \mathbf{Z}_{hi})$.

u_cJ **(per class,** $J = 1, 2, \ldots$**)** Conditional mean of the inefficiency term for class J: $E[u_{i|J} \mid \varepsilon_{i|J}]$.

teBC_cJ **(per class,** $J = 1, 2, \ldots$**)** Battese and Coelli (1988) technical efficiency for class J: $E[\exp(-u_{i|J}) \mid \varepsilon_{i|J}]$.

`teBC_reciprocal_cJ` **(per class, $J = 1, 2, \ldots$)** Reciprocal of the class-J Battese and Coelli (1988) efficiency: $1/TE_{i|J}^{BC}$.

`ineff_cJ` **(per class, $J = 1, 2, \ldots$)** Inefficiency estimate for the observation restricted to class J (i.e. the value assigned to the class to which the observation *does* belong; `NA` for other classes).

`effBC_cJ` **(per class, $J = 1, 2, \ldots$)** Battese and Coelli (1988) efficiency for the observation's assigned class; `NA` for non-assigned classes.

`ReffBC_cJ` **(per class, $J = 1, 2, \ldots$)** Reciprocal Battese and Coelli (1988) efficiency for the observation's assigned class; `NA` for non-assigned classes.

References

Battese, G. E., and Coelli, T. J. 1988. Prediction of firm-level technical efficiencies with a generalized frontier production function and panel data. *Journal of Econometrics*, **38**(3), 387–399. doi:10.1016/03044076(88)90053X

Battese, G. E., Rao, D. S. P., and O'Donnell, C. J. 2004. A metafrontier production function for estimation of technical efficiencies and technology gaps for firms operating under different technologies. *Journal of Productivity Analysis*, **21**(1), 91–103. doi:10.1023/B:PROD.0000012454.06094.29

Huang, C. J., Huang, T.-H., and Liu, N.-H. 2014. A new approach to estimating the metafrontier production function based on a stochastic frontier framework. *Journal of Productivity Analysis*, **42**(3), 241–254. doi:10.1007/s1112301404022

Jondrow, J., Lovell, C. A. K., Materov, I. S., and Schmidt, P. 1982. On the estimation of technical inefficiency in the stochastic frontier production function model. *Journal of Econometrics*, **19**(2-3), 233–238. doi:10.1016/03044076(82)900045

O'Donnell, C. J., Rao, D. S. P., and Battese, G. E. 2008. Metafrontier frameworks for the study of firm-level efficiencies and technology ratios. *Empirical Economics*, **34**(2), 231–255. doi:10.1007/s0018100701194

Orea, L., and Kumbhakar, S. C. 2004. Efficiency measurement using a latent class stochastic frontier model. *Empirical Economics*, **29**(1), 169–183. doi:10.1007/s0018100301842

Dakpo, K. H., Desjeux, Y., and Latruffe, L. 2023. sfaR: Stochastic Frontier Analysis using R. R package version 1.0.1. `https://CRAN.R-project.org/package=sfaR`

See Also

`smfa`, for the stochastic metafrontier analysis model fitting function using cross-sectional or pooled data; `efficiencies`, for the underlying group-level efficiency extractor.

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

Description

`fitted` returns the fitted frontier values from stochastic metafrontier models estimated with `smfa`.

Usage

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

Arguments

`object`	A stochastic metafrontier model returned by `smfa`.
`...`	Currently ignored.

Value

A vector of fitted values is returned.

Note

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

See Also

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

`ic` — *Extract information criteria of stochastic metafrontier models*

Description

`ic` returns information criterion from stochastic metafrontier models estimated with `smfa`.

Usage

```
## S3 method for class 'smfa'
ic(object, IC = NULL, ...)
```

Arguments

`object` — A stochastic metafrontier model returned by `smfa`.

`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 a data frame with the values of the information criteria (AIC, BIC and HQIC) of the maximum likelihood coefficients. If the `IC` argument is provided, it returns only the requested criterion as a numeric value.

See Also

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

`logLik` *Extract log-likelihood value of stochastic metafrontier models*

Description

`logLik` extracts the log-likelihood value(s) from stochastic metafrontier models estimated with `smfa`.

Usage

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

Arguments

`object`	A stochastic metafrontier model returned by `smfa`.
`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

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

---

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

Description

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

Usage

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

Arguments

object	a smfa 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.

Value

A single number, normally an integer.

See Also

smfa, for the stochastic metafrontier analysis model fitting function using cross-sectional or pooled data `residuals` *Extract residuals of stochastic metafrontier models*

Description

This function returns the residuals' values from stochastic metafrontier models estimated with `smfa`.

Usage

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

Arguments

`object`	A stochastic metafrontier model returned by `smfa`.
`...`	Currently ignored.

Value

`residuals` returns a vector of residuals values.

Note

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

See Also

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

`smfa` *Stochastic metafrontier estimation*

Description

`smfa` estimates a stochastic metafrontier model for cross-sectional or pooled data. The function follows the theoretical frameworks of Battese, Rao, and O'Donnell (2004) and O'Donnell, Rao, and Battese (2008), and additionally implements the two-stage stochastic approach of Huang, Huang, and Liu (2014). Three types of group-level frontier models are supported: standard stochastic frontier analysis (`sfacross`), sample selection stochastic frontier analysis (`sfaselectioncross`), and latent class stochastic frontier analysis (`sfalcmcross`).

Usage

```
smfa(
  formula,
  muhet,
  uhet,
  vhet,
  thet,
  logDepVar = TRUE,
  data,
  subset,
  weights,
  wscale = TRUE,
  group = NULL,
  S = 1L,
  udist = "hnormal",
  start = NULL,
  scaling = FALSE,
  modelType = "greene10",
  groupType = "sfacross",
  metaMethod = "lp",
  sfaApproach = "ordonnell",
  selectionF = NULL,
  lcmClasses = 2L,
  whichStart = 2L,
  initAlg = "nm",
  initIter = 100L,
  lType = "ghermite",
  Nsub = 100L,
  uBound = Inf,
  intol = 1e-06,
  method = "bfgs",
  hessianType = NULL,
  simType = "halton",
  Nsim = 100L,
  prime = 2L,
  burn = 10L,
  antithetics = FALSE,
  seed = 12345L,
  itermax = 2000L,
  printInfo = FALSE,
  tol = 1e-12,
  gradtol = 1e-06,
  stepmax = 0.1,
  qac = "marquardt",
  ...
)

## S3 method for class 'smfa'
```

```
print(x, ...)
```

Arguments

`formula` A symbolic description of the frontier model to be estimated, based on the generic function `formula`. For `groupType = "sfaselectioncross"`, this argument specifies the frontier (outcome) equation and must be a standard formula whose left-hand side is the output (or cost) variable and whose right-hand side contains the frontier regressors (see also `selectionF`).

`muhet` A one-part formula to account for heterogeneity in the mean of the pre-truncated normal distribution. Applicable only when `groupType = "sfacross"` and `udist = "tnormal"`. The variables specified model the conditional mean $\mu_i = \boldsymbol{\omega}'\mathbf{Z}_\mu$ of the truncated normal inefficiency distribution (see section 'Details').

`uhet` A one-part formula to account for heteroscedasticity in the one-sided error variance. Applicable for all three model types. The variance of the inefficiency term is modelled as $\sigma_u^2 = \exp(\boldsymbol{\delta}'\mathbf{Z}_u)$, where $\mathbf{Z}_u$ are the inefficiency drivers and $\boldsymbol{\delta}$ the associated coefficients (see section 'Details').

`vhet` A one-part formula to account for heteroscedasticity in the two-sided error variance. Applicable for all three model types. The variance of the noise term is modelled as $\sigma_v^2 = \exp(\boldsymbol{\phi}'\mathbf{Z}_v)$, where $\mathbf{Z}_v$ are the heteroscedasticity variables and $\boldsymbol{\phi}$ the coefficients (see section 'Details').

`thet` A one-part formula to account for technological heterogeneity in the construction of the latent classes. Applicable only when `groupType = "sfalcmcross"`. The variables specified enter the logit formulation that determines the prior class membership probabilities $\pi(i, j)$ (see section 'Details').

`logDepVar` Logical. Informs whether the dependent variable is logged (`TRUE`) or not (`FALSE`). Default `TRUE`. Must match the transformation applied to the left-hand side of `formula`.

`data` A data frame containing all variables referenced in `formula`, `selectionF`, `muhet`, `uhet`, `vhet`, `thet`, and `group`.

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

`weights` An optional vector of weights to be used for weighted log-likelihood estimation. Should be `NULL` or a numeric vector with strictly positive values. When `NULL` (default), all observations receive equal weight.

`wscale` Logical. When `weights` is not `NULL`, a scaling transformation is applied such that the weights sum to the sample size:

$$w_{\text{new}} = n \times \frac{w_{\text{old}}}{\sum w_{\text{old}}}$$

Default `TRUE`. When `FALSE`, the raw weights are used without scaling.

`group` Character string. The name of the column in `data` identifying the technology group of each observation. The column is coerced to a factor internally and must have at least two unique values. When `groupType = "sfalcmcross"` and `group` is `NULL`, a single pooled latent class model is estimated and class assignments serve as groups (see section 'Details').

`S` Integer. Frontier orientation.

- `S = 1` (default): production or profit frontier, $\varepsilon_i = v_i - u_i$.
- `S = -1`: cost frontier, $\varepsilon_i = v_i + u_i$.

`udist` Character string. Distribution for the one-sided error term $u_i \geq 0$. The following distributions are available for `groupType = "sfacross"`:

- `"hnormal"` (default): half-normal distribution (Aigner *et al.*, 1977; Meeusen and van den Broeck, 1977).
- `"exponential"`: exponential distribution.
- `"tnormal"`: truncated normal distribution (Stevenson, 1980).
- `"rayleigh"`: Rayleigh distribution (Hajargasht, 2015).
- `"uniform"`: uniform distribution (Li, 1996; Nguyen, 2010).
- `"gamma"`: Gamma distribution, estimated by maximum simulated likelihood (Greene, 2003).
- `"lognormal"`: log-normal distribution, estimated by maximum simulated likelihood (Migon and Medici, 2001; Wang and Ye, 2020).
- `"weibull"`: Weibull distribution, estimated by maximum simulated likelihood (Tsionas, 2007).
- `"genexponential"`: generalised exponential distribution (Papadopoulos, 2020).
- `"tslaplace"`: truncated skewed Laplace distribution (Wang, 2012).

For `groupType = "sfaselectioncross"` and `"sfalcmcross"`, only `"hnormal"` is currently supported.

`start` Numeric vector. Optional starting values for the maximum likelihood (ML) or maximum simulated likelihood (MSL) estimation of the group-level frontier models. When `NULL` (default), starting values are computed automatically. For `groupType = "sfacross"`, they are derived from OLS residuals. For `groupType = "sfalcmcross"`, they depend on `whichStart`.

`scaling` Logical. Applicable only when `groupType = "sfacross"` and `udist = "tnormal"`. When `TRUE`, the scaling property model (Wang and Schmidt, 2002) is estimated, whereby $u_i = h(\mathbf{Z}_u, \boldsymbol{\delta}) u_i^*$ and u_i^* follows a truncated normal distribution $N^+(\tau, \exp(c_u))$. Default `FALSE`.

`modelType` Character string. Applicable only when `groupType = "sfaselectioncross"`. Specifies the model used to correct for selection bias. Currently, only `"greene10"` (default) is supported, corresponding to the two-step approach of Greene (2010): a probit model is estimated for the selection equation, and its inverse Mills ratio is included as a correction term in the stochastic frontier second step.

`groupType` Character string. Type of frontier model estimated for each technology group. Three options are available:

- `"sfacross"` (default): standard cross-sectional stochastic frontier analysis (`sfacross`). Groups are defined by the `group` variable. All 10 distributions for `udist` are supported, along with heteroscedasticity in both error components (`uhet`, `vhet`), heterogeneity in the truncated mean (`muhet`), and the scaling property.

- "sfaselectioncross": sample selection stochastic frontier analysis (sfaselectioncross). Corrects for sample selection bias via the generalised Heckman approach (Greene, 2010). Requires selectionF. Only observations for which the selection indicator equals one enter the frontier and metafrontier; efficiency estimates for non-selected observations are NA. Only udist = "hnormal" is supported.
- "sfalcmcross": latent class stochastic frontier analysis (sfalcmcross). Estimates a finite mixture of frontier models with the number of classes determined by lcmClasses. When group is supplied, a separate latent class model is estimated per group-stratum and combined for the metafrontier. When group is omitted, a single pooled model is estimated and class assignments serve as technology groups. Supports thet for class-membership covariates and uhet, vhet for within-class heteroscedasticity. Only udist = "hnormal" is supported.

metaMethod
: Character string. Method for estimating the global metafrontier that envelopes all group frontiers. Three options are available:

- "lp" (default): deterministic linear programming envelope. Finds the parameter vector $\boldsymbol{\beta}^*$ minimising $\sum_i |\ln \hat{f}(x_i, \boldsymbol{\beta}^*) - \ln \hat{f}(x_i, \hat{\boldsymbol{\beta}}_{(g)})|$ subject to $\ln \hat{f}(x_i, \boldsymbol{\beta}^*) \geq \ln \hat{f}(x_i, \hat{\boldsymbol{\beta}}_{(g)})$ for all observations and all groups (Battese *et al.*, 2004).
- "qp": deterministic quadratic programming envelope. Minimises the sum of squared deviations under the same envelope constraint.
- "sfa": stochastic metafrontier estimated by a second-stage pooled SFA. The specific construction of the dependent variable is determined by sfaApproach.

sfaApproach
: Character string. Applicable only when metaMethod = "sfa". Determines how the second-stage SFA is constructed:

- "ordonnell" (default): The LP envelope of the group frontier predicted values is re-estimated with a stochastic frontier, following O'Donnell, Rao, and Battese (2008). The second-stage SFA directly targets the global technology envelope.
- "huang": the group-specific fitted frontier value $\ln \hat{y}_i^g$ for each observation is used as the dependent variable in a pooled cross-sectional SFA (Huang, Huang, and Liu, 2014). The technology gap $U_i \geq 0$ and second-stage noise V_i are estimated directly by the SFA procedure.
- "ordonnell": the column-wise maximum of all group-fitted frontier values (the deterministic LP envelope) is used as the dependent variable in the second-stage SFA (O'Donnell, Rao, and Battese, 2008).

selectionF
: A two-sided formula specifying the sample selection equation, e.g., selected ~ z1 + z2. The left-hand side must be a binary (0/1) indicator already present in data: 1 means the observation participates in the frontier and metafrontier; 0 means it is excluded (efficiency estimates will be NA). Alternatively, a named list of formulas, one per group level, may be supplied to allow group-specific selection equations. Required when groupType = "sfaselectioncross"; ignored otherwise.

lcmClasses
: Integer. Number of latent classes to be estimated per group when groupType = "sfalcmcross". Must be between 2 and 5 (default 2). The optimal number of classes can be selected based on information criteria (see ic).

whichStart — Integer. Strategy for obtaining starting values in the latent class model (groupType = "sfalcmcross"):

- 1: starting values are obtained from the method of moments.
- 2 (default): the model is initialised by first solving a homoscedastic pooled cross-sectional SFA using the algorithm specified by initAlg for at most initIter iterations.

initAlg — Character string. Optimisation algorithm used during the initialisation of the latent class model when whichStart = 2. Only algorithms from the maxLik package are supported:

- "nm" (default): Nelder-Mead (see maxNM).
- "bfgs": Broyden-Fletcher-Goldfarb-Shanno (see maxBFGS).
- "bhhh": Berndt-Hall-Hall-Hausman (see maxBHHH).
- "nr": Newton-Raphson (see maxNR).
- "cg": Conjugate Gradient (see maxCG).
- "sann": Simulated Annealing (see maxSANN).

initIter — Integer. Maximum number of iterations for the initialisation algorithm when whichStart = 2 and groupType = "sfalcmcross". Default 100.

lType — Character string. Specifies how the likelihood is evaluated for the selection model (groupType = "sfaselectioncross"). Five options are available:

- "ghermite" (default): Gauss-Hermite quadrature (see gaussHermiteData).
- "kronrod": Gauss-Kronrod quadrature (see integrate).
- "hcubature": adaptive integration over hypercubes (see hcubature).
- "pcubature": p-adaptive cubature (see pcubature).
- "msl": maximum simulated likelihood (controlled by simType, Nsim, prime, burn, antithetics, and seed).

Nsub — Integer. Number of quadrature nodes or integration subdivisions when lType is "ghermite", "kronrod", "hcubature", or "pcubature". Applicable only when groupType = "sfaselectioncross". Default 100.

uBound — Numeric. Upper bound for the numerical integration of the inefficiency component when lType is "kronrod", "hcubature", or "pcubature". For Gauss-Hermite the bound is automatically infinite. Applicable only when groupType = "sfaselectioncross". Default Inf.

intol — Numeric. Integration tolerance for the quadrature approaches "kronrod", "hcubature", and "pcubature". Applicable only when groupType = "sfaselectioncross". Default 1e-6.

method — Character string. Optimisation algorithm for the main ML/MSL estimation of each group-level frontier model. Default "bfgs". Eleven algorithms are available:

- "bfgs": Broyden-Fletcher-Goldfarb-Shanno (see maxBFGS).
- "bhhh": Berndt-Hall-Hall-Hausman (see maxBHHH).
- "nr": Newton-Raphson (see maxNR).
- "nm": Nelder-Mead (see maxNM).
- "cg": Conjugate Gradient (see maxCG).

- "sann": Simulated Annealing (see maxSANN).
- "ucminf": quasi-Newton optimisation with BFGS updating of the inverse Hessian and soft line search (see ucminf).
- "mla": Marquardt-Levenberg algorithm (see mla).
- "sr1": Symmetric Rank 1 trust-region method (see trust.optim).
- "sparse": trust-region method with sparse Hessian (see trust.optim).
- "nlminb": PORT routines optimisation (see nlminb).

hessianType
: Integer. Specifies which Hessian is returned for the group-level frontier estimation. The accepted values match those of the underlying sfaR function for each groupType:
  - For groupType = "sfacross": if 1 (default), the analytic Hessian is returned; if 2, the BHHH Hessian $\mathbf{G}'\mathbf{G}$ is estimated.
  - For groupType = "sfalcmcross": if 1 (default), the analytic Hessian is returned; if 2, the BHHH Hessian is estimated.
  - For groupType = "sfaselectioncross": if 1, the analytic Hessian is returned; if 2 (default), the BHHH Hessian $\mathbf{G}'\mathbf{G}$ is estimated. The BHHH default reflects the two-step nature of the selection estimator.

  When NULL (the package default), each group-level model uses the natural default of the corresponding sfaR function, ensuring that standard errors computed by smfa are identical to those from a standalone sfaR call on the same group subset.

simType
: Character string. Simulation method for maximum simulated likelihood (MSL). Applicable to groupType = "sfacross" when udist is "gamma", "lognormal", or "weibull", and to groupType = "sfaselectioncross" when lType = "msl":
  - "halton" (default): Halton quasi-random sequences.
  - "ghalton": Generalised-Halton sequences.
  - "sobol": Sobol low-discrepancy sequences.
  - "uniform": pseudo-random uniform draws.

Nsim
: Integer. Number of simulation draws for MSL. Default 100.

prime
: Integer. Prime number used to construct Halton or Generalised-Halton sequences. Default 2.

burn
: Integer. Number of leading draws discarded from the Halton sequence to reduce serial correlation. Default 10.

antithetics
: Logical. If TRUE, antithetic draws are added: the first Nsim/2 draws are taken, and the remaining Nsim/2 are $1 - \text{draw}$. Default FALSE.

seed
: Integer. Random seed for simulation draws, ensuring reproducibility of MSL estimates. Default 12345.

itermax
: Integer. Maximum number of iterations for the main optimisation. Default 2000. For method = "sann", it is recommended to increase this substantially (e.g., itermax = 20000).

printInfo
: Logical. If TRUE, optimisation progress is printed during estimation of each group-level model. Default FALSE.

`tol`	Numeric. Convergence tolerance. The algorithm is considered converged when the change in the log-likelihood between successive iterations is smaller than `tol` in absolute value. Default `1e-12`.
`gradtol`	Numeric. Gradient convergence tolerance. The algorithm is considered converged when the Euclidean norm of the gradient is smaller than `gradtol`. Default `1e-6`.
`stepmax`	Numeric. Maximum step length used by the `"ucminf"` algorithm. Default `0.1`.
`qac`	Character string. Quadratic Approximation Correction for the `"bhhh"` and `"nr"` algorithms when the Hessian is not negative definite: • `"marquardt"` (default): step length is decreased while also shifting closer to the gradient direction. • `"stephalving"`: step length is halved, preserving the current direction. See `maxBHHH` and `maxNR` for details.
`...`	Additional arguments passed through to the second-stage SFA call when `metaMethod = "sfa"`.
`x`	An object of class `"smfa"`, as returned by `smfa`, for use with the `print` method.

Details

Standard stochastic frontier (`groupType = "sfacross"`)**:** The stochastic frontier model is defined as:

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

where y is the output (cost, revenue, or profit), $\mathbf{x}$ is the vector of frontier regressors, $u_i \geq 0$ is the one-sided inefficiency term with variance σ_u^2, and v_i is the symmetric noise term with variance σ_v^2.

Estimation is by ML for all distributions except `"gamma"`, `"lognormal"`, and `"weibull"`, for which MSL is used with Halton, Generalised-Halton, Sobol, or uniform draws. Antithetic draws are available for the uniform case.

To account for heteroscedasticity, the variances are modelled as $\sigma_u^2 = \exp(\boldsymbol{\delta}'\mathbf{Z}_u)$ and $\sigma_v^2 = \exp(\boldsymbol{\phi}'\mathbf{Z}_v)$. For the truncated normal distribution, heterogeneity in the pre-truncation mean is modelled as $\mu_i = \boldsymbol{\omega}'\mathbf{Z}_\mu$. The scaling property (Wang and Schmidt, 2002) can also be imposed for the truncated normal.

Sample selection frontier (`groupType = "sfaselectioncross"`)**:** This model extends the Heckman (1979) selection framework to the stochastic frontier setting (Greene, 2010; Dakpo *et al.*, 2021). The selection and frontier equations are:

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

$$y_{2i}^* = \mathbf{x}_i'\boldsymbol{\beta} + v_i - Su_i$$

where $y_{1i} = \mathbf{1}(y_{1i}^* > 0)$ is the binary selection indicator and $y_{2i} = y_{2i}^*$ is observed only when $y_{1i} = 1$. Selection bias arises from $\rho = \mathrm{Corr}(w_i, v_i) \neq 0$. Only selected observations enter the frontier and metafrontier estimation; efficiency estimates for non-selected observations are `NA`.

Latent class frontier (`groupType = "sfalcmcross"`)**:** The latent class model (Orea and Kumbhakar, 2004) estimates a finite mixture of J frontier models:

$$y_i = \alpha_j + \mathbf{x}_i'\boldsymbol{\beta}_j + v_{i|j} - Su_{i|j}$$

The prior class probability follows a logit specification:

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

Class assignment is based on the maximum posterior probability computed via Bayes' rule. When `group` is omitted, a single pooled model is estimated and class assignments serve as technology groups.

Metafrontier estimation: The global metafrontier $f(x_i, \boldsymbol{\beta}^*)$ envelopes all group frontiers. With LP (Battese *et al.*, 2004), $\boldsymbol{\beta}^*$ minimises $\sum_i |\ln\hat{f}(x_i,\boldsymbol{\beta}^*) - \ln\hat{f}(x_i,\hat{\boldsymbol{\beta}}_{(g)})|$ subject to $\ln\hat{f}(x_i,\boldsymbol{\beta}^*) \geq \ln\hat{f}(x_i,\hat{\boldsymbol{\beta}}_{(g)})$. QP minimises the squared analogue. The stochastic approaches (Huang *et al.*, 2014; O'Donnell *et al.*, 2008) treat the technology gap U_i as a one-sided error in a second-stage SFA. Group and metafrontier efficiencies are:

$$TE_i^g = \exp(-u_i^g), \quad MTR_i = \exp(-U_i), \quad TE_i^* = TE_i^g \times MTR_i$$

Both Jondrow *et al.* (1982) and Battese and Coelli (1988) estimators are provided for each measure. See `efficiencies` for details.

Value

`smfa` returns an object of class `"smfa"`, which is a list containing:

- `call` — The matched call.
- `groupModels` — A named list of fitted group-level frontier objects, one per technology group. Each element is of class `"sfacross"`, `"sfaselectioncross"`, or `"sfalcmcross"`, depending on `groupType`.
- `metaSfaObj` — The fitted metafrontier object. For `metaMethod = "sfa"`, an object of class `"sfacross"` from the second-stage SFA. The dependent variable column in `metaSfaObj$dataTable` is named according to the approach used: `"lp_envelope"` when `sfaApproach = "ordonnell"` (the column-wise maximum of all group-evaluated frontier values is the dependent variable) and `"group_fitted_values"` when `sfaApproach = "huang"` (each observation's own-group fitted frontier value is the dependent variable). For `metaMethod = "lp"` or `"qp"`, a list containing the optimisation result and the estimated envelope coefficients.
- `metaRes` — Estimated metafrontier coefficients (with standard errors, z-values, and p-values for `metaMethod = "sfa"`, or the plain coefficient vector for deterministic envelopes).
- `formula` — The `formula` supplied to the call.
- `metaMethod` — The metafrontier estimation method used.
- `sfaApproach` — The second-stage SFA approach; `NA` when `metaMethod` is not `"sfa"`.
- `groupType` — The type of group-level frontier model estimated.

`group`	The name of the grouping variable.
`groups`	Character vector of unique group labels.
`S`	The frontier orientation (`1` or `-1`).
`dataTable`	The data used in estimation, augmented with `.mf_yhat_group` (group-specific fitted frontier values) and `.mf_yhat_meta` (metafrontier fitted values).
`lcmNoGroup`	Logical. `TRUE` when `groupType = "sfalcmcross"` and `group` was not supplied.
`lcmObj`	When `lcmNoGroup = TRUE`, the pooled `sfalcmcross` object.

References

Aigner, D. J., Lovell, C. A. K., and Schmidt, P. 1977. Formulation and estimation of stochastic frontier production function models. *Journal of Econometrics*, **6**(1), 21–37. doi:10.1016/0304-4076(77)900525

Battese, G. E., and Coelli, T. J. 1988. Prediction of firm-level technical efficiencies with a generalized frontier production function and panel data. *Journal of Econometrics*, **38**(3), 387–399. doi:10.1016/03044076(88)90053X

Battese, G. E., Rao, D. S. P., and O'Donnell, C. J. 2004. A metafrontier production function for estimation of technical efficiencies and technology gaps for firms operating under different technologies. *Journal of Productivity Analysis*, **21**(1), 91–103. doi:10.1023/B:PROD.0000012454.06094.29

Greene, W. 2003. Simulated likelihood estimation of the normal-gamma stochastic frontier function. *Journal of Productivity Analysis*, **19**(2-3), 179–190. doi:10.1023/A:1022853416499

Greene, W. 2010. A stochastic frontier model with correction for sample selection. *Journal of Productivity Analysis*, **34**(1), 15–24. doi:10.1007/s1112300901591

Hajargasht, G. 2015. Stochastic frontiers with a Rayleigh distribution. *Journal of Productivity Analysis*, **44**(2), 199–208. doi:10.1007/s1112301404178

Heckman, J. J. 1979. Sample selection bias as a specification error. *Econometrica*, **47**(1), 153–161. doi:10.2307/1912352

Huang, C. J., Huang, T.-H., and Liu, N.-H. 2014. A new approach to estimating the metafrontier production function based on a stochastic frontier framework. *Journal of Productivity Analysis*, **42**(3), 241–254. doi:10.1007/s1112301404022

Jondrow, J., Lovell, C. A. K., Materov, I. S., and Schmidt, P. 1982. On the estimation of technical inefficiency in the stochastic frontier production function model. *Journal of Econometrics*, **19**(2-3), 233–238. doi:10.1016/03044076(82)900045

Li, Q. 1996. Estimating a stochastic production frontier when the adjusted error is symmetric. *Economics Letters*, **52**(3), 221–228. doi:10.1016/S01651765(96)008579

Meeusen, W., and van den Broeck, J. 1977. Efficiency estimation from Cobb-Douglas production functions with composed error. *International Economic Review*, **18**(2), 435–444. doi:10.2307/2525757

Migon, H. S., and Medici, E. 2001. Bayesian inference for generalised exponential models. Working paper, Universidade Federal do Rio de Janeiro.

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

O'Donnell, C. J., Rao, D. S. P., and Battese, G. E. 2008. Metafrontier frameworks for the study of firm-level efficiencies and technology ratios. *Empirical Economics*, **34**(2), 231–255. doi:10.1007/s0018100701194

Orea, L., and Kumbhakar, S. C. 2004. Efficiency measurement using a latent class stochastic frontier model. *Empirical Economics*, **29**(1), 169–183. doi:10.1007/s0018100301842

Dakpo, K. H., Jeanneaux, P., and Latruffe, L. 2016. Modelling pollution-generating technologies in performance benchmarking: Recent developments, limits and future prospects in the nonparametric framework. *European Journal of Operational Research*, **250**(2), 347–359. doi:10.1016/j.ejor.2015.07.024

Papadopoulos, A. 2015. The half-normal specification for the two-tier stochastic frontier model. *Journal of Productivity Analysis*, **43**(2), 225–230. doi:10.1007/s1112301403898

Stevenson, R. E. 1980. Likelihood functions for generalised stochastic frontier estimation. *Journal of Econometrics*, **13**(1), 57–66. doi:10.1016/03044076(80)900421

Dakpo, K. H., Latruffe, L., Desjeux, Y., and Jeanneaux, P. 2021. Latent class modelling for a robust assessment of productivity: Application to French grazing livestock farms. *Journal of Agricultural Economics*, **72**(3), 760–781. doi:10.1111/14779552.12422

Dakpo, K. H., Latruffe, L., Desjeux, Y., and 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. doi:10.1111/agec.12683

Tsionas, E. G. 2007. Efficiency measurement with the Weibull stochastic frontier. *Oxford Bulletin of Economics and Statistics*, **69**(5), 693–706. doi:10.1111/j.14680084.2007.00475.x

Wang, H.-J. 2012. Stochastic frontier models. In *A Companion to Theoretical Econometrics*, ed. B. H. Baltagi, Blackwell, Oxford.

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**(2), 129–144. doi:10.1023/A:1016565719882

Dakpo, K. H., Desjeux, Y., and Latruffe, L. 2023. sfaR: Stochastic Frontier Analysis using R. R package version 1.0.1. https://CRAN.R-project.org/package=sfaR

See Also

sfacross, sfaselectioncross, sfalcmcross, efficiencies, summary.smfa, ic

Examples

```
###########################################################################
## -------- SECTION 1: Standard SFA Group Frontier ----------------------##
## Using the rice production dataset (ricephil) from Battese et al.      ##
## Groups are formed based on farm area terciles (small/medium/large).   ##
###########################################################################

data("ricephil", package = "sfaR")
ricephil$group <- cut(ricephil$AREA,
  breaks = quantile(ricephil$AREA, probs = c(0, 1 / 3, 2 / 3, 1), na.rm = TRUE),
  labels = c("small", "medium", "large"),
  include.lowest = TRUE
)
```

```
## 1a. sfacross groups + LP metafrontier
##     Deterministic envelope via linear programming (Battese et al., 2004).
meta_sfacross_lp <- smfa(
  formula    = log(PROD) ~ log(AREA) + log(LABOR) + log(NPK),
  data       = ricephil,
  group      = "group",
  S          = 1,
  udist      = "hnormal",
  groupType  = "sfacross",
  metaMethod = "lp"
)
summary(meta_sfacross_lp)
# Retrieve individual efficiency and metatechnology ratio estimates:
ef_lp <- efficiencies(meta_sfacross_lp)
head(ef_lp)

## 1b. sfacross groups + QP metafrontier
##     Deterministic envelope via quadratic programming.
meta_sfacross_qp <- smfa(
  formula    = log(PROD) ~ log(AREA) + log(LABOR) + log(NPK),
  data       = ricephil,
  group      = "group",
  S          = 1,
  udist      = "hnormal",
  groupType  = "sfacross",
  metaMethod = "qp"
)
summary(meta_sfacross_qp)


## 1c. sfacross groups + Two-stage SFA metafrontier (Huang et al., 2014)
##     The group-specific fitted frontier values serve as the dependent
##     variable in the second-stage SFA, yielding a stochastic technology gap.
meta_sfacross_huang <- smfa(
  formula     = log(PROD) ~ log(AREA) + log(LABOR) + log(NPK),
  data        = ricephil,
  group       = "group",
  S           = 1,
  udist       = "hnormal",
  groupType   = "sfacross",
  metaMethod  = "sfa",
  sfaApproach = "huang"
)
summary(meta_sfacross_huang)
ef_huang <- efficiencies(meta_sfacross_huang)
head(ef_huang)

## 1d. sfacross groups + O'Donnell et al. (2008) stochastic metafrontier
##     The LP deterministic envelope is used as the second-stage dependent
##     variable: the metafrontier is estimated stochastically around the
##     envelope.
meta_sfacross_odonnell <- smfa(
```

```
  formula     = log(PROD) ~ log(AREA) + log(LABOR) + log(NPK),
  data        = ricephil,
  group       = "group",
  S           = 1,
  udist       = "hnormal",
  groupType   = "sfacross",
  metaMethod  = "sfa",
  sfaApproach = "ordonnell"
)
summary(meta_sfacross_odonnell)


############################################################################
## -------- SECTION 2: Latent Class (LCM) Group Frontier ---------------##
## No observed group variable: a pooled sfalcmcross model assigns       ##
## observations to 2 latent technology classes; these classes become the ##
## technology groups for the metafrontier.                               ##
############################################################################

data("utility", package = "sfaR")

## 2a. sfalcmcross (pooled, 2 classes) + LP metafrontier
meta_lcm_lp <- smfa(
  formula    = log(tc / wf) ~ log(y) + log(wl / wf) + log(wk / wf),
  data       = utility,
  S          = -1,
  groupType  = "sfalcmcross",
  lcmClasses = 2,
  metaMethod = "lp"
)
summary(meta_lcm_lp)
ef_lcm_lp <- efficiencies(meta_lcm_lp)
head(ef_lcm_lp)


## 2b. sfalcmcross (pooled, 2 classes) + QP metafrontier
meta_lcm_qp <- smfa(
  formula    = log(tc / wf) ~ log(y) + log(wl / wf) + log(wk / wf),
  data       = utility,
  S          = -1,
  groupType  = "sfalcmcross",
  lcmClasses = 2,
  metaMethod = "qp"
)
summary(meta_lcm_qp)

## 2c. sfalcmcross (pooled, 2 classes) + Two-stage SFA metafrontier
##     (Huang et al., 2014)
meta_lcm_huang <- smfa(
  formula     = log(tc / wf) ~ log(y) + log(wl / wf) + log(wk / wf),
  data        = utility,
  S           = -1,
  groupType   = "sfalcmcross",
```

```
  lcmClasses  = 2,
  metaMethod  = "sfa",
  sfaApproach = "huang"
)
summary(meta_lcm_huang)
ef_lcm_huang <- efficiencies(meta_lcm_huang)
head(ef_lcm_huang)

## 2d. sfalcmcross (pooled, 2 classes) + O'Donnell et al. (2008)
meta_lcm_odonnell <- smfa(
  formula     = log(tc / wf) ~ log(y) + log(wl / wf) + log(wk / wf),
  data        = utility,
  S           = -1,
  groupType   = "sfalcmcross",
  lcmClasses  = 2,
  metaMethod  = "sfa",
  sfaApproach = "ordonnell"
)
summary(meta_lcm_odonnell)


###########################################################################
## -------- SECTION 3: Sample Selection SFA Group Frontier -------------##
###########################################################################

## 3a. Small toy example for automatic testing (< 5s)
N <- 100
set.seed(12345)
z1 <- rnorm(N); v1 <- rnorm(N); g <- rnorm(N)
ds <- z1 + v1; d <- ifelse(ds > 0, 1, 0)
group <- ifelse(g > 0, 1, 0)
x1 <- rnorm(N); y <- x1 + rnorm(N) - abs(rnorm(N))
dat <- data.frame(y = y, x1 = x1, z1 = z1, d = d, group = group)

meta_toy <- smfa(
  formula    = y ~ x1,
  selectionF = d ~ z1,
  data       = dat,
  group      = "group",
  groupType  = "sfaselectioncross",
  lType      = "ghermite",
  Nsub       = 10,
  itermax    = 100,
  metaMethod = "lp"
)
summary(meta_toy)


## 3b. More complex selection models
## Simulated dataset with a Heckman selection mechanism.

N <- 2000
set.seed(12345)
```

```
z1 <- rnorm(N); z2 <- rnorm(N); v1 <- rnorm(N); v2 <- rnorm(N); g <- rnorm(N)
e1 <- v1; e2 <- 0.7071 * (v1 + v2)
ds <- z1 + z2 + e1; d <- ifelse(ds > 0, 1, 0)
group <- ifelse(g > 0, 1, 0)
u <- abs(rnorm(N)); x1 <- rnorm(N); x2 <- rnorm(N)
y <- x1 + x2 + e2 - u
dat <- data.frame(y = y, x1 = x1, x2 = x2, z1 = z1, z2 = z2, d = d, group = group)

meta_sel_lp <- smfa(
  formula    = y ~ x1 + x2,
  selectionF = d ~ z1 + z2,
  data       = dat,
  group      = "group",
  S          = 1L,
  udist      = "hnormal",
  groupType  = "sfaselectioncross",
  modelType  = "greene10",
  lType      = "kronrod",
  Nsub       = 100,
  metaMethod = "lp"
)
summary(meta_sel_lp)
```

summary — *Summary of results for stochastic metafrontier models*

Description

Create and print summary results for stochastic metafrontier models returned by smfa.

Usage

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

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

Arguments

object	An object of class 'smfa' returned by the function smfa.
...	Currently ignored.
x	An object of class 'summary.smfa'.
digits	Numeric. Number of digits displayed in values.

Value

The `summary` method returns a list of class `'summary.smfa'` that contains the same elements as an object returned by `smfa` with the following additional elements:

`AIC`	Akaike information criterion.
`BIC`	Bayesian information criterion.
`HQIC`	Hannan-Quinn information criterion.
`metaRes`	Matrix of metafrontier estimates, their standard errors, z-values, and asymptotic P-values.
`effStats`	A list of efficiency statistics including group means and class membership probabilities.
`grpSummaries`	A list of summary objects for each group model.

See Also

`smfa`, for the stochastic metafrontier analysis model fitting function for cross-sectional or pooled data.

`vcov` — *Compute variance-covariance matrix of stochastic metafrontier models*

Description

`vcov` computes the variance-covariance matrix of the maximum likelihood (ML) coefficients from stochastic metafrontier models estimated with `smfa`.

Usage

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

Arguments

`object`	A stochastic metafrontier model returned by `smfa`.
`...`	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.

Value

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

See Also

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

Index