Package 'snreg'

January 26, 2026

Type Package

Title Regression with Skew-Normally Distributed Error Term

Version 1.1.0.9000

Date 2021-06-19

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

Description Models with a skew-normally distributed error term: Badunenko & Henderson (2023). Production analysis with asymmetric noise. Journal of Productivity Analysis, https://doi.org/10.1007/s11123-023-00680-5.

URL https://olegbadunenko.github.io/snreg/

Depends Formula, npsf

License GPL-3

Encoding UTF-8

LazyData TRUE

RoxygenNote 7.3.3

Suggests knitr, rmarkdown

VignetteBuilder knitr

Contents

banks07 *U.S. Commercial Banks Data*

Description

banks07 is a data frame containing selected variables for 500 U.S. commercial banks, randomly sampled from approximately 5000 banks, based on the dataset of Koetter et al. (2012) for year 2007. The dataset is provided solely for illustration and pedagogical purposes and is not suitable for empirical research.

Usage

```
data(banks07)
```

Format

A data frame with the following variables:

`year` Year (2007).

`id` Entity (bank) identifier.

`TA` Gross total assets.

`LLP` Loan loss provisions.

`Y1` Total securities (thousands of USD).

`Y2` Total loans and leases (thousands of USD).

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

`W2` Cost of labor (thousands of USD) divided by the cost of borrowed funds.

`ER` Equity-to-assets ratio (gross).

`TC` Total operating cost.

`LA` Ratio of total loans and leases to gross total assets.

Details

U.S. Commercial Banks Data (2007)

The dataset was created by sampling and transforming variables as shown in the section **Examples**. It is intended to illustrate the usage of functions from this package (e.g. stochastic frontier models with skew-normal noise).

Source

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

References

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

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

Examples

```
## Not run:

## ------------------------------------------------------------------
## Construct sample panel dataset (banks00_07)
## ------------------------------------------------------------------

# Download data from the link in "Source"
banks00_07 <- read.delim("2b_QLH.txt")

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

# keep only years 2000-2007
banks00_07 <- banks00_07[
  banks00_07$year >= 2000 & banks00_07$year <= 2007, ]

# restrict sample to interquartile range of total assets
q1q3 <- quantile(banks00_07$TA, probs = c(.25, .75))
banks00_07 <- banks00_07[
  banks00_07$TA >= q1q3[1] & banks00_07$TA <= q1q3[2], ]

# generate required variables
banks00_07$TC <- banks00_07$TOC
banks00_07$ER <- banks00_07$Z  / banks00_07$TA   # Equity ratio
banks00_07$LA <- banks00_07$Y2 / banks00_07$TA   # Loans-to-assets ratio

# keep only needed variables
keep.vars <- c("id", "year", "Ti", "TC", "Y1", "Y2", "W1","W2",
               "ER", "LA", "TA", "LLP")
banks00_07 <- banks00_07[, colnames(banks00_07) %in% keep.vars]

# number of periods per id
t0 <- as.vector( by(banks00_07$id, banks00_07$id,
                    FUN = function(qq) length(qq)) )
banks00_07$Ti <- rep(t0, times = t0)

# keep if Ti > 4
banks00_07 <- banks00_07[banks00_07$Ti > 4, ]

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

# sample 500 banks at random
set.seed(816376586)
id_names <- unique(banks00_07$id)
ids2choose <- sample(id_names, 500)
banks00_07 <- banks00_07[banks00_07$id %in% ids2choose, ]

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

# sort
```

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


banks07 <- banks00_07[banks00_07$year == 2007, ]


## End(Not run)
```

`coef.snreg` *Extract Model Coefficients*

Description

`coef.snreg` is the S3 method for extracting the estimated regression coefficients from an object of class `"snreg"`.

Usage

```
## S3 method for class 'snreg'
coef(obj, ...)
```

Arguments

`obj`	an object of class `"snreg"`, typically returned by `snreg`.
`...`	additional arguments (currently unused).

Details

Coefficients from an snreg Model

This method simply returns the `coef` component stored inside the fitted `"snreg"` object. If the object does not contain coefficient estimates (e.g., if estimation was not completed in a scaffold), an informative error is raised.

Value

A numeric vector containing the model coefficients.

Examples

```
## Not run:
  m <- snreg(y ~ x1 + x2, data = df)
  coef(m)

## End(Not run)
```

`lm.mle` *Linear Regression via MLE*

Description

`lm.mle` fits a linear regression model by maximum likelihood, allowing for optional multiplicative heteroskedasticity in the disturbance variance via a log-linear specification provided through `ln.var.v`.

Usage

```
lm.mle(
  formula,
  data,
  subset,
  ln.var.v = NULL,
  technique = c("bfgs"),
  lmtol = 1e-05,
  reltol = 1e-12,
  maxit = 199,
  optim.report = 1,
  optim.trace = 10,
  print.level = 3,
  digits = 4,
  only.data = FALSE,
  ...
)
```

Arguments

`formula`	an object of class `formula` specifying the regression: typically `y ~ x1 + ...`, where `y` is the dependent variable and the `x`'s are regressors.
`data`	an optional `data.frame` containing the variables referenced in `formula`. If not found in `data`, variables are taken from `environment(formula)`.
`subset`	an optional logical or numeric vector specifying the subset of observations to be used in estimation.
`ln.var.v`	optional one-sided formula; e.g. `ln.var.v ~ z1 + z2`. When provided, the error variance is modeled as $\log(\sigma_i^2) = w_i^\top \gamma_v$. If `NULL`, the variance is homoskedastic.
`technique`	character vector specifying the preferred optimization routine(s) in order of preference. Recognized keywords (for future implementation) include `"bfgs"` `"bhhh"`, `"nm"` (Nelder–Mead), `"bfgs"`, and `"cg"`. Default is `"bfgs"`. This scaffold records but does not execute the chosen routine.
`lmtol`	numeric. Convergence tolerance based on scaled gradient (when applicable). Default `1e-5`.
`reltol`	numeric. Relative convergence tolerance for likelihood maximization. Default `1e-12`.
`maxit`	integer. Maximum number of iterations for the optimizer. Default `199`.
`optim.report`	integer. Verbosity level for reporting progress (if implemented). Default `1`.

optim.trace	integer. Trace level for optimization (if implemented). Default 1.
print.level	integer. Printing level for summaries. Default 3.
digits	integer. Number of digits for printing. Default 4.
only.data	logical. If TRUE, returns only constructed data/matrices without estimation. Default FALSE.
...	additional arguments reserved for future methods (e.g., bounds, penalties).

Details

Linear Model by Maximum Likelihood (with optional heteroskedasticity)

The model is

$$y_i = x_i^\top \beta + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma_i^2).$$

When ln.var.v is supplied, the variance follows

$$\log(\sigma_i^2) = w_i^\top \gamma_v,$$

otherwise $\sigma_i^2 = \sigma^2$ is constant (homoskedastic).

This function:

- Builds the model frame and X, y.
- Builds Zv for the log-variance index when ln.var.v is provided.
- Returns a structured object with placeholders for coef, vcov, loglik.

Insert your MLE engine to estimate β, and (optionally) σ^2 or γ_v; compute standard errors via AIM/OPG as required by vcetype.

Value

A list of class "snreg" containing (and extending) the fields returned by optim:

- par — numeric vector of the MLE parameter estimates.
- value — numeric scalar: maximized log-likelihood value.
- ll — numeric scalar: maximized log-likelihood value.
- counts, convergence, message — standard optim outputs.
- hessian — the observed Hessian at the solution (as returned by optim(hessian=TRUE)).
- coef — named numeric vector equal to par (estimates).
- vcov — variance–covariance matrix, computed as solve(-hessian).
- sds — standard errors, sqrt(diag(vcov)).
- ctab — coefficient table with columns: Estimate, Std.Err, Z value, Pr(>z).
- esample — logical vector indicating the observations used in estimation.
- n — scalar, number of observations in the estimation sample.

The object inherits the default optim components and is assigned class "snreg".

Examples

```
## Not run:

library(snreg)

data("banks07")
head(banks07)

# Translog cost function specification
spe.tl <- log(TC) ~ (log(Y1) + log(Y2) + log(W1) + log(W2))^2 +
  I(0.5 * log(Y1)^2) + I(0.5 * log(Y2)^2) +
  I(0.5 * log(W1)^2) + I(0.5 * log(W2)^2)

# ------------------------------------------------------------
# Specification 1: homoskedastic noise (ln.var.v = NULL)
# ------------------------------------------------------------
formSV <- NULL

m1 <- lm.mle(
  formula   = spe.tl,
  data      = banks07,
  ln.var.v  = formSV
)

coef(m1)


# ------------------------------------------------------------
# Specification 2: heteroskedastic noise (variance depends on TA)
# ------------------------------------------------------------
formSV <- ~ log(TA)

m2 <- lm.mle(
  formula   = spe.tl,
  data      = banks07,
  ln.var.v  = formSV
)

coef(m2)


## End(Not run)
```

`print.summary.snreg` *Print Summary of snreg Results*

Description

Prints the contents of a `"summary.snreg"` object in a structured format. The method reports convergence status (based on gradient-Hessian scaling), log-likelihood, estimation results, and—when present—summaries for technical/cost efficiencies and marginal effects.

Usage

```
## S3 method for class 'summary.snreg'
print(obj, digits = NULL, ...)
```

Arguments

obj	an object of class "summary.snreg" (produced by summary.snreg).
digits	integer indicating the number of digits to print; default NULL (internally set to 4).
...	additional arguments (currently unused).

Details

Print Method for Summary of snreg Objects

This method expects a fitted "snreg" object.

Value

The input obj is returned (invisibly) after printing.

See Also

summary.snreg

---

residuals.snreg *Residuals for snreg Objects*

Description

residuals.snreg is the S3 method for extracting residuals from a fitted snreg model. Residuals may be returned either for the full data or only for the estimation sample.

Usage

```
## S3 method for class 'snreg'
residuals(obj, esample = TRUE, ...)
```

Arguments

obj	an object of class "snreg", typically produced by snreg.
esample	logical. If TRUE (default), residuals are returned only for observations used in estimation (others are NA). If FALSE, the raw vector of residuals (obj$resid) is returned.
...	additional arguments (currently unused).

Details

Extract Residuals from an snreg Model

This method simply accesses the obj$resid component of a fitted "snreg" object. An informative error is produced if residuals are not available.

Value

A numeric vector of residuals. If `esample = TRUE`, the vector matches the length of the original data and contains `NA` for non-estimation observations. If `esample = FALSE`, only the computed residuals are returned.

See Also

`snreg`, `fitted.snreg`, `coef.snreg`

Examples

```
## Not run:
  m <- snreg(y ~ x1 + x2, data = df)

  # Residuals for estimation sample only
  residuals(m)

  # Residuals for all observations
  residuals(m, esample = FALSE)

## End(Not run)
```

`snreg` *Linear Regression with Skew-Normal Errors*

Description

`snreg` fits a linear regression model where the disturbance term follows a skew-normal distribution. The function supports multiplicative heteroskedasticity of the noise variance via a log-linear specification (`ln.var.v`) and allows the skewness parameter to vary linearly with exogenous variables (`skew.v`).

Usage

```
snreg(
  formula,
  data,
  subset,
  init.sk = NULL,
  ln.var.v = NULL,
  skew.v = NULL,
  start.val = NULL,
  technique = c("nr"),
  vcetype = c("aim"),
  lmtol = 1e-05,
  reltol = 1e-12,
  maxit = 199,
  optim.report = 1,
  optim.trace = 1,
  print.level = 3,
  digits = 4,
```

```
  only.data = FALSE,
  ...
)
```

Arguments

formula	an object of class formula specifying the regression: typically y ~ x1 + ..., where y is the dependent variable and x's are regressors.
data	an optional data.frame containing the variables in formula. If not found in data, variables are taken from environment(formula).
subset	an optional logical or numeric vector specifying the subset of observations to be used in estimation.
init.sk	numeric. Initial value for the (global) skewness parameter of the noise; can be NULL if skew.v is supplied with its own coefficients to initialize.
ln.var.v	optional one-sided formula; e.g. ln.var.v ~ z1 + z2. Specifies exogenous variables entering the (log) variance of the random noise component. If NULL, the noise variance is homoskedastic.
skew.v	optional one-sided formula; e.g. skew.v ~ z3 + z4. Specifies exogenous variables determining the skewness of the noise via a linear index; if NULL, the skewness is constant (scalar).
start.val	optional numeric vector of starting values for all free parameters (regression coefficients, variance/heteroskedasticity parameters, skewness parameters).
technique	character vector giving the preferred maximization routine(s) in order of preference. Currently recognized keywords include "nr" (Newton–Raphson), "bhhh", "nm" (Nelder–Mead), "bfgs", "cg". This scaffold does not implement them yet, but records the choice.
vcetype	character specifying the variance-covariance estimator type: "aim" for the approximated information matrix or "opg" for the outer product of gradients. Default is "aim".
lmtol	numeric. Convergence tolerance based on the scaled gradient (if applicable). Default is 1e-5.
reltol	numeric. Relative convergence tolerance for likelihood maximization. Default is 1e-12.
maxit	integer. Maximum number of iterations for the optimizer. Default is 199.
optim.report	integer. Verbosity for reporting progress (if implemented). Default is 1.
optim.trace	integer. If positive, tracing information is printed (if implemented). Default is 1.
print.level	integer. Printing level for summaries: 1—print estimation results; 2—print optimization details; 3—print compact summary. Default 3.
digits	integer. Number of digits for printing. Default 4.
only.data	logical. If TRUE, the function returns only the constructed model matrices and design sets (no estimation). Default FALSE.
...	additional arguments reserved for future methods (e.g., box constraints).

Details

Linear Regression with Skew-Normal Errors

The model is

$$y_i = x_i^\top \beta + \varepsilon_i, \quad \varepsilon_i \sim SN(0, \sigma_i^2, \alpha_i),$$

where SN denotes the skew-normal distribution (Azzalini).

Heteroskedasticity in the noise variance (if specified via `ln.var.v`) is modeled as

$$\log(\sigma_i^2) = w_i^\top \gamma_v,$$

and the (optional) covariate-driven skewness (if specified via `skew.v`) as

$$\alpha_i = s_i^\top \delta.$$

This function constructs the model frame and design matrices for β, γ_v, and δ, and is designed to be paired with a maximum likelihood routine to estimate parameters and (optionally) their asymptotic covariance via either AIM or OPG.

Value

An object of class `"snreg"` containing the maximum-likelihood results and, depending on the optimization routine, additional diagnostics:

`par` Numeric vector of parameter estimates at the optimum.

`coef` Named numeric vector equal to `par`.

`vcov` Variance–covariance matrix of the estimates.

`sds` Standard errors, computed as `sqrt(diag(vcov))`.

`ctab` Coefficient table with columns: `Estimate`, `Std.Err`, `Z value`, `Pr(>z)`.

`RSS` Residual sum of squares.

`esample` Logical vector indicating which observations were used in estimation.

`n` Number of observations used in the estimation sample.

`skewness` Vector of the fitted skewness index.

`hessian` (BFGS only) Observed Hessian at the optimum. If `vcetype == "opg"`, this is set to the negative outer product of the individual gradients; otherwise a numerical Hessian is computed.

`value` (BFGS only) Objective value returned by `optim`. With `control$fnscale = -1`, this equals the maximized log-likelihood.

`counts` (BFGS only) Number of iterations / function evaluations returned by `optim`.

`convergence` (BFGS only) Convergence code from `optim`.

`message` (BFGS only) Additional `optim` message, if any.

`ll` Maximized log-likelihood value.

`gradient` (NR only) Gradient at the solution.

`gg` (NR only) Optional gradient-related diagnostic.

`gHg` (NR only) Optional Newton-step diagnostic.

`theta_rel_ch` (NR only) Relative parameter change metric across iterations.

The returned object has class `"snreg"`.

References

Azzalini, A. (1985). *A Class of Distributions Which Includes the Normal Ones*. Scandinavian Journal of Statistics, 12(2), 171–178.

Azzalini, A., & Capitanio, A. (2014). *The Skew-Normal and Related Families*. Cambridge University Press.

Examples

```
## Not run:

library(snreg)

data("banks07")
head(banks07)

# Translog cost function
spe.tl <- log(TC) ~ (log(Y1) + log(Y2) + log(W1) + log(W2))^2 +
  I(0.5 * log(Y1)^2) + I(0.5 * log(Y2)^2) +
  I(0.5 * log(W1)^2) + I(0.5 * log(W2)^2)


# ------------------------------------------------------------
# Specification 1: homoskedastic & symmetric noise
# ------------------------------------------------------------
formSV <- NULL     # variance equation
formSK <- NULL     # skewness equation

m1 <- snreg(
  formula  = spe.tl,
  data     = banks07,
  ln.var.v = formSV,
  skew.v   = formSK
)

coef(m1)


# ------------------------------------------------------------
# Specification 2: heteroskedastic + skewed noise
# ------------------------------------------------------------
formSV <- ~ log(TA)   # heteroskedasticity in v
formSK <- ~ ER        # skewness driven by equity ratio

m2 <- snreg(
  formula  = spe.tl,
  data     = banks07,
  ln.var.v = formSV,
  skew.v   = formSK
)

coef(m2)


## End(Not run)
```

---

snsf *Stochastic Frontier Model with a Skew-Normally Distributed Error Term*

Description

snsf performs maximum likelihood estimation of the parameters and technical or cost efficiencies in a Stochastic Frontier Model with a skew-normally distributed error term.

Usage

```
snsf(
  formula,
  data,
  subset,
  distribution = "e",
  prod = TRUE,
  start.val = NULL,
  init.sk = NULL,
  ln.var.u = NULL,
  ln.var.v = NULL,
  skew.v = NULL,
  mean.u = NULL,
  technique = c("nr"),
  vcetype = c("aim"),
  optim.report = 1,
  optim.trace = 1,
  reltol = 1e-12,
  lmtol = 1e-05,
  maxit = 199,
  print.level = 3,
  report = 1,
  trace = 1,
  threads = 1,
  only.data = FALSE,
  digits = 4,
  ...
)
```

Arguments

formula	an object of class formula specifying the frontier: a typical model is y ~ x1 + ..., where y is the log of output (or total cost), and x's are inputs (or outputs and input prices, in logs). See **Details**.
data	an optional data.frame containing the variables in formula. If not found in data, variables are taken from environment(formula).
subset	an optional logical or numeric vector specifying a subset of observations for which the model is estimated and efficiencies are computed.
distribution	character scalar specifying the distribution of the inefficiency term: default "e" (exponential). "h" (half-normal) and "t" (truncated normal) to be implemented.
prod	logical. If TRUE, estimates correspond to a stochastic *production* frontier and technical efficiencies are returned; if FALSE, estimates correspond to a stochastic *cost* frontier and cost efficiencies are returned. Default is TRUE.
start.val	optional numeric vector of starting values for the optimizer.
init.sk	numeric. Initial value for the skewness parameter of the noise component; default is 0.5.

ln.var.u	optional one-sided formula; e.g. ln.var.u = ~ z3 + z4. Specifies exogenous variables entering the (log) variance of the inefficiency component. If NULL, the inefficiency variance is homoskedastic, i.e., $\sigma^2_{u0} = \exp(\gamma_{u0}[0])$.
ln.var.v	optional one-sided formula; e.g. ln.var.v = ~ z1 + z2. Specifies exogenous variables entering the (log) variance of the random noise component. If NULL, the noise variance is homoskedastic, i.e., $\sigma^2_{v0} = \exp(\gamma_{v0}[0])$.
skew.v	optional one-sided formula; e.g. skew.v = ~ z5 + z6. Allows the skewness of the noise to depend linearly on exogenous variables. If NULL, the skewness is constant across units.
mean.u	optional one-sided formula; e.g. mean.u = ~ z7 + z8. Specifies whether the mean of the pre-truncated normal distribution of the inefficiency term is a linear function of exogenous variables. In cross-sectional models, used only when distribution = "t". If NULL, the mean is constant across units. To be implemented.
optim.report	integer. Verbosity level for reporting during optimization (if implemented). Default is 1.
optim.trace	integer. Trace level for optimization (if implemented). Default is 1.
reltol	numeric. Relative convergence tolerance used when maximizing the log-likelihood with optim. The algorithm stops if it cannot reduce the objective by a factor of reltol * (abs(val) + reltol) at a step. Default is sqrt(.Machine$double.eps).
lmtol	numeric. Convergence tolerance based on the scaled gradient (when applicable). Default is 1e-5.
maxit	numeric. Maximum number of iterations for the optimizer. Default is 199.
print.level	integer. Printing level: 1—estimation results; 2—optimization details; 3—summary of (cost/technical) efficiencies; 4—unit-specific point and interval estimates of efficiencies. Default is 3.
digits	integer. Number of digits for displaying estimates and efficiencies. Default is 4.
...	additional arguments passed to internal methods or to optim, as relevant (e.g., cost.eff.less.one = TRUE for cost-frontier conventions).
optim	logical. If TRUE, estimation proceeds via stats::optim; if FALSE, an internal routine (if provided) would be used. Default is FALSE.
optim.method	character. Method passed to stats::optim when optim = TRUE. Default is "bfgs".
optim.reltol	numeric. Relative tolerance specifically for optim; default 1e-8.

Details

Stochastic Frontier Model with a Skew-Normally Distributed Error Term

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

Options ln.var.u and ln.var.v allow for multiplicative heteroskedasticity in the inefficiency and/or noise components; i.e., their variances can be modeled as exponential functions of exogenous variables (including an intercept), as in Caudill et al. (1995).

Value

An object of class `"snreg"` with maximum-likelihood estimates and diagnostics:

`par` Numeric vector of ML parameter estimates at the optimum.

`coef` Named numeric vector equal to `par`.

`vcov` Variance–covariance matrix of the estimates.

`sds` Standard errors, `sqrt(diag(vcov))`.

`ctab` Coefficient table with columns `Coef.`, `SE`, `z`, `P>|z|`.

`ll` Maximized log-likelihood value.

`hessian` (When computed) Observed Hessian or OPG used to form `vcov`.

`value` (Optim-only, before aliasing) Objective value from `optim`.

`counts` (Optim-only) Iteration and evaluation counts from `optim`.

`convergence` Convergence code).

`message` (Optim-only) Message returned by `optim`, if any.

`gradient` (NR-only) Gradient at the solution.

`gg` (NR-only) Gradient-related diagnostic.

`gHg` (NR-only) Newton-step diagnostic.

`theta_rel_ch` (NR-only) Relative parameter change metric across iterations.

`resid` Regression residuals.

`RSS` Residual sum of squares `crossprod(resid)`.

`shat2` Residual variance estimate `var(resid)`.

`shat` Residual standard deviation `sqrt(shat2)`.

`aic` Akaike Information Criterion.

`bic` Bayesian Information Criterion.

`Mallows` Mallows' C_p-like statistic.

`u` Estimated inefficiency term (vector). Returned for models with an inefficiency component (e.g., exponential).

`eff` Efficiency scores `exp(-u)` (technical or cost, depending on `prod`).

`sv` Estimated (possibly unit-specific) standard deviation of the noise term.

`su` Estimated (possibly unit-specific) standard deviation or scale of the inefficiency term. For exponential models.

`skewness` Estimated skewness index (e.g., from the skewness equation).

`esample` Logical vector marking observations used in estimation.

`n` Number of observations used.

The returned object has class `"snreg"`.

Author(s)

Oleg Badunenko <Oleg.Badunenko.brunel.ac.uk>

References

Badunenko, O., & Henderson, D. J. (2023). *Production analysis with asymmetric noise*. Journal of Productivity Analysis, **61**(1), 1–18. https://doi.org/10.1007/s11123-023-00680-5

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

See Also

sf

Examples

```
## Not run:

library(snreg)

data("banks07")
head(banks07)

myprod <- FALSE

# Translog cost function
spe.tl <- log(TC) ~ (log(Y1) + log(Y2) + log(W1) + log(W2))^2 +
  I(0.5 * log(Y1)^2) + I(0.5 * log(Y2)^2) +
  I(0.5 * log(W1)^2) + I(0.5 * log(W2)^2)


# ------------------------------------------------------------
# Specification 1: homoskedastic & symmetric
# ------------------------------------------------------------
formSV <- NULL   # variance equation
formSK <- NULL   # skewness equation
formSU <- NULL   # inefficiency equation (unused here)

m1 <- snsf(
  formula  = spe.tl,
  data     = banks07,
  prod     = myprod,
  ln.var.v = formSV,
  skew.v   = formSK
)

coef(m1)


# ------------------------------------------------------------
# Specification 2: heteroskedastic + skewed noise
# ------------------------------------------------------------
formSV <- ~ log(TA)      # heteroskedastic variance
formSK <- ~ ER           # skewness driver
formSU <- ~ LA + ER      # inefficiency

m2 <- snsf(
  formula  = spe.tl,
  data     = banks07,
```

```
  prod     = myprod,
  ln.var.v = formSV,
  skew.v   = formSK
)

coef(m2)


## End(Not run)
```

---

su *Summary Utility (*su*)*

Description

Computes a compact table of summary statistics for each variable in a vector, matrix, or data frame. The following metrics are returned per variable: number of observations (Obs), missing values (NAs), mean, standard deviation (StDev), interquartile range (IQR), minimum (Min), user-specified quantiles (probs), and maximum (Max).

Usage

```
su(
  x,
  mat.var.in.col = TRUE,
  digits = 4,
  probs = c(0.1, 0.25, 0.5, 0.75, 0.9),
  print = FALSE
)
```

Arguments

x	a numeric vector, matrix, or data frame. For matrices, variables are assumed to be in columns; set mat.var.in.col = FALSE to treat rows as variables.
mat.var.in.col	logical. If TRUE (default), a matrix is interpreted as variables in columns. If FALSE, the matrix is transposed so that rows are treated as variables.
digits	integer. Number of digits to use when printing (only affects printed output when print = TRUE). Default is 4.
probs	numeric vector of probabilities in $[0, 1]$ for which quantiles are computed. Default is c(0.1, 0.25, 0.5, 0.75, 0.9).
print	logical. If TRUE, prints the transposed summary table using the specified number of digits. Default is FALSE.

Details

Compact Summary Statistics for Vectors, Matrices, and Data Frames

Input handling:

- If x is a matrix with a single row or column, it is treated like a vector. Column or row names are used (if available). Otherwise, a default name is created.

- If x is a matrix with multiple variables, variables are taken as columns. Use mat.var.in.col = FALSE to transpose and treat rows as variables.
- If x is a vector, its deparsed symbol name is used as the variable name.
- If x is a data frame, each column is summarized.

Missing values are excluded in all summary computations.

Value

A matrix (coercible to data.frame) where each row corresponds to a variable and columns contain the summary statistics: Obs, NAs, Mean, StDev, IQR, Min, the requested probs quantiles (named), and Max. The returned object is given class "snreg" for compatibility with package-specific print/summarization methods.

Examples

```
## Not run:
  # Vector
  set.seed(1)
  v <- rnorm(100)
  su(v, print = TRUE)

  # Matrix: variables in columns
  M <- cbind(x = rnorm(50), y = runif(50))
  su(M)

  # Matrix: variables in rows
  Mr <- rbind(x = rnorm(50), y = runif(50))
  su(Mr, mat.var.in.col = FALSE)

  # Data frame
  DF <- data.frame(a = rnorm(30), b = rexp(30), c = rbinom(30, 1, 0.3))
  out <- su(DF)
  head(out)

## End(Not run)
```

---

summary.snreg *Summary for Skew-Normal Regression Models*

Description

Produces a summary object for objects of class "snreg". The function assigns the class "summary.snreg" to the fitted model object, enabling a dedicated print method (print.summary.snreg) to display results in a structured format.

Usage

```
## S3 method for class 'snreg'
summary(obj, ...)
```

Arguments

obj	an object of class "snreg", typically returned by snreg.
...	additional arguments (currently not used).

Details

Summary Method for snreg Objects

This method expects a fitted "snreg" object.

summary.snreg does not modify the contents of the object; it only updates the class attribute to "summary.snreg". The corresponding print method (print.summary.snreg) is responsible for formatting and displaying estimation details, such as convergence criteria, log-likelihood, coefficient tables, and (if present) heteroskedastic and skewness components.

Value

An object of class "summary.snreg", identical to the input obj except for its class attribute.

See Also

snreg, print.summary.snreg

Examples

```
## Not run:
  # m <- snreg(TC ~ Y1 + Y2, data = banks07)
  # s <- summary(m)
  # print(s)

## End(Not run)
```

TOwen *Compute Owen's T Function* T(h, a)

Description

TOwen1 computes an Owen's T-function variant (or a related special function) for vectors h and a based on the t function in https://people.sc.fsu.edu/~jburkardt/c_src/owen/owen.html. Non-finite inputs (in h or a) produce NA at corresponding positions, while finite pairs are computed in C in a vectorized fashion.

Usage

```
TOwen(h, a, threads = 1)
```

Arguments

h	numeric vector of h arguments.
a	numeric vector of a arguments. Must be either the same length as h or of length 1 (will be recycled by standard R rules).
threads	integer. Number of threads to request from the C implementation (if supported). Default is 1.

Details

Owen's T Function via C Backend

Owen's T function is commonly defined as

$$T(h,a) \;=\; \frac{1}{2\pi}\int_0^a \frac{\exp\!\left(-\tfrac{1}{2}h^2(1+t^2)\right)}{1+t^2}\,dt,$$

for real h and a.

The function accepts vector inputs and:

- Computes results only for entries where both `h` and `a` are finite.
- Returns `NA` where either `h` or `a` is non-finite.
- Optionally passes a `threads` hint to the C backend (ignored if not supported).

Value

A numeric vector of length `length(h)` containing $T(h_i, a_i)$. Elements where either `h_i` or `a_i` is not finite are `NA`. The returned object is given class `"snreg"` for downstream compatibility with your package's print/summary helpers.

See Also

`pnorm`, `dnorm`

Examples

```
## Not run:
  # Basic usage. Vectorized 'a'
  h <- c(-1, 0, 1, 2)
  a <- 0.5
  TOwen(h, a)

  # Vectorized 'a' with non-finite entries; non-finite entries yield NA
  a2 <- c(0.2, NA, 1, Inf)
  TOwen(h, a2)

## End(Not run)
```

`TOwen1` *Compute Owen-like Function* `TOwen1(h, a)`

Description

`TOwen1` computes an Owen's T-function variant (or a related special function) for vectors `h` and `a` based on the `tha` function in https://people.sc.fsu.edu/~jburkardt/c_src/owen/owen.html. Non-finite inputs in `h` or `a` yield `NA` at the corresponding positions.

Usage

```
TOwen1(h, a, threads = 1)
```

Arguments

threads	integer. Number of threads to request from the C implementation (if supported). Default is 1.

Details

Owen's T Function Variant via C Backend

This is a thin R wrapper around a native routine with signature:

```
void TOwen1(int *n, double *h, double *a, double *out, int *threads)
```

Value

A numeric vector of length length(h) with the computed values. Elements where either h or a is non-finite are NA. The returned vector is given class "snreg" for downstream compatibility.

See Also

TOwen

Examples

```
## Not run:
  library(snreg)

  # Basic usage. Vectorized 'a':
  h <- c(-1, 0, 1, 2)
  a <- 0.3
  TOwen1(h, a)

  # Vectorized 'a' with non-finite entries:
  a2 <- c(0.2, NA, 1, Inf)
  TOwen1(h, a2)

## End(Not run)
```

---

vcov.snreg *Extract the Variance–Covariance Matrix*

Description

vcov.snreg is the vcov S3 method for objects of class "snreg". It returns the model-based variance–covariance matrix stored in the fitted object.

Usage

```
## S3 method for class 'snreg'
vcov(obj, ...)
```

Arguments

obj	an object of class "snreg", typically returned by snreg.
...	additional arguments (currently unused).

Details

Variance–Covariance Matrix for snreg Objects

This method expects a fitted "snreg" object.

This method simply returns the vcov component stored in obj. If your estimator did not compute standard errors (e.g., because estimation hasn't been run yet in a scaffold), this field may be NULL, and the method will error accordingly.

Value

A numeric matrix containing the variance–covariance of the estimated parameters.

See Also

snreg, summary.snreg

Examples

```
## Not run:
  library(snsf)

  data("banks07")
  head(banks07)
  # V <- vcov(m)
  # diag(V)

## End(Not run)
```

Index