Package 'infometrics'

September 10, 2026

Type Package

Title Information-Theoretic Methods for Econometric Estimation

Version 0.3.0

Date 2026-08-29

Description Implements the class of Information-Theoretic (IT) estimators for econometric models, following the unified framework of Golan (2008) <doi:10.1561/0800000004>.
Provides Generalized Maximum Entropy (GME) and Generalized Cross-Entropy (GCE) estimators for linear regression, instrumental variables, one-way error-component panel data, multinomial response, matrix balancing, and first-order Markov transition matrices, together with pure and noisy inverse-problem solvers. All estimators use the concentrated (dual) formulation for computational efficiency and report normalized-entropy, entropy-ratio, and Fano-bound diagnostics.

License GPL-3

URL https://github.com/GenMaxEnt/infometrics_R

BugReports https://github.com/GenMaxEnt/infometrics_R/issues

Encoding UTF-8

Depends R (>= 4.1.0)

Imports stats

Suggests bookdown, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Ganbaatar Jambal [aut, cre]

Maintainer Ganbaatar Jambal <jg3169a@gmail.com>

Repository CRAN

Date/Publication 2026-09-10 13:00:02 UTC

Contents

`default_supports` *Default Support Spaces from OLS*

Description

Constructs default signal (Z) and error (V) support spaces for GME estimation using OLS estimates as reference points. The signal support for each coefficient is centered on the OLS estimate with half-range equal to `signal_scale` times the absolute OLS estimate. The error support is symmetric around zero with half-range equal to `error_scale` times the OLS residual standard deviation.

Usage

```
default_supports(
  y,
  X,
  M_signal = 5L,
  M_error = 5L,
  signal_scale = 2,
  error_scale = 3
)
```

Arguments

y	Numeric vector of length T. The dependent variable.
X	Numeric matrix of dimension T x K. The regressor matrix.
M_signal	Integer. Number of support points per coefficient (default 5).
M_error	Integer. Number of support points for error terms (default 5).
signal_scale	Positive numeric. Half-range multiplier for the signal support (default 2: support spans +/- 2 * |OLS estimate|).
error_scale	Positive numeric. Half-range multiplier for the error support (default 3: support spans +/- 3 * OLS residual SD, the "three-sigma rule" of Pukelsheim, 1994).

Value

A list with two elements:

Z K x M_signal matrix. Row k is the support for beta_k.

V Numeric vector of length M_error. The error support, symmetric around zero.

References

Golan, A. (2008). Information and Entropy Econometrics. *Foundations and Trends in Econometrics*, **2**(1-2), 1-145. Section 6.1.

Pukelsheim, F. (1994). The three sigma rule. *The American Statistician*, **48**, 88-91.

Examples

```
set.seed(42)
n <- 50; K <- 3
X <- cbind(1, matrix(rnorm(n * (K-1)), n, K-1))
y <- X %*% c(1, 2, -1) + rnorm(n)
sp <- default_supports(y, X)
sp$Z   # 3 x 5 signal support matrix
sp$V   # 5-point error support
```

---

fano_bounds *Information-Theoretic (Fano) Error Bounds*

Description

Generic for Fano's-inequality error bounds on a fitted probability matrix.

Usage

```
fano_bounds(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Passed to methods.

Value

A method-specific object of error bounds.

See Also

`fano_bounds.multinomial_gce`

`fano_bounds.inverse_ce`

Fano Error Bounds for a Pure Inverse-Problem (Cross-Entropy) Fit

---

Description

The recovered `p` is a single distribution over the K states, so Golan's (2008, sec 7.5) Fano bound applies to it directly and is the natural companion to the normalized entropy `S`: reading `S` as an irreducible prediction error, a modal classifier that guesses $\arg\max_k p_k$ errs with probability $pe = 1 - \max_k p_k$, and Fano's inequality lower-bounds this by

$$pe \geq S(p) - \log 2/\log K, \qquad S(p) = H(p)/\log K.$$

The `S` used *here* is the uniform-reference $H(p)/\log K$ the Fano bound requires – distinct from `inverse_ce`'s prior-relative reported $S = H(p)/H(p_0)$. This is an information-theoretic error bound on prediction accuracy, *not* a sampling standard error (for the identification SEs of the multipliers see `vcov.inverse_ce` / `summary`).

Usage

```
## S3 method for class 'inverse_ce'
fano_bounds(object, ...)
```

Arguments

`object`	An `inverse_ce` object.
`...`	Unused.

Value

A one-row data frame with columns `p_max`, `pe` (modal error), `H` (entropy, nats), `S` (uniform-normalized entropy), and `pe_lower` (Fano weak lower bound). An `"overall"` attribute holds `mean_pe`, `mean_pe_lower`, and `S_system` (here identical to the single row).

References

Golan, A. (2008). Information and Entropy Econometrics. *Foundations and Trends in Econometrics*, **2**(1-2), 1-145 (sec 7.5); Fano, R. (1961). *Transmission of Information*.

See Also

fano_bounds, inverse_ce

---

fano_bounds.inverse_noise

Fano Error Bounds for a Noisy Inverse-Problem (GME/GCE) Fit

Description

The recovered p_hat is a single distribution over the K states, so Golan's (2008, sec 7.5) Fano bound applies to it directly (a parallel of fano_bounds.inverse_ce): a modal classifier that guesses $\arg\max_k p_k$ errs with probability $pe = 1 - \max_k p_k$, and $pe \geq S(p) - \log 2/\log K$ with $S(p) = H(p)/\log K$ (uniform-reference, distinct from the prior-relative S/S_w the object reports). An information-theoretic error bound, *not* a sampling SE (for those see vcov.inverse_noise / summary).

Usage

```
## S3 method for class 'inverse_noise'
fano_bounds(object, ...)
```

Arguments

object	An inverse_noise object.
...	Unused.

Value

A one-row data frame with columns p_max, pe, H, S, and pe_lower; an "overall" attribute mirrors the row.

References

Golan, A. (2008). Information and Entropy Econometrics. *Foundations and Trends in Econometrics*, **2**(1-2), 1-145 (sec 7.5); Fano, R. (1961). *Transmission of Information*.

See Also

fano_bounds, inverse_noise

fano_bounds.mixed_gce *Fano Error Bounds for a Mixed GCE Fit*

Description

For each observation (a row of `p_hat`, a distribution over the J categories) the modal classifier predicts $\arg\max_j p_{ij}$ with error $pe_i = 1 - \max_j p_{ij}$, lower-bounded (Golan 2008, sec 7.5) by $S(p_i) - \log 2/\log J$. This is an information-theoretic error bound, *not* a sampling standard error (for those see `margins` with `se = TRUE`).

Usage

```
## S3 method for class 'mixed_gce'
fano_bounds(object, ...)
```

Arguments

`object`	A `mixed_gce` object.
`...`	Unused.

Value

A data frame with one row per observation (`p_max`, `pe`, `H`, `S`, `pe_lower`) and an `"overall"` attribute.

See Also

`fano_bounds`, `margins`

fano_bounds.multinomial_gce *Fano Error Bounds for a Multinomial GCE Fit*

Description

For each observation (a row of `p_hat`, a distribution over the J alternatives) the modal classifier predicts $\arg\max_j p_{ij}$ with error probability $pe_i = 1 - \max_j p_{ij}$. Fano's inequality (Golan 2008, sec 7.5) lower-bounds this error by the normalized entropy:

$$pe_i \geq S(p_i) - \log 2/\log J, \qquad S(p_i) = H(p_i)/\log J.$$

This is an **information-theoretic error bound** on prediction accuracy, *not* a sampling standard error (for the latter see `margins` with `se = TRUE`).

Usage

```
## S3 method for class 'multinomial_gce'
fano_bounds(object, ...)
```

Arguments

`object`	A `multinomial_gce` object.
`...`	Unused.

Value

A data frame with one row per observation and columns `p_max`, `pe` (modal error), `H` (entropy, nats), `S` (normalized entropy), and `pe_lower` (Fano weak lower bound). An `"overall"` attribute holds `mean_pe`, `mean_pe_lower`, and `S_system`.

References

Golan, A. (2008). Information and Entropy Econometrics. *Foundations and Trends in Econometrics*, **2**(1-2), 1-145 (sec 3.6, 7.5); Fano, R. (1961). *Transmission of Information*.

See Also

`margins`

`inverse_ce` *Pure Inverse Problem via Cross-Entropy (Formula Interface)*

Description

Estimates the K-dimensional distribution `p` closest (in Kullback-Leibler divergence) to a prior `p0` subject to the T pure moment constraints $y = Xp$, by solving the unconstrained dual (concentrated) model of Golan (2008, Section 4.2; see also Golan, Judge & Miller, 1996, Ch. 3). This is the `lm`-style formula interface to the maximum-entropy / cross-entropy problem, where a uniform prior `p0` gives Maximum Entropy; a non-uniform `p0` gives Cross-Entropy (hence the name).

Usage

```
inverse_ce(formula, data, p0 = NULL, subset, na.action, control = list(), ...)

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

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

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

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

## S3 method for class 'inverse_ce'
```

```
print(x, digits = max(3L, getOption("digits") - 3L), ...)

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

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

Arguments

`formula`	A model formula. The response is the moment vector `y`; each RHS term is a state. Usually written without an intercept, e.g. `y ~ x1 + x2 + x3 - 1`.
`data`	A data frame (or environment) with one row per moment.
`p0`	Numeric vector of length K (number of model-matrix columns): strictly-positive prior probabilities for the Cross-Entropy formulation. Defaults to the uniform distribution, which makes CE reduce to ME. (Named `p0` to match the GME/GCE signal-prior convention; for `inverse_ce` it is a K-vector prior over the states, not a K-by-M support-point prior.)
`subset`	Optional vector specifying a subset of moments (rows) to use, as in `lm`.
`na.action`	A function indicating how to handle `NA`s in the model frame, as in `lm`.
`control`	A named list of control parameters merged over the defaults and passed to `optim` (BFGS): `maxit` (default 500) and `reltol` (default 1e-10).
`...`	Currently unused.
`object, x`	An `inverse_ce` object.
`digits`	Number of significant digits to print.

Details

The dual concentrated objective (Golan 2008, Eq. 4.4-4.5), minimised here to match the package's dual sign convention, is

$$\ell(\lambda) = -\sum_t \lambda_t y_t + \log \Omega(\lambda),$$

with partition function $\Omega(\lambda) = \sum_k p_{0,k} \exp(\sum_t \lambda_t x_{tk})$. Minimising this convex objective is equivalent to maximising the entropy of `p` subject to the moment constraints. With uniform `p0` this is Maximum Entropy; with a non-uniform prior it is Cross-Entropy. The estimated probabilities are recovered as

$$\hat{p}_k = p_{0,k} \exp(\textstyle\sum_t \hat{\lambda}_t x_{tk}) / \Omega(\hat{\lambda})$$

(Golan 2008, Eq. 4.3), and the analytic dual Hessian is $\nabla^2 \ell = X \operatorname{diag}(p)\, X' - (Xp)(Xp)' = \operatorname{Cov}_p(\text{moments})$ (Eq. 4.7), which is positive semidefinite.

Value

An object of class `c("inverse_ce", "infometrics")`: a list with components

`p_hat` Named K-vector of estimated probabilities.

`lambda_hat` Named T-vector of Lagrange multipliers (dual variables).

`fitted.values` Fitted moments Xp (length T).

`residuals` Moment residuals $y - Xp$ (length T).

`hessian` Analytic T-by-T dual Hessian = information matrix $I(\lambda) = \mathrm{Cov}_p(\text{moments})$ (Eq. 4.7), positive semidefinite.

`vcov_lambda` $\mathrm{Var}(\lambda) = I^{-1}(\lambda)$ (T-by-T), or `NULL` when I is singular.

`se_lambda` Information-matrix SEs of λ (length T), or `NA` when I is singular.

`se_p` Delta-method SEs of the probabilities `p` (length K), or `NA` when I is singular.

`H_signal` Shannon entropy $H(\hat{p})$.

`S` Normalized entropy $H(\hat{p})/H(p_0)$ in [0, 1] (pseudo-R2 = 1 - S).

`objective` Dual objective at the optimum.

`p0` The prior distribution used.

`convergence` Raw `optim` convergence code.

`method` Solver used (`"dual"`).

`terms`, `model`, `call` Standard model components.

Methods (by generic)

- `coef(inverse_ce)`: Estimated probabilities `p`.
- `fitted(inverse_ce)`: Fitted moments Xp.
- `residuals(inverse_ce)`: Moment-fitting residuals $y - Xp$.
- `vcov(inverse_ce)`: Information-matrix covariance of the multipliers $\mathrm{Var}(\lambda) = I^{-1}(\lambda)$ (T-by-T), or `NULL` when I is singular.
- `print(inverse_ce)`: Compact printed display.
- `summary(inverse_ce)`: lm()-style summary (information-matrix SE tables + Fano).

Functions

- `print(summary.inverse_ce)`: Print method for the summary object.

Information-matrix standard errors

The Hessian above *is* the Fisher information matrix of the multipliers, $I(\lambda) = \mathrm{Cov}_p(\text{moments})$, so its inverse gives their covariance $\mathrm{Var}(\lambda) = I^{-1}(\lambda)$ (Golan 2008, p. 59; Cover & Thomas, 2006, Ch. 17). `inverse_ce()` reports the resulting `se_lambda`, the delta-method probability SEs `se_p` ($\mathrm{Var}(p) = J\, I^{-1} J'$ with $J_{kt} = \partial p_k / \partial \lambda_t = p_k(x_{tk} - E_p[x_t])$), and the full `vcov_lambda` (via `vcov`). These are **information-matrix / curvature** quantities: they measure how well the moments identify λ – following the multiplier's reading as the *marginal information content of a moment* (a moment t with larger $\mathrm{Var}_p(x_t)$ carries more information, hence a smaller $SE(\lambda_t)$; $\lambda_t \approx 0$ means moment t adds little) – and are *not* frequentist sampling standard errors, because a pure inverse problem is deterministic. $I(\lambda)$ is singular when a moment row is constant/collinear or $T \geq K$ (its rank is at most $K - 1$); there the affected SEs are returned as `NA` rather than a pseudo-inverse's misleadingly finite value. For sampling-based inference use the stochastic-moment `inverse_noise`.

Formula orientation

Unlike a regression, the response is the vector of *moments* and each right-hand-side term is a *state*. The model matrix built from `formula` and `data` is therefore the T-by-K matrix X with rows = moment constraints (T) and columns = states / support points (K); the data frame has one row per moment. The estimated coefficients (`coef`) are the K probabilities `p`, one per column. Because columns are states, an intercept would add a spurious extra state; drop it with `- 1`. A warning is issued if an intercept is present.

References

Golan, A. (2008). *Information and Entropy Econometrics - A Review and Synthesis*. Foundations and Trends in Econometrics, 2(1-2), 1-145. Section 4.2. Cover, T. M. & Thomas, J. A. (2006). *Elements of Information Theory*, 2nd ed., Ch. 17. Golan, A., Judge, G. and Miller, D. (1996). *Maximum Entropy Econometrics: Robust Estimation with Limited Data*. Wiley. Chapter 3.

See Also

`inverse_noise` for the stochastic-moment (GME/GCE) estimator that reports sampling standard errors; `fano_bounds` for the Fano error bound on the recovered `p`.

Examples

```
# Recover a 5-state distribution from 2 exact moment constraints.
# The data frame has one row per moment (T = 2) and one column per
# state (K = 5). Because p_true is the maximum-entropy distribution for
# its own moments, it is recovered exactly.
X <- rbind(c(1, 2, 3, 4, 5),
           c(1, 4, 9, 16, 25))           # 2 moments x 5 states
p_true <- exp(as.vector(crossprod(X, c(0.2, -0.05))))
p_true <- p_true / sum(p_true)
dat <- data.frame(y  = as.vector(X %*% p_true),
                  s1 = X[, 1], s2 = X[, 2], s3 = X[, 3],
                  s4 = X[, 4], s5 = X[, 5])
fit <- inverse_ce(y ~ s1 + s2 + s3 + s4 + s5 - 1, data = dat)
fit
coef(fit)          # ~ p_true
summary(fit)       # info-matrix SEs of lambda / p + a Fano line
vcov(fit)          # Var(lambda) = I^{-1}
fano_bounds(fit)   # Fano error bound for the recovered p
```

`inverse_noise` *Noisy Inverse Problem via Generalized Cross-Entropy (Formula Interface)*

Description

Estimates a K-dimensional signal distribution `p` together with per-moment noise distributions `w` for the stochastic-moment problem $y = Xp + e$, $e_t = \sum_j v_j w_{tj}$, by solving the unconstrained dual (concentrated) model of Golan (2008, Section 6.1). This is the noisy-moment counterpart of `inverse_ce`.

Usage

```
inverse_noise(
  formula,
  data,
  v = NULL,
  nu = 0.5,
  p0 = NULL,
  w0 = NULL,
  se_method = c("sandwich", "delta", "bootstrap", "none"),
  B = 1000L,
  subset,
  na.action,
  control = list(),
  ...
)

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

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

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

## S3 method for class 'inverse_noise'
vcov(object, type = c("sandwich", "delta", "bootstrap"), B = 1000L, ...)

## S3 method for class 'inverse_noise'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

## S3 method for class 'inverse_noise'
summary(object, se_method = NULL, B = 1000L, ...)

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

Arguments

`formula`	A model formula. The response is the moment vector; each RHS term is a state. Usually written without an intercept, e.g. `y ~ x1 + x2 + x3 - 1`.

data	A data frame (or environment) with one row per moment.
v	Noise support. One of: NULL (default 3-point symmetric support of half-width $\max(3, \sqrt{2\log T})\, s$ with $s = sd(y)$); a single whole number M >= 2 (build an M-point symmetric support); or an explicit numeric vector of support points (should be symmetric around 0 for zero-mean noise). The default widens with T rather than using a fixed three-sigma rule, because the largest of T noise draws grows like $\sigma\sqrt{2\log T}$; see the *Support feasibility* section.
nu	Entropy weight in $(0, 1)$ on the noise term; default 0.5 (equal-weight GME/GCE).
p0	Prior signal probabilities, length K (= model-matrix columns); strictly positive, default uniform (GME). A non-uniform prior gives GCE.
w0	Prior noise probabilities, a T x M matrix (one row per moment); strictly positive rows summing to 1, default uniform. Its column count must match length(v).
se_method	For summary: SE method to report ("sandwich", "delta", or "bootstrap"); defaults to the stored method.
B	Integer number of bootstrap resamples when se_method = "bootstrap" (or vcov/summary with type/se_method = "bootstrap"); default 1000.
subset	Optional vector specifying a subset of moments (rows) to use, as in lm.
na.action	A function indicating how to handle NAs in the model frame, as in lm.
control	A named list merged over the defaults and passed to optim (BFGS): maxit (default 500) and reltol (default 1e-10).
...	Currently unused.
object, x	An inverse_noise object.
type	Covariance method for vcov: "sandwich" (default), "delta", or "bootstrap".
digits	Number of significant digits to print.

Details

Unlike inverse_ce, the moments are treated as *noisy*: the constraint is $y = Xp + e$ rather than $y = Xp$. The estimator therefore shrinks p toward its prior p0; how much depends on the width of the noise support v. As v collapses toward 0 the solution approaches the pure-CE solution of inverse_ce.

The dual objective is minimised here (convex) to match the package convention used by inverse_ce:

$$\ell(\lambda) = -\sum_t \lambda_t y_t + (1-\nu)\log\Omega(\lambda) + \nu\sum_t \log\Psi_t(\lambda),$$

with signal partition function $\Omega = \sum_k p_{0,k}\exp(\sum_t \lambda_t x_{tk}/(1-\nu))$ and per-moment noise partition functions $\Psi_t = \sum_j w_{0,tj}\exp(\lambda_t v_j/\nu)$. The weight $\nu \in (0,1)$ splits entropy between signal and noise; $\nu = 0.5$ gives the standard equal-weight GME/GCE. Minimising this convex objective is equivalent to maximising the joint signal-plus-noise entropy subject to the noisy moment constraints. The estimates are recovered as $\hat{p}_k \propto p_{0,k}\exp(\sum_t \hat{\lambda}_t x_{tk}/(1-\nu))$ and $\hat{w}_{tj} \propto w_{0,tj}\exp(\hat{\lambda}_t v_j/\nu)$, with $\hat{e}_t = \sum_j v_j \hat{w}_{tj}$. Uniform priors (p0 = NULL, w0 = NULL) give the GME version; user-supplied priors give GCE.

Value

An object of class `c("inverse_noise", "infometrics")`: a list with components

`p_hat` Named K-vector of signal probabilities.

`foc_residual` $\max|y - Xp - e|$ at the optimum; near 0 for a healthy fit (see *Support feasibility*).

`lambda_hat` Named T-vector of Lagrange multipliers.

`w_hat` `T` x `M` matrix of noise probabilities.

`fitted.values` Signal-fitted moments Xp (length T).

`residuals` Moment residuals $y - Xp$, equal to the estimated noise $e_t = \sum_j v_j w_{tj}$ at the optimum.

`hessian` Analytic T-by-T dual Hessian (positive definite).

`vcov_lambda`, `se_lambda`, `se_p` Covariance of λ and standard errors of λ/p by `se_method` (`NULL` when `se_method = "none"`).

`se_method` The SE method stored.

`H_signal`, `H_error`, `H_error0` Shannon entropies of the signal $H(\hat{p})$, the noise $H(\hat{w})$, and the noise prior $H(w_0)$.

`S`, `S_p` Normalized *signal* entropy $H(\hat{p})/H(p_0)$ in [0, 1] (pseudo-R2 = 1 - S).

`S_w` Normalized *noise* entropy $H(\hat{w})/H(w_0)$.

`objective` Dual objective at the optimum.

`nu`, `prior`, `noise_prior`, `support` The entropy weight, signal prior `p0`, noise prior `w0`, and the noise support `v`.

`X`, `y`, `control` Stored for SE recomputation and the bootstrap.

`convergence` Raw `optim` convergence code.

`method`, `terms`, `model`, `call` Solver and standard model components.

Methods (by generic)

- `coef(inverse_noise)`: Estimated signal probabilities `p`.
- `fitted(inverse_noise)`: Fitted (signal) moments Xp.
- `residuals(inverse_noise)`: Residuals $y - Xp$ (the estimated noise).
- `vcov(inverse_noise)`: Covariance of the Lagrange multipliers by method. `type = "sandwich"` (default) is the robust $H^{-1}\mathrm{diag}(\hat{e}^2)H^{-1}$; `"delta"` is the naive H^{-1} (overstates ~10x); `"bootstrap"` refits `B` residual resamples. **Note**: the default changed from the raw H^{-1} to the sandwich, since H^{-1} is not a valid sampling covariance here.
- `print(inverse_noise)`: Compact printed display.
- `summary(inverse_noise)`: lm()-style summary; prints a `p` coefficient table with standard errors and t-stats, the signal/noise normalized entropies, and a Fano line. `se_method` selects the SE method (default: the one stored on the object).

Functions

- `print(summary.inverse_noise)`: Print method for the summary object.

Standard errors

Structurally this is a GME regression $y = Xp + e$ with p on the simplex, so the T moment rows are the effective sample size and sampling SEs are meaningful. Because $\hat{\lambda}$ solves $Xp(\lambda)+e(\lambda)-y = 0$, the implicit-function theorem gives $\partial\hat{\lambda}/\partial y = H^{-1}$, so $\mathrm{Var}(\hat{\lambda}) = H^{-1}\Sigma_e H^{-1}$ and $\mathrm{Var}(\hat{p}) = J\,\mathrm{Var}(\hat{\lambda})\,J'$ with $J_{kt} = \partial p_k/\partial\lambda_t = (1-\nu)^{-1}p_k(x_{tk} - (Xp)_t)$. Three methods are provided via `se_method` / `vcov(type=)` / `summary(se_method=)`:

`"sandwich"` **(default)** robust HC0 $H^{-1}\,\mathrm{diag}(\hat{e}^2)\,H^{-1}$ (meat = squared fitted noise). Accurate for p (validated against a Monte-Carlo SD and the bootstrap); per-element `lambda` SEs are noisier (each rests on one $\hat{e}_t^2$).

`"delta"` the naive H^{-1}. Kept for comparison but it **overstates the sampling SE by roughly ten-fold** (it is not a sampling covariance) – do not use it for inference.

`"bootstrap"` residual bootstrap: keep the rows fixed, resample the fitted noise $\hat{e}$, refit. Gives stable, aligned SEs for both `lambda` and p; the gold-standard cross-check.

All three mildly understate (~15-20%) because the finite-support entropy penalty shrinks the fitted noise below the true noise dispersion (the bootstrap shows the same, so it is a property of the estimator and of an over-wide support, not a bug).

Support feasibility

The dual is bounded below only if y can be represented as $Xp+e$ with p on the simplex and every e_t inside the noise support v. If it cannot, the dual is **unbounded**: the multipliers diverge, the softmax saturates, and `p_hat` collapses to a vertex of the simplex (a 0/1 corner) even though `optim` reports convergence. `inverse_noise()` therefore checks the first-order condition $y - Xp - e = 0$ at the optimum, reports it as `foc_residual`, and **warns** when it is not satisfied. The remedy is a wider noise support v.

Formula orientation

Unlike a regression, the response is the vector of *moments* and each right-hand-side term is a *state*. The model matrix built from `formula` and `data` is therefore the T-by-K matrix X with rows = moment constraints (T) and columns = states / support points (K); the data frame has one row per moment. The estimated coefficients (`coef`) are the K probabilities p, one per column. Because columns are states, an intercept would add a spurious extra state; drop it with `- 1`. A warning is issued if an intercept is present.

References

Golan, A. (2008). *Information and Entropy Econometrics - A Review and Synthesis*. Foundations and Trends in Econometrics, 2(1-2), 1-145. Section 6.1.

See Also

`inverse_ce` for the exact-moment (pure) version; `linreg` for the GME/GCE regression estimator; `fano_bounds` for the Fano error bound on `p_hat`.

Examples

```
set.seed(123)
X <- matrix(runif(30), nrow = 10, ncol = 3)   # T = 10 moments, K = 3 states
p_true <- c(0.5, 0.2, 0.3)
y <- as.vector(X %*% p_true) + rnorm(nrow(X), 0, 0.3)
dat <- data.frame(y = y, x1 = X[, 1], x2 = X[, 2], x3 = X[, 3])
fit <- inverse_noise(y ~ x1 + x2 + x3 - 1, data = dat)
fit
summary(fit)                # p table (SE + t), signal/noise entropy, Fano
vcov(fit)                   # sandwich Cov(lambda)
fano_bounds(fit)            # Fano error bound for p_hat
```

---

linreg *Generalized Cross-Entropy Linear Regression (Formula Interface)*

Description

Fits the linear model $y = X\beta + e$ by Generalized Cross-Entropy (Golan 2008, Section 6.1, Eq. 6.5; see also Golan, Judge & Miller, 1996, Ch. 6). Each coefficient is reparameterized on a bounded support, $\beta_k = \sum_m z_{km} p_{km}$, and each error on a noise support, $e_t = \sum_j v_j w_{tj}$; the signal and noise entropies (relative to priors `p0`, `w0`) are maximized subject to the data constraints, with weight $\nu \in (0, 1)$ on the noise. Estimation uses the concentrated dual in the T Lagrange multipliers.

Usage

```
linreg(
  formula,
  data,
  Z = NULL,
  nu = 0.5,
  p0 = NULL,
  v = NULL,
  w0 = NULL,
  subset,
  na.action,
  control = list(),
  ...
)

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

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

## S3 method for class 'linreg'
```

```
residuals(object, ...)

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

## S3 method for class 'linreg'
predict(object, newdata, ...)

## S3 method for class 'linreg'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

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

## S3 method for class 'summary.linreg'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  signif.stars = getOption("show.signif.stars"),
  ...
)
```

Arguments

`formula`, `data`, `subset`, `na.action`
: Standard model-frame arguments, as in `lm`. The intercept is kept by default.

`Z`
: Coefficient support. Either a length-M numeric vector (a common support recycled across all coefficients) or a K-by-M matrix whose rows correspond, in order, to the columns of the model matrix (so row 1 is the intercept's support when an intercept is present). Default: a 5-point symmetric support spanning +/- a multiple of `sd(y)`.

`nu`
: Entropy weight in $(0, 1)$ on the noise term, hence the weight on the signal is $1 - \nu$; default 0.5 (equal weight on both the signal and the noise).

`p0`
: Prior signal probabilities, a K-by-M matrix matching `Z`; strictly positive rows summing to 1. Default uniform.

`v`, `w0`
: Noise support and prior, as in `inverse_noise`: `v` is `NULL` (3-point default of half-width $\max(3, \sqrt{2 \log T})\, sd(y)$, which widens with the sample size – see *Support feasibility*), a whole-number support count, or an explicit vector; `w0` is a strictly-positive T-by-J matrix with rows summing to 1 (default uniform).

`control`
: A named list merged over the defaults and passed to `optim` (BFGS): `maxit` (default 500) and `reltol` (default 1e-12).

`...`
: Currently unused.

`object`, `x`
: A `linreg` object.

`newdata`
: Optional data frame of new observations.

`digits`
: Significant digits to print.

`signif.stars`
: Logical; show significance stars (default from options).

Details

This fits the Generalized Cross-Entropy (GCE) linear regression estimator with an lm-style interface, adding a `predict` method, an `lm`-style coefficient table, and `r.squared`. The dual is minimised here (convex) to match the package convention used by `inverse_ce` and `inverse_noise`; this is equivalent to maximising the joint signal-plus-noise entropy. Uniform priors (`p0 = NULL`, `w0 = NULL`) give GME; user priors give GCE.

`summary()` reports a per-coefficient Entropy-Ratio (ER) test of $H_0 : \beta_k = 0$ (Golan 2008, Sec. 6.4/6.6) in place of a Wald z test. To do so it refits the model $K + 1$ times (one per coefficient plus one joint test of all coefficients = 0), each warm-started from the fitted multipliers; the resolved `Z`, `p0`, `v`, `w0`, `X`, `y` are stored on the object to make these refits faithful.

Value

An object of class `c("linreg", "infometrics")`, which is a list containing the following components:

`coefficients` Numeric vector of length K: the estimated coefficients $\hat{\beta} = Z\hat{p}$, named by the model-matrix columns. Extracted by `coef`.

`p`, `p_hat` K-by-M matrix of estimated signal probabilities (two names for the same object).

`lambda`, `lambda_hat` Numeric vector of length T: the estimated Lagrange multipliers (two names for the same object).

`w`, `w_hat` T-by-J matrix of estimated noise probabilities (two names for the same object).

`e` Numeric vector of length T: the estimated noise, $e_t = \sum_j v_j w_{tj}$.

`fitted.values` Numeric vector of length T: $X\hat{\beta}$. Extracted by `fitted`.

`residuals` Numeric vector of length T: $y - X\hat{\beta}$, which equals `e` at a converged optimum. Extracted by `residuals`.

`vcov` K-by-K asymptotic covariance matrix of $\hat{\beta}$ (Golan 2008, p. 96). Extracted by `vcov`.

`hessian` T-by-T dual Hessian at the optimum; positive definite.

`r.squared` Single number: the ordinary coefficient of determination.

`H_signal` Numeric vector of length K: the per-coefficient signal entropies $H(\hat{p}_k)$.

`S` Single number: the normalized signal entropy $H(\hat{p})/H(p_0)$. With a uniform prior this reduces to $H/(K \log M)$ and lies in [0, 1]; with an informative prior it is measured relative to that prior and may exceed 1.

`objective`, `value` Single number: the minimised dual objective (two names for the same value).

`foc_residual` Single number: $\max|y - X\hat{\beta} - \hat{e}|$ at the optimum. Near 0 for a healthy fit; see *Support feasibility*.

`converged` Logical: `TRUE` when `optim` converged *and* the first-order condition is satisfied.

`convergence` Integer: the raw `optim` convergence code.

`nu` Single number: the entropy weight used.

`method` Character string naming the solver, `"dual"`.

`Z`, `p0`, `v`, `w0`, `X`, `y`, `control` The resolved inputs, retained so that `summary()` can refit the model for the entropy-ratio test.

`terms`, `model`, `xlevels`, `call` The standard model components, as returned by `lm`.

Methods (by generic)

- `coef(linreg)`: Estimated coefficients β.
- `fitted(linreg)`: Fitted values $X\beta$.
- `residuals(linreg)`: Residuals $y - X\beta$ (the estimated noise `e`).
- `vcov(linreg)`: Asymptotic covariance of β, Golan (2008) p. 96 (coincides with OLS as the supports widen).
- `predict(linreg)`: Predictions for `newdata` (or fitted values).
- `print(linreg)`: Compact display.
- `summary(linreg)`: lm()-style summary with an entropy-ratio coefficient table. For each coefficient k an Entropy-Ratio (ER) test of $H_0 : \beta_k = 0$ is computed by refitting with row k of the signal support collapsed to zero and forming $\mathrm{ER}_k = 2[H^*_{\text{unrestricted}} - H^*_{\text{restricted},k}]$ with $H^* = \sum H(p) + \sum H(w)$; under H_0, $\mathrm{ER}_k \sim \chi^2_1$ (Golan 2008, Sec. 6.4/6.6). An overall ER test of H_0 : all coefficients (intercept and slopes) $= 0$ is also reported. Note this performs $K + 1$ warm-started refits.

Functions

- `print(summary.linreg)`: Print method for the summary object.

Support feasibility

The dual is bounded below only if `y` can be represented as $X\beta + e$ with every β_k inside its signal support `Z` and every e_t inside the noise support `v`. If it cannot, the dual is **unbounded**: the multipliers diverge, the signal softmax saturates and β pins to a vertex of `Z` even though `optim` reports convergence. This is why the default `v` widens with T – the largest of T errors grows like $\sigma\sqrt{2\log T}$, so a fixed three-sigma support becomes infeasible in large samples. `linreg()` checks the first-order condition $y - X\beta - e = 0$ at the optimum, reports it as `foc_residual`, and **warns** (setting `converged = FALSE`) when it is not satisfied. The remedy is a wider `v` and/or `Z`.

References

Golan, A. (2008). *Information and Entropy Econometrics - A Review and Synthesis*. Foundations and Trends in Econometrics, 2(1-2), 1-145. Section 6.1. Golan, A., Judge, G. and Miller, D. (1996). *Maximum Entropy Econometrics: Robust Estimation with Limited Data*. Wiley. Chapter 6.

See Also

`inverse_ce`, `inverse_noise` for related information-theoretic estimators.

Examples

```
set.seed(1)
d <- data.frame(x1 = rnorm(100), x2 = rnorm(100))
d$y <- 2 - 1.5 * d$x1 + 0.8 * d$x2 + rnorm(100, sd = 0.7)
fit <- linreg(y ~ x1 + x2, data = d, Z = seq(-20, 20, length.out = 5))
summary(fit)
```

linreg_iv *Stochastic-Moments GME Estimator for IV Regression*

Description

Fits the linear model $y = X\beta + e$ by the (relaxed) stochastic-moments Generalized Maximum Entropy estimator of Golan (2008, pp. 89-91), identified by instrument moments $IV'(y - X\beta - e) = 0$. Each coefficient is reparameterized on a bounded signal support, $\beta_k = \sum_m z_{km} p_{km}$, and each error on a noise support, $e_i = \sum_j v_j w_{ij}$; the signal and noise entropies are maximized subject to the instrument moments, with weight $\nu \in (0, 1)$. It is the instrumental-variables sibling of `linreg`.

Usage

```
linreg_iv(
  y,
  X,
  IV,
  Z,
  p0 = NULL,
  v = NULL,
  w0 = NULL,
  nu = 0.5,
  se_method = c("sandwich", "delta", "bootstrap", "none"),
  control = list(),
  boot = 200L
)
```

Arguments

`y`	Numeric response vector of length N.
`X`	N-by-K design matrix (include an intercept column if wanted).
`IV`	N-by-P instrument matrix, `P >= K`. For exogenous regressors, use the regressor as its own instrument.
`Z`	Coefficient (signal) support: a K-by-M matrix whose row k is the support for β_k (as in `linreg`).
`p0`	Optional K-by-M signal prior (rows strictly positive, summing to 1); default uniform.
`v`	Optional error support: `NULL` (default 3-point symmetric grid on $\pm 3\,\mathrm{sd}(y)$, the three-sigma rule), a single whole number giving the number of support points, or an explicit numeric vector. Unlike `linreg`, this default does *not* need to widen with the sample size; see *Choosing the supports*.
`w0`	Optional N-by-J error prior (rows strictly positive, summing to 1); default uniform.
`nu`	Entropy weight in $(0, 1)$ on the noise term, hence the weight on the signal is $1 - \nu$; default 0.5 (equal weight on both the signal and the noise).

`se_method`	Standard-error method: `"sandwich"` (default, robust), `"delta"` (classical), `"bootstrap"` (pairs resampling), or `"none"` (skip; `se_beta`/`vcov` left `NULL`).
`control`	Named list merged over defaults and passed to `optim` (BFGS): `maxit` (default 1000) and `reltol` (default 1e-12). `fnscale` is forced to -1 (the dual is maximized).
`boot`	Number of bootstrap resamples when `se_method = "bootstrap"` (default 200).

Details

The concentrated dual is maximized over the Lagrange multipliers λ (one per instrument moment): with $p_{km} \propto p0_{km} \exp(z_{km}(X'IV\lambda)_k/\nu)$ and $w_{ij} \propto w0_{ij} \exp((IV\lambda)_i v_j/(1-\nu))$, the gradient is the instrument moment $IV'(y - X\beta - e)$. Supports just-identified (`ncol(IV) == ncol(X)`) and over-identified (`ncol(IV) > ncol(X)`) systems. The instruments are standardized internally for numerical conditioning (β is invariant to instrument scaling); `lambda` is reported on the original scale. `coef()` returns β.

As the signal support widens the GME estimate approaches the exact (2SLS-type) IV solution; narrower supports shrink β toward the support centers.

Standard errors. Because $\hat{\lambda}$ solves the instrument moments, $\hat{\beta} = Zp(\hat{\lambda})$ is a Z-estimator with sandwich covariance $\mathrm{Var}(\hat{\beta}) = J_\beta A^{-1} \Omega A^{-1} J_\beta'$, where A is the dual Hessian and $J_\beta = \partial\beta/\partial\lambda$. The "meat" Ω is set by `se_method`: `"sandwich"` (default, robust HC0 $IV'\mathrm{diag}(r^2)IV$), `"delta"` (classical homoskedastic $\hat{\sigma}^2 IV'IV$), or `"bootstrap"` (pairs resampling). The robust sandwich matches a Monte-Carlo sampling SD and reduces to the 2SLS robust SE as the support widens; the classical delta assumes homoskedasticity. These are *asymptotic* SEs: the sampling distribution of $\hat{\beta}$ is support-bounded and can be skewed, so symmetric Wald intervals are approximate with weak instruments or small n.

Value

An object of class `c("linreg_iv", "infometrics")`, which is a list containing the following components:

`coefficients`, `b_hat` Numeric vector of length K: the estimated coefficients $\hat{\beta} = Z\hat{p}$ (two names for the same object). Extracted by `coef`.

`lambda_hat` Numeric vector of length P: the estimated Lagrange multipliers, one per instrument moment, reported on the original instrument scale.

`p_hat` K-by-M matrix of estimated signal probabilities.

`w_hat` N-by-J matrix of estimated noise probabilities.

`e_hat`, `e` Numeric vector of length N: the estimated noise, $e_i = \sum_j v_j w_{ij}$ (two names for the same object).

`fitted.values` Numeric vector of length N: $X\hat{\beta}$. Extracted by `fitted`.

`residuals` Numeric vector of length N: $y - X\hat{\beta}$. Extracted by `residuals`.

`se_beta` Numeric vector of length K: standard errors of $\hat{\beta}$, or `NULL` when `se_method = "none"`.

`vcov` K-by-K covariance matrix of $\hat{\beta}$, or `NULL` when `se_method = "none"`. Extracted by `vcov`.

`se_method`, `boot` The standard-error method used and, for the bootstrap, the number of resamples.

`H_signal` Numeric vector of length K: the per-coefficient signal entropies.

`S` Single number: the normalized signal entropy.

`objective`, `value` Single number: the maximized dual objective (two names for the same value).

`moment_resid` Single number: the largest absolute instrument moment at the optimum. Not normalized by N – see *Note on moment_resid*.

`converged` Logical: `TRUE` when `optim` reported convergence.

`convergence` Integer: the raw `optim` convergence code.

`method` Character string naming the solver, `"dual"`.

`Z`, `p0`, `v`, `w0`, `nu`, `X`, `y`, `IV` The resolved inputs.

`N`, `K`, `P`, `M` Integers: the numbers of observations, coefficients, instruments and signal support points.

`call` The matched call.

Choosing the supports

The **signal** support `Z` is the consequential input here. It must contain the true coefficients: if it does not, $\hat{\beta}$ is pinned to the boundary of `Z` (for example, a support of ± 1 returns $\hat{\beta}_k = 1$ for a coefficient whose true value is 1.5). Widening `Z` moves the estimate toward the exact (2SLS-type) IV solution; narrowing it shrinks β toward the support centers.

The **error** support `v` is far less critical, and – unlike `linreg` and `inverse_noise` – its default does not need to grow with the sample size. Those estimators impose one dual condition per observation, $y_t = x_t'\beta + e_t$, so every realized error must fit inside `v`; because the largest of T errors grows like $\sigma\sqrt{2\log T}$, a fixed three-sigma support eventually becomes infeasible and their duals become unbounded. `linreg_iv()` instead imposes only the P aggregate instrument moments $IV'(y - X\beta - e) = 0$, so the noise never has to absorb individual residuals and no per-observation feasibility condition arises. In practice $\hat{\beta}$ is nearly invariant to the width of `v`, and large samples pose no difficulty: the multipliers stay small and the estimate tracks the exact IV solution.

Note on `moment_resid`

`moment_resid` is $\max_p |IV'(y - X\beta - e)|_p$ evaluated on the standardized instruments. It is a *sum* over the N observations and is not divided by N, so its magnitude grows with the sample size even when the fit is excellent; judge it relative to N (or compare fits of the same size) rather than against a fixed threshold.

References

Golan, A. (2008). *Information and Entropy Econometrics - A Review and Synthesis*. Foundations and Trends in Econometrics, 2(1-2), 1-145. Pages 89-91. Golan, A., Judge, G. and Miller, D. (1996). *Maximum Entropy Econometrics: Robust Estimation with Limited Data*. Wiley.

See Also

`linreg` for the (non-IV) GME/GCE regression.

Examples

```
set.seed(1)
n  <- 200L
z  <- rnorm(n)                          # instrument
u  <- rnorm(n)                          # structural error
xe <- 0.7 * z + u + rnorm(n)            # endogenous regressor (corr. with u)
y  <- 1 + 1.5 * xe + u                  # true slope 1.5; OLS biased
X  <- cbind(1, xe)                      # intercept + endogenous regressor
IV <- cbind(1, z)                       # intercept is its own instrument
Z  <- matrix(c(-10, 0, 10), nrow = 2, ncol = 3, byrow = TRUE)  # signal support
fit <- linreg_iv(y, X, IV, Z)
coef(fit)                               # slope ~ 1.5 (vs OLS biased upward)
```

---

make_support *Construct a Symmetric Support Space*

Description

Creates an M-point symmetric support vector centered at zero (or a specified center) over a given half-range. Support spaces are required inputs for the GME and GCE estimators, bounding the parameter vector beta and the error vector epsilon.

Usage

```
make_support(half_range, M = 5L, center = 0)
```

Arguments

half_range	Positive numeric. The half-width of the support interval. The support spans from center - half_range to center + half_range.
M	Positive integer. Number of support points. Must be >= 2. The default of 5 (a 5-point support) is commonly used in the GME literature.
center	Numeric. Center of the support interval (default 0).

Details

In the GME framework (Golan, 2008, Section 6.1), each parameter beta_k is reparameterized as the expected value of a random variable defined on a bounded support z_k = (z_{k1}, ..., z_{kM}). The support must contain the true parameter value. A common default is a symmetric 5-point grid with half-range set to 3 standard deviations of the OLS residuals (for the error support V) or 2 times the absolute OLS estimate (for the signal support Z).

The complexity of the GME dual model is invariant to M (the number of support points), since the real parameters are the Lagrange multipliers whose dimension equals T (the number of observations), not K*M.

Value

A numeric vector of length M giving the support points, evenly spaced from `center - half_range` to `center + half_range`.

References

Golan, A. (2008). Information and Entropy Econometrics. *Foundations and Trends in Econometrics*, **2**(1-2), 1-145. Section 6.1.

Examples

```
# 5-point support for error terms: [-3, -1.5, 0, 1.5, 3]
make_support(half_range = 3, M = 5)

# 3-point support centered at a non-zero value: [0, 5, 10]
make_support(half_range = 5, M = 3, center = 5)

# 7-point support for a coefficient bounded in [-2, 2]
make_support(half_range = 2, M = 7)
```

`margins` *Marginal Effects*

Description

Generic for average (or per-observation) marginal effects of a fitted model.

Usage

```
margins(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Passed to methods.

Value

A method-specific object of marginal effects.

See Also

`margins.multinomial_gce`

`margins.mixed_gce` *Marginal Effects for a Mixed GCE Fit*

Description

Partial derivatives of the signal probabilities with respect to each design slot, holding λ and ρ fixed:

$$\partial p_{ij}/\partial x_{ijk} = (\lambda_{kj}/\nu)\,\mathrm{Var}_\theta(s)_{ij}, \qquad \mathrm{Var}_\theta(s)_{ij} = \sum_m s_m^2 \theta_{ijm} - p_{ij}^2.$$

Usage

```
## S3 method for class 'mixed_gce'
margins(
  object,
  average = TRUE,
  se = FALSE,
  se_method = c("sandwich", "bootstrap"),
  B = 500L,
  ...
)
```

Arguments

`object`	A `mixed_gce` object.
`average`	Logical (default `TRUE`). If `TRUE`, return the K x J matrix of marginal effects averaged over observations; if `FALSE`, the full N x J x K array of per-observation derivatives.
`se`	Logical (default `FALSE`). If `TRUE` (requires `average = TRUE`), return a `margins_gce` object with estimates, standard errors, z-values, and p-values.
`se_method`	`"sandwich"` (default) or `"bootstrap"`; see Details.
`B`	Integer number of bootstrap resamples when `se_method = "bootstrap"` (default 500).
`...`	Unused.

Details

With `se = TRUE` (and `average = TRUE`) standard errors for the average marginal effects are attached. These are a *sampling* quantity, **unrelated** to the Fano `fano_bounds` (which bound classification error). `se_method = "sandwich"` (default) uses the robust covariance $(-H)^{-1}\hat{V}(-H)^{-1}$ of the dual multipliers, with $\hat{V} = \sum_i g_i g_i'$ the per-observation score outer products, sandwiched with a numerical Jacobian of the average effects. This matches a row bootstrap; the naive Hessian-inverse would overstate by ~7-9x for this model (the support reparameterization plus per-observation ρ block make `solve(-H)` a poor sampling-covariance estimate), so it is not used. `se_method = "bootstrap"` resamples observations, refits, and takes the across-resample SD.

Value

With `se = FALSE`: a K x J matrix (average effects) or an N x J x K array. With `se = TRUE`: a `margins_gce` object.

See Also

`fano_bounds` for information-theoretic error bounds.

`margins.multinomial_gce` *Marginal Effects for a Multinomial GCE Fit*

Description

Partial derivatives of the signal probabilities with respect to each column of `X`:

$$\partial p_{ij} / \partial x_{ik} = (p_{ij}/\nu)\,(\lambda_{kj} - \textstyle\sum_l p_{il}\lambda_{kl}).$$

The reference alternative's column is included like any other; its "intercept" effect (derivative w.r.t. a constant) is rarely interpreted.

Usage

```
## S3 method for class 'multinomial_gce'
margins(
  object,
  average = TRUE,
  se = FALSE,
  se_method = c("sandwich", "delta", "bootstrap"),
  B = 500L,
  ...
)
```

Arguments

`object`	A `multinomial_gce` object.
`average`	Logical (default `TRUE`). If `TRUE`, return the K x J matrix of marginal effects averaged over observations; if `FALSE`, the full N x J x K array of per-observation derivatives.
`se`	Logical (default `FALSE`). If `TRUE` (requires `average = TRUE`), return a `margins_gce` object carrying the estimates, standard errors, z-values, and p-values.
`se_method`	`"sandwich"` (default), `"delta"`, or `"bootstrap"`; see Details.
`B`	Integer number of bootstrap resamples when `se_method = "bootstrap"` (default 500).
`...`	Unused.

Details

With `se = TRUE` (and `average = TRUE`) standard errors for the average marginal effects are attached. These are a *sampling* quantity, **unrelated** to the Fano `fano_bounds` (which bound classification error, not sampling variance). All methods sandwich a numerical Jacobian J of the average effects in the free multipliers with a covariance of $\hat{\lambda}$:

- `se_method = "sandwich"` (default) uses the robust covariance $(-H)^{-1}\hat{V}(-H)^{-1}$ with $\hat{V} = \sum_i g_i g_i'$ the per-observation dual scores. This matches a row bootstrap and is the accurate choice under the GME's finite-support regularization.
- `se_method = "delta"` uses the naive Hessian inverse $V =$`vcov(object)` (the classical Golan 2008 sec. 7.3 observed-information SE); being tied to $(-H)^{-1}$ it is *conservative* (~1.35x the bootstrap here).
- `se_method = "bootstrap"` resamples observations, refits, and takes the across-resample SD (accurate, slower).

Value

With `se = FALSE`: a K x J matrix (average effects) or an N x J x K array. With `se = TRUE`: a `margins_gce` object.

See Also

`fano_bounds` for information-theoretic error bounds.

---

markov_ce *Cross-Entropy Estimation of a Markov Transition/Balancing Matrix (Panel)*

Description

Estimates a first-order, stationary row-stochastic matrix P ($P_{kj} = k \to j$) linking simplex-valued state vectors over time, from a balanced long panel, by Cross-Entropy (Golan 2008, Section 7.7.1, eqs. 7.25-7.27). States may be one-hot membership indicators or compositional shares (a panel/time-indexed extension of matrix balancing). Data are supplied in long form (one row per unit-period) and reshaped internally; transitions $t-1 \to t$ are formed within each unit.

Usage

```
markov_ce(
  data,
  id,
  time,
  states,
  covariates = NULL,
  p0 = NULL,
  control = list()
)
```

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

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

## S3 method for class 'markov_ce'
fano_bounds(object, ...)

## S3 method for class 'markov_ce'
margins(
  object,
  average = TRUE,
  se = FALSE,
  se_method = c("sandwich", "bootstrap"),
  B = 500L,
  ...
)

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

## S3 method for class 'markov_ce'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

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

Arguments

`data`	A balanced long-format `data.frame`, one row per unit-period.
`id`, `time`	Names of the unit-identifier and time columns.
`states`	Character vector of state columns; each row over them is non-negative and sums to 1. `K = length(states)`.
`covariates`	Optional character vector of covariate columns (`S = length(covariates)`); default uses the lagged-state indicator.
`p0`	Optional `K`-by-`K` prior transition matrix, strictly positive with rows summing to 1. Defaults to uniform.
`control`	A named list merged over the defaults and passed to `optim` (BFGS): `maxit` (default 1000) and `reltol` (default 1e-12). `fnscale` is forced to -1 (the dual is maximized).
`object`, `x`	A `markov_ce` object.
`...`	Unused.
`average`	Logical; if `TRUE` (default) return the origin-weighted average marginal effect (`S x K`); if `FALSE` the full `K x K x S` array of per-origin effects.

se	Logical (default FALSE); if TRUE (with average = TRUE) attach standard errors.
se_method	"sandwich" (default, robust and analytic) or "bootstrap" (resample units, resolve). See margins.markov_gce.
B	Integer bootstrap resamples when se_method = "bootstrap" (default 500).
digits	Significant digits to print.

Details

Optional covariates z enter lead/lagged like the states: per transition the lagged states X pair with lagged covariates Z_2 and the lead states Y with lead covariates Z_1, forming the cross moments $A = \sum X' Z_2$ (K x S) and $M = \sum Z_1' Y$ (S x K), S =number of covariates. The concentrated dual

$$\ell(\lambda) = \sum_{s,j} M_{sj}\lambda_{sj} - \sum_k \log \Omega_k(\lambda), \quad \hat{P}_{kj} = p0_{kj} \exp(\textstyle\sum_s A_{ks}\lambda_{sj})/\Omega_k$$

is maximized over the S x K multipliers. With no covariates the conditioning is the lagged-state indicator ($S = K$, transition counts).

When the conditioning is full rank (A = X'X}, the common indicator case), the moment condition \eqn{A'P = M exactly determines P as the empirical/regression transition matrix, so the prior is irrelevant for identified origin states. Only **under-identified** rows ($\mathrm{rank}(A) < K$, e.g. a state never observed as an origin) fall back to the prior (maximum entropy when p0 is uniform); see the rank warning and the normalized entropy S in summary().

Standard errors of λ (se_lambda) use the analytic Hessian of the CE log-partition (Golan 2008, eq. 4.7), $H_{(s,j),(s',j')} = \sum_k A_{ks}A_{ks'}P_{kj}(\delta_{jj'} - P_{kj'})$ – the covariance of the moment functions under $\hat{P}$. Note that H^{-1} *alone* overstates the SE (~$\sqrt{n}$), because the moments A, M are cross-moment sums ($H = n\hat{V}$); the reported SE is the robust sandwich $H^{-1}\hat{V}H^{-1}$ (Golan 2008, Sec. 3.3), which matches a unit bootstrap. Because λ is identified only up to a per-conditioning additive shift, se_lambda is relative to a reference destination (lambda_ref = 1, that column NA); a (near-)boundary fit ($P_{kj} \in \{0, 1\}$) yields NA (an infinite log-odds SE).

Value

An object of class c("markov_ce", "infometrics"), which is a list containing the following components:

p, p_hat K-by-K row-stochastic transition matrix, $\hat{P}_{kj} = k \to j$ (two names for the same object). Extracted by coef.

lambda, lambda_hat S-by-K matrix of estimated Lagrange multipliers, reported on the original covariate scale (two names for the same object).

se K-by-K matrix of large-sample multinomial standard errors of $\hat{P}$.

se_lambda S-by-K matrix of analytic-Hessian sandwich standard errors of λ. The reference-destination column is NA, as are entries at a (near-)boundary fit.

lambda_ref Integer: the index of the reference destination state used to normalize se_lambda.

vcov_lambda Covariance matrix of the free (non-reference) elements of λ, or NULL if the sandwich could not be formed.

n_from Numeric vector of length K: the total weight of each origin state across transitions.

`states` Character vector of length K: the state labels.

`se_data` List of the cross moments, lead/lagged design blocks and scaling used by `margins.markov_ce`, or `NULL` when no covariates were supplied.

`n_transitions` Integer: the number of unit-period transitions, $N(T-1)$.

`state_type` Character string, `"indicator"` or `"shares"`, describing the supplied state columns.

`fitted.values` Matrix of fitted lead states, $X\hat{P}$, one row per transition. Extracted by `fitted`.

`residuals` Matrix of the same dimension: $Y - X\hat{P}$. Extracted by `residuals`.

`entropy`, `H_signal` Single number: the mean row entropy of $\hat{P}$ (two names for the same value).

`S` Single number: the prior-relative normalized entropy, the mean row entropy of $\hat{P}$ divided by that of `p0`.

`moment_residual` Single number: $\max|M - A'\hat{P}|$ at the optimum.

`objective`, `value` Single number: the maximized dual objective (two names for the same value).

`prior` The resolved K-by-K prior `p0`.

`converged` Logical: `TRUE` when `optim` reported convergence.

`convergence` Integer: the raw `optim` convergence code.

`method` Character string naming the solver, `"dual"`.

`call` The matched call.

References

Golan, A. (2008). *Information and Entropy Econometrics - A Review and Synthesis*. Foundations and Trends in Econometrics, 2(1-2), 1-145. Section 7.7.1.

See Also

`matrix_ce`, `matrix_gce`

Examples

```
set.seed(1)
P_true <- matrix(c(.6,.3,.1, .2,.5,.3, .1,.3,.6), nrow = 3, byrow = TRUE)
N <- 150L; Tn <- 5L; K <- 3L
s  <- sample(K, N, replace = TRUE)
id <- rep(seq_len(N), each = Tn); tm <- rep(seq_len(Tn), N)
state <- integer(N * Tn)
for (t in seq_len(Tn)) {
  state[tm == t] <- s
  s <- vapply(s, function(k) sample(K, 1L, prob = P_true[k, ]), integer(1))
}
oh <- diag(K)[state, ]
panel <- data.frame(i = id, t = tm, y1 = oh[, 1], y2 = oh[, 2], y3 = oh[, 3])
fit <- markov_ce(panel, id = "i", time = "t", states = c("y1", "y2", "y3"))
coef(fit)
```

markov_gce *Generalized Cross-Entropy Estimation of a Markov Matrix with Noisy Moments*

Description

Estimates a row-stochastic $K \times K$ matrix P linking simplex- valued state vectors over time, conditioning on covariates, by Generalized Cross-Entropy from a balanced long panel (Golan 2008, Section 7.7.1, eqs. 7.25a and 7.28). Unlike markov_ce (which imposes the moments exactly), each conditional moment carries an additive error $\varepsilon_{itj} = \sum_m w_{itjm} v_m$ on a support v symmetric on $[-1, 1]$. This makes over-identified moment systems – in particular time-varying covariates such as household income – feasible, where the exact problem is unbounded.

Usage

```
markov_gce(
  data,
  id,
  time,
  states,
  covariates,
  v = NULL,
  nu = 0.5,
  p0 = NULL,
  w0 = NULL,
  control = list()
)

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

## S3 method for class 'markov_gce'
fano_bounds(object, ...)

## S3 method for class 'markov_gce'
margins(
  object,
  average = TRUE,
  se = FALSE,
  se_method = c("sandwich", "bootstrap"),
  B = 500L,
  ...
)

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

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

## S3 method for class 'markov_gce'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

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

Arguments

data	A balanced long-format data.frame, one row per unit-period.
id, time	Names of the unit-identifier and time columns.
states	Character vector of state columns; each row over them is non-negative and sums to 1. K = length(states).
covariates	Character vector of covariate columns (S = length(covariates)); required. Time-varying covariates are supported.
v	Numeric error support, symmetric about 0 with at least two points; default c(-1, 0, 1) (Golan's symmetric $[-1, 1]$ support).
nu	Signal/noise weight in $(0, 1)$; default 0.5. Smaller nu penalizes the noise more strongly (driving $\hat{\varepsilon} \to 0$, toward the exact markov_ce); larger nu lets the noise absorb more of the moment discrepancy.
p0	Optional K-by-K prior transition matrix, strictly positive with rows summing to 1. Defaults to uniform.
w0	Optional error prior on the support (length length(v), positive, summing to 1). Defaults to uniform (mean-zero error prior).
control	A named list merged over the defaults and passed to optim (BFGS): maxit (default 1000) and reltol (default 1e-12). fnscale is forced to -1 (the dual is maximized).
object, x	A markov_gce object.
...	Unused.
average	Logical; if TRUE (default) return the origin-weighted average marginal effect (S x K); if FALSE the full K x K x S array of per-origin effects.
se	Logical (default FALSE); if TRUE (with average = TRUE) attach standard errors.
se_method	"sandwich" (default) or "bootstrap". The naive Hessian-inverse delta overstates ~4-5x here; the robust sandwich $(-H)^{-1}\hat{V}(-H)^{-1}$ matches the "bootstrap" (which resamples units and re-solves the dual).
B	Integer bootstrap resamples when se_method = "bootstrap" (default 500).
digits	Significant digits to print.

Details

The ν-weighted concentrated dual maximized over the $S \times K$ multipliers λ is

$$\ell(\lambda) = \sum_{t \geq 2} \sum_{j} \sum_{i,s} y_{itj} z_{its} \lambda_{sj} - \nu \sum_{k} \log \Omega_k(\lambda) - (1 - \nu) \sum_{i,t,j} \log \Psi_{itj}(\lambda),$$

with $\hat{p}_{kj} \propto p0_{kj} \exp(\nu^{-1} \sum A_{ks}\lambda_{sj})$ and $\hat{w}_{itjm} \propto w0_m \exp((1-\nu)^{-1} \sum_s z_{its} v_m \lambda_{sj})$. The first-order conditions are the noisy moments $M - A'\hat{P} - \sum_{i,t} z\,\hat{\varepsilon} = 0$. The exact estimator `markov_ce` is recovered in two limits: as the support $v \to 0$ (the slack vanishes mechanically), and as $\nu \to 0$ (the noise-divergence term dominates, forcing $\hat{\varepsilon} \to 0$). Conversely, a wide support or $\nu \to 1$ lets the errors absorb the moments and pulls $\hat{P}$ toward the prior.

Standard errors of λ (`se_lambda`) use the analytic Hessian of the dual (Golan 2008, eq. 4.7): the signal covariance $\nu^{-1} \sum_k A_{ks} A_{ks'} P_{kj}(\delta_{jj'} - P_{kj'})$ plus the noise covariance $(1-\nu)^{-1} \sum_{i,t} z_{its} z_{its'} \mathrm{Var}_w(v)_{itj}$ (block-diagonal in the destination j). The noise term makes the Hessian full rank – λ is identified, so no reference normalization is needed (unlike `markov_ce`). The reported SE is the robust **unit-clustered** sandwich $H^{-1}\hat{V}H^{-1}$ (Golan 2008, Sec. 3.3), which matches a unit bootstrap; the naive H^{-1} and the non-clustered sandwich are conservative because covariate scores are correlated within a unit.

Value

An object of class `c("markov_gce", "infometrics")`, which is a list containing the following components:

`p`, `p_hat` K-by-K row-stochastic transition matrix, $\hat{P}_{kj} = k \to j$ (two names for the same object). Extracted by `coef`.

`lambda`, `lambda_hat` S-by-K matrix of estimated Lagrange multipliers, reported on the original covariate scale (two names for the same object).

`epsilon` Matrix of fitted per-transition errors $\hat{\varepsilon}_{itj}$, one row per transition and one column per destination state.

`se` K-by-K matrix of large-sample multinomial standard errors of $\hat{P}$.

`se_lambda` S-by-K matrix of analytic-Hessian, unit-clustered sandwich standard errors of λ. All entries are reported: the noise identifies λ, so no reference normalization is needed.

`vcov_lambda` Covariance matrix of λ, or `NULL` if the sandwich could not be formed.

`states` Character vector of length K: the state labels.

`se_data` List of the cross moments, lead/lagged design blocks and scaling used by `margins.markov_gce`.

`n_transitions` Integer: the number of unit-period transitions, $N(T-1)$.

`n_from` Numeric vector of length K: the total weight of each origin state across transitions.

`nu` Single number: the signal/noise weight used.

`v`, `w0` The resolved error support and its prior.

`H_p`, `H_signal`, `entropy` Single number: the mean row entropy of $\hat{P}$ (three names for the same value).

`H_w` Single number: the mean noise entropy $H(\hat{w})$.

`S` Single number: the prior-relative normalized entropy, the mean row entropy of $\hat{P}$ divided by that of `p0`.

`fitted.values` Matrix of fitted lead states, $X\hat{P}$, one row per transition. Extracted by `fitted`.

`residuals` Matrix of the same dimension: $Y - X\hat{P}$. Extracted by `residuals`.

`moment_residual` Single number: $\max|M - A'\hat{P}|$, the *exact*-moment gap. Expected to be nonzero – it is absorbed by the noise.

`foc_residual` Single number: $\max |M - A'\hat{P} - \sum z\,\hat{\varepsilon}|$, the noisy first-order condition. Near 0 at a healthy optimum.

`objective`, `value` Single number: the maximized dual objective (two names for the same value).

`prior` The resolved K-by-K prior `p0`.

`converged` Logical: `TRUE` when `optim` reported convergence.

`convergence` Integer: the raw `optim` convergence code.

`method` Character string naming the solver, `"dual"`.

`call` The matched call.

Choosing nu and v

There is no universally correct default. The noise enters per observation and the literal support $v = (-1, 0, 1)$ is permissive relative to share-valued moment contributions, so at a wide support or large `nu` the errors can absorb the signal and pull $\hat{P}$ toward the prior. In controlled recovery experiments the estimate approaches the exact `markov_ce` solution monotonically as the support narrows ($v \to 0$) or as `nu` decreases toward 0, though an ultra-narrow support can make the over-identified exact problem numerically unbounded (non-convergence). Users should report sensitivity to `nu` and `v` for their data rather than rely on a single setting.

References

Golan, A. (2008). *Information and Entropy Econometrics - A Review and Synthesis*. Foundations and Trends in Econometrics, 2(1-2), 1-145. Section 7.7.1, eqs. (7.25a), (7.28).

See Also

`markov_ce`, `matrix_gce`

Examples

```
set.seed(1)
P_true <- matrix(c(.6,.3,.1, .2,.5,.3, .1,.3,.6), nrow = 3, byrow = TRUE)
N <- 120L; Tn <- 5L; K <- 3L
Y <- matrix(rgamma(N * K, 1), N, K); Y <- Y / rowSums(Y); inc <- rnorm(N)
id <- rep(seq_len(N), each = Tn); tm <- rep(seq_len(Tn), N)
sh <- matrix(0, N * Tn, K); income <- numeric(N * Tn); infl <- numeric(N * Tn)
for (t in seq_len(Tn)) {
  idx <- which(tm == t); inc <- 0.8 * inc + rnorm(N, 0, 0.5)
  sh[idx, ] <- Y; income[idx] <- inc; infl[idx] <- 0.02 * t
  Y <- (Y %*% P_true) * exp(matrix(rnorm(N * K, 0, 0.03), N, K)); Y <- Y / rowSums(Y)
}
panel <- data.frame(i = id, t = tm, y1 = sh[, 1], y2 = sh[, 2], y3 = sh[, 3],
                    income = income, infl = infl)
fit <- markov_gce(panel, id = "i", time = "t", states = c("y1","y2","y3"),
                  covariates = c("income", "infl"), v = c(-0.2, 0, 0.2),
                  nu = 0.2)
coef(fit)
```

`matrix_ce` *Cross-Entropy Matrix Balancing*

Description

Recovers a non-negative matrix p (each column a probability distribution) from column weights `x` and row-weighted aggregates `y` by minimizing the cross-entropy $D(p\|p0)$ relative to a prior `p0`, solving the concentrated dual of Golan (2008), Section 7.2. With a uniform `p0` this reduces to Maximum Entropy. The model is

$$y_i = \sum_j p_{ij} x_j, \qquad \sum_i p_{ij} = 1,$$

with solution $\hat{p}_{ij} = p0_{ij} \exp(\hat{\lambda}_i x_j)/\Omega_j$ and $\Omega_j(\lambda) = \sum_i p0_{ij} \exp(\lambda_i x_j)$.

Usage

```
matrix_ce(y, x, p0 = NULL, control = list())

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

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

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

## S3 method for class 'matrix_ce'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

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

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

Arguments

`y`	Numeric vector of length n: the row-weighted aggregates $y_i = \sum_j p_{ij} x_j$.
`x`	Numeric vector of length m: the column weights (in `print`, the fitted `matrix_ce` object).
`p0`	Optional n-by-m matrix of strictly-positive prior probabilities whose columns sum to 1. Defaults to uniform (`1/n`).
`control`	A named list merged over the defaults and passed to `optim` (BFGS): `maxit` (default 500) and `reltol` (default 1e-12).

object	A matrix_ce object.
...	Additional arguments passed to or from methods (currently ignored).
digits	Number of significant digits to print.

Details

The dual is minimised here (convex) to match the package convention used by inverse_ce:

$$\ell(\lambda) = -\sum_i \lambda_i y_i + \sum_j \log \Omega_j(\lambda),$$

minimised over the n row multipliers (equivalent to maximising the column entropies subject to the data). The problem is typically under-determined ($n \times m$ unknowns, $n + m$ constraints), so the returned matrix is the minimum cross-entropy matrix relative to p0 consistent with the data (the maximum-entropy matrix when p0 is uniform) – not necessarily the matrix that generated y. To bias the solution toward a known target, pass that target as p0.

The normalized entropy is $S = H(\hat{p})/H(p0)$ (Golan §7.5), the ratio of the total Shannon entropy to the prior's entropy. For a uniform p0 (Maximum Entropy) $H(p0) = m \log n$ (the absolute maximum, m columns uniform over n rows), so $S \in [0, 1]$ with $S = 1$ when every column is uniform; for a non-uniform prior S is relative to that prior. summary() reports the §7.5 information measures built on S: the information index $I = 1 - S$, pseudo-$R^2 = 1 - S$, the per-column normalized entropy $S(p_j) = H(p_j)/H(p0_j)$, and the entropy-ratio test $W = 2[H(p0) - H(\hat{p})] \sim \chi^2_{n-1}$ for $H_0 : P = P0$.

Value

An object of class c("matrix_ce", "infometrics"), which is a list containing the following components:

p, p_hat n-by-m matrix of estimated column-stochastic probabilities (two names for the same object). Extracted by coef.

lambda, lambda_hat Numeric vector of length n: the estimated row multipliers (two names for the same object).

fitted, fitted.values Numeric vector of length n: $\sum_j \hat{p}_{ij} x_j$ (two names for the same object). Extracted by fitted.

residuals Numeric vector of length n: $y - \hat{y}$, the row-aggregate residual. Extracted by residuals.

entropy, H_signal Single number: the Shannon entropy $H(\hat{p})$ (two names for the same value).

entropy_p0 Single number: the prior entropy $H(p_0)$.

S Single number: the prior-relative normalized entropy $H(\hat{p})/H(p_0)$, which lies in [0, 1] for a uniform p0.

cross_entropy Single number: the cross-entropy $D(\hat{p}\|p_0)$ at the optimum.

objective, value Single number: the minimised dual objective (two names for the same value).

prior The resolved n-by-m prior p0.

converged Logical: TRUE when optim reported convergence.

convergence Integer: the raw optim convergence code.

method Character string naming the solver, "dual".

y, x, n, m The resolved inputs and their lengths n = length(y), m = length(x).

call The matched call.

Methods (by generic)

- `coef(matrix_ce)`: Estimated probability matrix $\hat{p}$ (n x m).
- `fitted(matrix_ce)`: Fitted row aggregates $\sum_j p_{ij} x_j$.
- `residuals(matrix_ce)`: Moment residuals $y-$ fitted.
- `print(matrix_ce)`: Compact printed display.
- `summary(matrix_ce)`: Summary with the Section-7.5 information measures, the entropy-ratio test for the signal matrix, and a moment-fit table.

Functions

- `print(summary.matrix_ce)`: Print method for the summary object.

References

Golan, A. (2008). *Information and Entropy Econometrics - A Review and Synthesis*. Foundations and Trends in Econometrics, 2(1-2), 1-145. Section 7.2.

See Also

`inverse_ce` for the formula-interface pure-moment ME/CE estimator.

Examples

```
P <- sweep(matrix(1:6, ncol = 2), 2, colSums(matrix(1:6, ncol = 2)), "/")
x <- c(3, 2)
y <- as.vector(P %*% x)
fit <- matrix_ce(y, x)
fit$p
summary(fit)
```

`matrix_gce` *Generalized Cross-Entropy Matrix Balancing (Stochastic Moments)*

Description

Noisy extension of `matrix_ce` (Golan 2008, Section 7.4). The aggregates are treated as stochastic, $y_i = \sum_j p_{ij} x_j + e_i$ with $e_i = \sum_h v_h w_{ih}$, and the cross-entropy of both the signal `p` (columns sum to 1) and the noise `w` (rows sum to 1) is minimized relative to priors `p0`, `w0`.

Usage

```
matrix_gce(y, x, p0 = NULL, w0 = NULL, v = NULL, nu = 0.5, control = list())

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

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

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

## S3 method for class 'matrix_gce'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

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

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

Arguments

y, x	Numeric vectors (lengths n and m), each normalized to [0, 1] so the implied errors lie in [-1, 1].
p0	Optional n-by-m signal prior, columns summing to 1, strictly positive. Default uniform.
w0	Optional n-by-H noise prior, rows summing to 1, strictly positive. Default uniform; its column count must match `length(v)`.
v	Noise support, symmetric around 0 in [-1, 1]: `NULL` (default `c(-1, 0, 1)`), a single whole number read as a support-point count `H >= 2`, or an explicit numeric vector (used as given).
nu	Entropy weight in $(0, 1)$; default 0.5.
control	A named list merged over the defaults and passed to `optim` (BFGS): `maxit` (default 500) and `reltol` (default 1e-12).
object	A `matrix_gce` object (`x` in `print`).
...	Additional arguments passed to or from methods (currently ignored).
digits	Significant digits to print.

Details

The dual is minimised here (convex) to match the package convention used by `inverse_noise` and `matrix_ce`:

$$\ell(\lambda) = -\sum_i \lambda_i y_i + \nu \sum_j \log \Omega_j(\lambda) + (1-\nu) \sum_i \log \Psi_i(\lambda),$$

minimised over the n row multipliers, with $\hat{p}_{ij} \propto p0_{ij} \exp(\lambda_i x_j / \nu)$ (normalised within each column) and $\hat{w}_{ih} \propto w0_{ih} \exp(\lambda_i v_h / (1-\nu))$ (normalised within each row); $\hat{e}_i = \sum_h v_h \hat{w}_{ih}$. Equivalent to maximising the joint signal-plus-noise entropy subject to the stochastic-moment constraints.

The weight $\nu \in (0, 1)$ (an extension to Eq. 7.22) balances prediction against precision by trading signal entropy against noise entropy. At $\nu = 0.5$ this is the equal-weight GCE of Eq. (7.22) and the estimates coincide with it (only the λ scale differs). As the noise support `v` shrinks toward 0 the noise is forced out and the solution returns to `matrix_ce`.

Value

An object of class `c("matrix_gce", "infometrics")`, which is a list containing the following components:

`p`, `p_hat` n-by-m matrix of estimated column-stochastic signal probabilities (two names for the same object). Extracted by `coef`.

`w`, `w_hat` n-by-H matrix of estimated row-stochastic noise probabilities (two names for the same object).

`e` Numeric vector of length n: the estimated noise, $e_i = \sum_h v_h \hat{w}_{ih}$.

`lambda`, `lambda_hat` Numeric vector of length n: the estimated row multipliers (two names for the same object).

`nu` Single number: the entropy weight used.

`fitted.values` Numeric vector of length n: $\hat{P}x$. Extracted by `fitted`.

`residuals` Numeric vector of length n: $y - \hat{P}x$, which equals `e` at a converged optimum. Extracted by `residuals`.

`entropy_p`, `H_signal` Single number: the signal entropy $H(\hat{p})$ (two names for the same value).

`entropy_w` Single number: the noise entropy $H(\hat{w})$.

`entropy_p0`, `entropy_w0` Single numbers: the signal and noise prior entropies $H(p_0)$ and $H(w_0)$.

`S_p`, `S` Single number: the normalized signal entropy $H(\hat{p})/H(p_0)$ (two names for the same value).

`S_w` Single number: the normalized noise entropy $H(\hat{w})/H(w_0)$.

`objective`, `value` Single number: the minimised dual objective (two names for the same value).

`signal_prior`, `noise_prior` The resolved priors `p0` (n-by-m) and `w0` (n-by-H).

`support` Numeric vector of length H: the resolved noise support `v`.

`converged` Logical: `TRUE` when `optim` reported convergence.

`convergence` Integer: the raw `optim` convergence code.

`method` Character string naming the solver, `"dual"`.

`call` The matched call.

Methods (by generic)

- `coef(matrix_gce)`: Estimated signal matrix `p`.
- `fitted(matrix_gce)`: Fitted aggregates Px.
- `residuals(matrix_gce)`: Residuals $y - Px$ (the estimated noise `e`).
- `print(matrix_gce)`: Compact display.
- `summary(matrix_gce)`: Summary with the Section-7.5 information measures and entropy-ratio test for the signal matrix.

Functions

- `print(summary.matrix_gce)`: Print method for the summary object.

References

Golan, A. (2008). *Information and Entropy Econometrics - A Review and Synthesis*. Foundations and Trends in Econometrics, 2(1-2), 1-145. Section 7.4.

See Also

`matrix_ce` (the exact-moment version this reduces to as ν shrinks to 0).

Examples

```
set.seed(1)
P <- matrix(runif(6), ncol = 2); P <- sweep(P, 2, colSums(P), "/")
x <- c(0.6, 0.3); y <- as.vector(P %*% x)
fit <- matrix_gce(y, x, nu = 0.5)
coef(fit)        # estimated signal matrix p
residuals(fit)   # y - Px  (= estimated noise e)
summary(fit)
```

---

mixed_gce *Doubly-Reparameterized Generalized Cross-Entropy Mixed Model*

Description

Fits the Generalized Cross-Entropy "mixed" model in which the signal probabilities are themselves reparameterized on a bounded support. From a share response `Y` (N x J) and a design array `X` (N x J x K) it recovers the coefficients λ (K x J), the per-observation adding-up multipliers ρ (N), the signal weights θ (N x J x M) with $p_{ij} = \sum_m s_m \theta_{ijm}$, and the noise weights w (N x J x H) with $e_{ij} = \sum_h u_h w_{ijh}$, via the concentrated dual.

Usage

```
mixed_gce(
  Y,
  X,
  theta0 = NULL,
  w0 = NULL,
  s = NULL,
  u = NULL,
  nu = 0.5,
  control = list()
)
```

Arguments

`Y`	N x J numeric response matrix of shares (rows are compositional, summing to about 1 up to the noise term).
`X`	N x J x K design array (`X[i, j, ]` is the covariate vector for observation i and category j).
`theta0`	Optional N x J x M signal prior (each (i, j) fiber > 0 and summing to 1); default uniform `1/M`. `M` is taken from `s`.
`w0`	Optional N x J x H noise prior (each (i, j) fiber > 0 and summing to 1); default uniform `1/H`. `H` is taken from `u`.
`s`	Signal support in $[0, 1]$: `NULL` (default 3-point grid `c(0, 0.5, 1)`), a single whole number giving the number of equally-spaced points, or an explicit numeric vector in $[0, 1]$.
`u`	Noise support in $[-1, 1]$: `NULL` (default 3-point grid `c(-1, 0, 1)`), a single whole number giving the number of equally-spaced points, or an explicit symmetric numeric vector in $[-1, 1]$. Supply your own `u` to widen or tighten the noise term independently of `s` and the priors.
`nu`	Signal/noise entropy weight in $(0, 1)$; default 0.5.
`control`	Named list merged over defaults and passed to `optim` (BFGS): `maxit` (default 1000) and `reltol` (default 1e-12). `fnscale` is forced to -1.

Details

With $V_{ij} = \sum_k X_{ijk}\lambda_{kj}$, $\theta_{ijm} \propto \theta^0_{ijm} \exp[s_m(V_{ij} + \rho_i)/\nu]$ and $w_{ijh} \propto w^0_{ijh} \exp[u_h V_{ij}/(1-\nu)]$. The dual is **maximized** (`fnscale = -1`):

$$LL = \sum_{ij} Y_{ij}V_{ij} + \sum_i \rho_i - \nu \sum_{ij} \log \Omega_{ij} - (1-\nu) \sum_{ij} \log \Psi_{ij},$$

whose first-order conditions are the adding-up constraint $1 - \sum_j p_{ij} = 0$ (in ρ_i) and the data moment $\sum_i X_{ijk}(Y_{ij} - p_{ij} - e_{ij}) = 0$ (in λ_{kj}). Uniform priors give the GME (maximum-entropy) solution; user priors give GCE. As the noise support $u \to 0$ the fit approaches the pure-signal solution.

Value

An object of class `c("mixed_gce", "infometrics")`, which is a list containing the following components:

`lambda`, `lambda_hat` K-by-J matrix of estimated coefficients (two names for the same object). Extracted by `coef`.

`rho`, `rho_hat` Numeric vector of length N: the per-observation adding-up multipliers enforcing $\sum_j p_{ij} = 1$ (two names for the same object).

`theta`, `theta_hat` N-by-J-by-M array of doubly-reparameterized signal weights, each (i, j) fiber summing to 1, with $p_{ij} = \sum_m s_m \theta_{ijm}$ (two names for the same object).

`p`, `p_hat`, `fitted.values` N-by-J matrix of estimated signal probabilities, rows summing to 1 (three names for the same object). Extracted by `fitted`.

`w`, `w_hat` N-by-J-by-H array of estimated noise probabilities, each (i, j) fiber summing to 1 (two names for the same object).

`e`, `e_hat` N-by-J matrix of estimated noise, $e_{ij} = \sum_h u_h \hat{w}_{ijh}$ (two names for the same object). The residuals returned by `residuals` are $Y - \hat{p} - e$.

`V` N-by-J matrix of linear indices $V_{ij} = \sum_k X_{ijk} \lambda_{kj}$.

`vcov` Covariance matrix of the full parameter vector from the dual Hessian, of dimension $(N + KJ) \times (N + KJ)$ with the ρ block first, then λ. `NA` throughout if the Hessian was singular. Extracted by `vcov`.

`se_lambda`, `se_rho` Standard errors of $\hat{\lambda}$ (K-by-J) and $\hat{\rho}$ (length N), taken from `vcov`. Being the naive Hessian inverse these are conservative; for accurate marginal-effect standard errors use `margins` with `se = TRUE`.

`hessian` Dual Hessian at the optimum, of the same dimension as `vcov`.

`X`, `y_mat` The resolved design array (N-by-J-by-K) and response matrix (N-by-J).

`s`, `u` The resolved signal support (length M, in $[0, 1]$) and noise support (length H, symmetric in $[-1, 1]$).

`theta0`, `w0`, `nu` The resolved signal prior (N-by-J-by-M), noise prior (N-by-J-by-H) and entropy weight.

`H_signal` Numeric vector of length N: the per-observation signal entropies of $\hat{\theta}$, summed over the J categories and M support points.

`S`, `S_p` Single number: the normalized signal entropy $H(\hat{\theta})/(NJ \log M)$, in [0, 1] (two names for the same value). It is measured against the uniform distribution, not against `theta0`.

`S_w` Single number: the normalized noise entropy $H(\hat{w})/(NJ \log H)$, in [0, 1].

`objective`, `value` Single number: the maximized dual objective (two names for the same value).

`convergence` Integer: the raw `optim` convergence code.

`converged` Logical: `TRUE` when `optim` reported convergence.

`method` Character string naming the solver, `"dual"`.

`N`, `J`, `K`, `M`, `H` Integers: the numbers of observations, categories, covariates, signal support points and noise support points.

`call` The matched call.

References

Golan, A., Judge, G. and Perloff, J.M. (1996). A maximum entropy approach to recovering information from multinomial response data. *Journal of the American Statistical Association*, **91**(434), 841-853.

See Also

`multinomial_gce` (the reference-normalized sibling), `margins` for marginal effects.

Examples

```
set.seed(1)
N <- 25L; J <- 3L; K <- 2L
X <- array(0, dim = c(N, J, K))
X[, , 1] <- 1                                    # intercept-like column
X[, , 2] <- matrix(rnorm(N * J), N, J)           # a covariate
Y <- matrix(runif(N * J), N, J); Y <- Y / rowSums(Y)   # compositional shares
fit <- mixed_gce(Y, X, nu = 0.5)
coef(fit)                                         # lambda (K x J)
margins(fit)                                      # K x J average marginal effects
margins(fit, se = TRUE)                           # ... with robust sandwich SEs
head(fano_bounds(fit))                            # Fano error bounds (Golan sec 7.5)
```

`multinomial_gce` *nu-Weighted Generalized Cross-Entropy Multinomial Estimator (matrix interface)*

Description

Fits an unordered multinomial response model by the Generalized Cross-Entropy estimator of Golan, Judge & Perloff (1996), extended with a signal/noise entropy weight `nu`. Recovers an N x J signal probability matrix and the N x J x M noise weights from a one-hot / share response matrix `y` and a design matrix `X`, via the concentrated dual over the $K(J-1)$ free Lagrange multipliers (one alternative normalized to zero).

Usage

```
multinomial_gce(
  y,
  X,
  which_alternative = 1L,
  p0 = NULL,
  w0 = NULL,
  v = NULL,
  nu = 0.5,
  control = list()
)
```

Arguments

`y`	N x J numeric response matrix: rows are one-hot indicators or compositional shares summing to 1.
`X`	N x K design matrix.
`which_alternative`	Integer in `1:J`: the alternative normalized to $\lambda = 0$ (default 1).

p0	Optional N x J signal prior (rows sum to 1, strictly positive); default uniform 1/J.
w0	Optional N x J x M noise prior ((i, j) fibers sum to 1, strictly positive); default uniform 1/M.
v	Error support: NULL (default, a 3-point data-scaled grid on $[-1/\sqrt{N}, 1/\sqrt{N}]$), a single whole number giving the support count, or an explicit numeric vector in $[-1, 1]$.
nu	Signal/noise entropy weight in $(0, 1)$; default 0.5.
control	Named list merged over defaults and passed to optim: maxit (default 500), reltol (default 1e-10). fnscale is forced to -1 (the dual is maximized).

Details

$p_{ij} \propto p_{0,ij} \exp(x_i' \lambda_j / \nu)$ and $w_{ijm} \propto w_{0,ijm} \exp(x_i' \lambda_j v_m / (1 - \nu))$, with $e_{ij} = \sum_m v_m w_{ijm}$. The dual is **maximized** (fnscale = -1): $\sum Y\eta - \nu \sum_i \log \Omega_i - (1 - \nu) \sum_{ij} \log \Psi_{ij}$, $\eta = X\Lambda$, and the free gradient is $X'(Y - P - E)$ dropping the normalized column.

At $\nu = 0.5$, which_alternative = 1, and a matched support this recovers the same Golan-Judge-Perloff (1996) multinomial GME estimate. The new pieces are nu, the matrix (y, X) interface, and which_alternative. coef() returns the multipliers lambda (not a transformed beta).

Because the GCE noise forces the reference alternative's error to zero, the arbitrary which_alternative choice slightly shifts the fitted probabilities (a model property, not a numerical artefact).

Value

An object of class c("multinomial_gce", "infometrics"), which is a list containing the following components:

lambda, lambda_hat K-by-J matrix of estimated Lagrange multipliers (two names for the same object). The reference alternative's column is 0. Extracted by coef.

p, p_hat, fitted.values N-by-J matrix of estimated signal probabilities, rows summing to 1 (three names for the same object). Extracted by fitted.

w, w_hat N-by-J-by-M array of estimated noise probabilities, each (i, j) fiber summing to 1 (two names for the same object).

ee, e N-by-J matrix of estimated noise, $e_{ij} = \sum_m v_m \hat{w}_{ijm}$ (two names for the same object). The residuals returned by residuals are $y - \hat{p} - e$.

p0, w0, v, nu The resolved signal prior (N-by-J), noise prior (N-by-J-by-M), noise support (length M) and entropy weight.

objective, value Single number: the maximized dual objective (two names for the same value).

convergence Integer: the raw optim convergence code.

converged Logical: TRUE when optim reported convergence.

method Character string naming the solver, "dual".

H_signal Numeric vector of length N: the per-observation signal entropies $H(\hat{p}_i)$.

S Single number: S_p + S_w, so it lies in [0, 2]. Note that this differs from the other estimators in the package, where S is the normalized signal entropy alone.

`S_p`, `S_w` Single numbers: the normalized signal and noise entropies, $H(\hat{p})/(N \log J)$ and $H(\hat{w})/(NJ \log M)$, each in [0, 1]. These are measured against the uniform distribution, not against `p0` and `w0`.

`y_mat`, `X` The resolved response matrix (N-by-J) and design matrix (N-by-K).

`se` K-by-J matrix of standard errors of $\hat{\lambda}$. The reference alternative's column is 0 (that alternative is normalized, not estimated). These come from the naive Hessian inverse and are conservative; for accurate marginal-effect standard errors use `margins` with `se = TRUE`.

`vcov` Covariance matrix of the *free* multipliers, of dimension $K(J-1) \times K(J-1)$ – the reference alternative is excluded. `NA` throughout if the Hessian was singular.

`hessian` Dual Hessian at the optimum, over the same free multipliers as `vcov`.

`which_alternative` Integer: the index of the reference alternative.

`n_misses` Integer: the number of observations whose modal fitted alternative differs from the modal observed one.

`N`, `J`, `K`, `M` Integers: the numbers of observations, alternatives, covariates and noise support points.

`call` The matched call.

References

Golan, A., Judge, G. and Perloff, J.M. (1996). A maximum entropy approach to recovering information from multinomial response data. *Journal of the American Statistical Association*, **91**(434), 841-853.

See Also

`margins` for marginal effects, `fano_bounds` for information-theoretic error bounds.

Examples

```
set.seed(123)
n <- 200L; J <- 3L; x <- rnorm(n); X <- cbind(1, x)
bt <- cbind(c(0, 0), c(1, 2), c(-0.5, -1))
P  <- exp(X %*% bt); P <- P / rowSums(P)
yc <- apply(P, 1, function(p) sample(J, 1, prob = p))
Y  <- matrix(0, n, J); Y[cbind(seq_len(n), yc)] <- 1
fit <- multinomial_gce(Y, X, which_alternative = 1L, nu = 0.5)
coef(fit)        # lambda (K x J, reference column 0)
margins(fit)              # K x J average marginal effects
margins(fit, se = TRUE)   # ... with delta-method standard errors
head(fano_bounds(fit))    # Fano error bounds for p_hat (Golan sec 7.5)
```

normalize_data *Normalize a numeric vector to [0, 1]*

Description

Rescales a numeric vector so that all values lie in [0, 1] by dividing by the maximum absolute value. Useful for pre-scaling moment matrices before ME/CE estimation to avoid numerical overflow in the exponential terms of the partition function.

Usage

```
normalize_data(x, by = c("max", "range"))
```

Arguments

x	Numeric vector or matrix.
by	Character. Either "max" (divide by max(abs(x)), default) or "range" (min-max scaling to [0, 1]).

Details

Golan (2008, Section 7.1) notes that normalization is often necessary when data contain exponential terms (as in the partition function Omega), since the concentrated (dual) ME/CE model involves expressions of the form exp(lambda * x). A common normalization is dividing each element by max{x_j, y_i}.

Value

A numeric vector or matrix of the same dimensions as x, scaled to [0, 1].

Examples

```
x <- c(10, 30, 50, 20, 40)
normalize_data(x)
normalize_data(x, by = "range")
```

panel_gce *Dual GCE Estimator for the One-Way Error-Components Panel Model*

Description

Estimates the panel regression $y_{nt} = x'_{nt}\beta + \mu_n + \varepsilon_{nt}$ by dual Generalized Cross-Entropy (Lee & Cheon 2014, eq. 3.12), extended with priors p0, g0, w0 and a signal/noise weight nu. The coefficients β, the individual effects μ_n, and the errors ε_{nt} are each reparameterized on bounded supports and recovered jointly via the concentrated dual over the NT Lagrange multipliers.

Usage

```
panel_gce(
  y,
  X,
  Z,
  tt,
  FF,
  p0 = NULL,
  g0 = NULL,
  v = NULL,
  w0 = NULL,
  nu = 0.5,
  layout = c("period", "unit"),
  vcov_type = c("within", "cluster", "disturbance"),
  control = list()
)
```

Arguments

y	Numeric response vector of length N*tt.
X	(N*tt)-by-K design matrix.
Z	Coefficient (signal) support: a K-by-M matrix, row k the support for β_k.
tt	Integer number of time periods (balanced panel; nrow(X) must be a multiple of tt). The number of units is N = nrow(X)/tt.
FF	Individual-effects support: an N-by-R matrix, row n the support for μ_n (typically the same symmetric grid for every unit).
p0	Optional K-by-M signal prior (rows > 0, summing to 1); default uniform.
g0	Optional N-by-R effects prior (rows > 0, summing to 1); default uniform.
v	Optional error support: NULL (default 3-point symmetric grid on ± 3 within-residual sd), a single whole number giving the number of support points, or an explicit numeric vector.
w0	Optional (N*tt)-by-J error prior (rows > 0, summing to 1); default uniform.
nu	Signal/noise entropy weight in $(0, 1)$; default 0.5.
layout	How lambda/rows map to (n, t): "period" (all units for t=1, then t=2, ...) or "unit" (all periods for unit 1, then unit 2, ...). Must match the ordering of y/X.
vcov_type	Covariance estimator for $\hat{\beta}$ (Lee & Cheon 2014, Sec. 3.3); one of:
	"within" (default) the within/fixed-effects form $\hat{\sigma}_e^2(\tilde{X}'\tilde{X})^{-1}$ with $\tilde{X}$ the within-demeaned design and $\hat{\sigma}_e^2 = \sum \hat{e}^2/(NT - N - K)$. The most accurate SE when μ is well identified ($\hat{\beta}$ is within-identified); matched simulation sampling SDs to within ~1%.
	"cluster" the within estimator with a unit-clustered meat $(\tilde{X}'\tilde{X})^{-1}[\sum_n \tilde{X}_n'\hat{e}_n\hat{e}_n'\tilde{X}_n](\tilde{X}'\tilde{X})^{-1}$; robust to heteroskedasticity and serial correlation in ε.

`"disturbance"` the paper's stated asymptotic variance $Q^{-1}\Xi Q^{-1}$ (eq. 3.17), i.e. $(X'X)^{-1}[\sum_n X_n'\hat{u}_n\hat{u}_n'X_n](X'X)^{-1}$ using the composite disturbance $\hat{u} = y - X\hat{\beta}$ (unit-clustered). Conservative relative to `"within"` because it counts the between-unit variation that $\hat{\mu}$ absorbs.

The p and μ SEs are delta-method transforms of the chosen $Var(\hat{\beta})$ (see **Details**).

`control` Named list merged over defaults and passed to `optim` (BFGS): `maxit` (default 1000) and `reltol` (default 1e-12). `fnscale` is forced to -1.

Details

With $\beta_k = \sum_m z_{km}p_{km}$ (support `Z`, prior `p0`), $\mu_n = \sum_r f_{nr}g_{nr}$ (support `FF`, prior `g0`), and $e_{nt} = \sum_j v_j w_{ntj}$ (support `v`, prior `w0`), the dual is maximized (`fnscale = -1`); β and μ are weighted by `nu`, the errors by `1 - nu`. The gradient is the model equation $y - X\beta - \mu - e$, so at the optimum it holds to numerical zero (`foc_residual`). As the supports widen the estimate approaches the within (fixed-effects) estimate.

Standard errors follow Lee & Cheon (2014, Sec. 3.3). Because μ is estimated, $\hat{\beta}$ is identified from within-unit variation, so its sampling variance is the within (fixed-effects) form $\hat{\sigma}_e^2(\tilde{X}'\tilde{X})^{-1}$ (option `vcov_type = "within"`, the default and the most accurate in Monte Carlo). The paper's stated asymptotic variance $Q^{-1}\Xi Q^{-1}$ (eq. 3.17) with the composite disturbance is available as `vcov_type = "disturbance"` (it is conservative here). SEs for p and μ are obtained by the delta method from $Var(\hat{\beta})$: $SE(\hat{p}_{km}) = |p_{km}(z_{km}-\beta_k)|/Var_p(z_k)\cdot SE(\hat{\beta}_k)$ and $SE(\hat{\mu}_n) = \sqrt{\hat{\sigma}_e^2/T_n + \bar{x}_n' Var(\hat{\beta})\,\bar{x}_n}$ (the latter a finite-T prediction SE, since individual effects are not $\sqrt{N}$-consistent). `X` must not include an intercept column (the individual effects subsume the level; a constant column makes the within design singular).

Value

An object of class `c("panel_gce", "infometrics")` with `coefficients`/`b_hat` (β, K; `coef()` returns it), `mu_hat` (N individual effects), `e_hat` (N*tt errors), `p_hat`/`g_hat`/`w_hat` (reparameterization weights), `lambda_hat`, `fitted.values`, `residuals`, `foc_residual`, `H_p`/`H_g`/`H_w`, `H_signal`, `S_p`/`S`, `objective`/`value`, `converged`/ `convergence`, `method`, `nu`, `v`, `layout`, the stored inputs, and `call`. Standard errors: `vcov` (K-by-K, accessible via `vcov`), `se_beta` (K), `se_p` (K-by-M), `se_mu` (N), `sigma2_eps`, and `vcov_type`.

References

Lee, S. and Cheon, S. (2014). Dual generalized maximum entropy estimation for panel data regression models. *Communications for Statistical Applications and Methods*, **21**(5), 395-409.

See Also

`linreg` for the (non-panel) GME/GCE regression.

Examples

```
set.seed(1)
N <- 20L; T <- 4L; K <- 2L
```

```
mu   <- rnorm(N)                          # individual effects
id   <- rep(seq_len(N), each = T)         # unit-major ordering
X    <- matrix(rnorm(N * T * K), N * T, K)
y    <- as.vector(X %*% c(1.5, -0.8)) + mu[id] + rnorm(N * T, 0, 0.5)
Z  <- matrix(c(-10, 0, 10), nrow = K, ncol = 3, byrow = TRUE)  # beta support
FF <- matrix(c(-6, 0, 6),  nrow = N, ncol = 3, byrow = TRUE)   # mu support
fit <- panel_gce(y, X, Z, tt = T, FF = FF, layout = "unit")
coef(fit)                                 # ~ (1.5, -0.8) (within/FE estimate)
```

`shannon_entropy` *Shannon Entropy*

Description

Computes the Shannon entropy H(p) = -sum(p * log(p)), using the natural logarithm (nats) by default. The convention 0 * log(0) = 0 is applied.

Usage

```
shannon_entropy(p, base = exp(1))
```

Arguments

`p`	Numeric vector of probabilities. Must be non-negative and sum to 1.
`base`	Positive numeric. Logarithm base. Use `base = 2` for bits, `base = exp(1)` (default) for nats, `base = 10` for hartleys.

Details

Shannon's entropy measure satisfies three axioms: normalization (maximum entropy for uniform distributions), continuity, and additivity for independent sub-systems. It equals zero if and only if all probability mass is concentrated on a single outcome (perfect certainty) and reaches its maximum value of `log(K)` for the uniform distribution over K outcomes. See Golan (2008, Section 3.1) for a detailed treatment.

Value

A single non-negative numeric value. Returns 0 for a degenerate (point-mass) distribution and `log(length(p))` for the uniform distribution.

References

Golan, A. (2008). Information and Entropy Econometrics. *Foundations and Trends in Econometrics*, **2**(1-2), 1-145.

Shannon, C.E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, **27**, 379-423.

Examples

```
# Uniform distribution over 6 outcomes (maximum entropy)
shannon_entropy(rep(1/6, 6))

# Non-uniform distribution (lower entropy)
shannon_entropy(c(0.5, 0.3, 0.2))

# In bits (base 2)
shannon_entropy(rep(1/2, 2), base = 2)  # = 1 bit
```

Index