Package: inequality (via r-universe)

May 30, 2026

Title Inequality Measurement, Decomposition, and Poverty Analysis

Version 0.2.0

Description Tools for measuring income and wealth inequality. Computes the Gini coefficient with bootstrap or asymptotic confidence intervals following Davidson (2009) <doi:10.1016/j.jeconom.2008.11.004>, the extended S-Gini family, Theil T and L indices (generalised entropy family), the Atkinson index, the Kolm absolute inequality index, Palma ratio, Hoover index, percentile ratios, and Lorenz curves. Supports between-within group decomposition following Bourguignon (1979) <doi:10.2307/1914138>, income share tabulation, concentration indices for health inequality with Erreygers (2009) correction, Kakwani tax progressivity and Reynolds-Smolensky redistribution indices, Foster-Greer-Thorbecke poverty measures including the Sen index, growth incidence curves following Ravallion and Chen (2003) <doi:10.1016/S0165-1765(02)00205-7>, and Wolfson polarisation. All functions accept optional survey weights and work with data from any source.

Depends R (>= 4.1.0)

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports cli (>= 3.6.0), grDevices, graphics, stats

Suggests ineq, testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://charlescoverdale.github.io/inequality/, https://github.com/charlescoverdale/inequality

BugReports https://github.com/charlescoverdale/inequality/issues

Repository https://charlescoverdale.r-universe.dev

Date/Publication 2026-05-30 14:58:57 UTC

RemoteUrl https://github.com/charlescoverdale/inequality

RemoteRef HEAD

RemoteSha ab659804766aac14c496f76d35f5bcadc6991892

Contents

`iq_atkinson` *Atkinson index*

Description

Computes the Atkinson inequality index, which incorporates an explicit normative judgement about inequality aversion through the parameter epsilon. Higher epsilon gives more weight to transfers at the bottom of the distribution.

Usage

```
iq_atkinson(
  x,
  weights = NULL,
  epsilon = 0.5,
  na.rm = FALSE,
  ci = FALSE,
```

```
  R = 1000L,
  level = 0.95
)
```

Arguments

x	Numeric vector of incomes (strictly positive).
weights	Optional numeric vector of survey weights.
epsilon	Numeric. Inequality aversion parameter (> 0). Default 0.5. Common values: 0.5 (moderate), 1.0 (high), 2.0 (very high aversion).
na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute bootstrap confidence intervals? Default FALSE.
R	Integer. Number of bootstrap replicates. Default 1000.
level	Numeric. Confidence level. Default 0.95.

Details

The Atkinson index involves either a power transformation x^(1 - epsilon) or log(x) (when epsilon = 1) and so requires strictly positive values. Use the Gini, S-Gini, or Kolm index for distributions that include zeros or negatives.

Value

An S3 object of class "iq_atkinson" with elements:

value Numeric. The Atkinson index (0 to 1).

epsilon Numeric. The inequality aversion parameter used.

ede Numeric. The equally distributed equivalent income.

mean_income Numeric. The mean income.

n Integer. Number of observations.

se, ci_lower, ci_upper, level Bootstrap CI fields, NULL unless ci = TRUE.

References

Atkinson, A. B. (1970). "On the Measurement of Inequality." *Journal of Economic Theory*, 2(3), 244–263.

Biewen, M. and Jenkins, S. P. (2006). "Variance Estimation for Generalized Entropy and Atkinson Inequality Indices: The Complex Survey Data Case." *Oxford Bulletin of Economics and Statistics*, 68(3), 371–383.

Examples

```
d <- iq_sample_data("income")

# Moderate inequality aversion
iq_atkinson(d$income, epsilon = 0.5)
```

```
# With bootstrap CIs
iq_atkinson(d$income, epsilon = 0.5, ci = TRUE, R = 200)

# High inequality aversion
iq_atkinson(d$income, epsilon = 1)

# Very high inequality aversion
iq_atkinson(d$income, epsilon = 2)
```

`iq_compare` *Compare inequality measures*

Description

Computes all major inequality indices on the same data and returns a summary table for easy comparison.

Usage

```
iq_compare(
  x,
  weights = NULL,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95,
  negatives = c("error", "keep")
)
```

Arguments

`x`	Numeric vector of incomes (strictly positive by default; see `negatives`).
`weights`	Optional numeric vector of survey weights.
`na.rm`	Logical. Remove `NA` values? Default `FALSE`.
`ci`	Logical. Compute bootstrap CIs for every measure in the table? Default `FALSE`.
`R`	Integer. Number of bootstrap replicates. Default `1000`.
`level`	Numeric. Confidence level. Default `0.95`.
`negatives`	Character. `"error"` (default) requires strictly positive `x`; `"keep"` permits zero or negative values, with `NA` returned for measures that are undefined on those values.

Details

When `ci = TRUE` the function runs a single bootstrap loop, recomputing every measure on each resample. This is far cheaper than calling each measure with its own `ci = TRUE` and produces a CI for every row of the table.

By default `iq_compare()` requires strictly positive values because the Theil and Atkinson rows are mathematically undefined at zero or below. Pass `negatives = "keep"` to permit zero or negative values: the Theil and Atkinson rows are returned as `NA` in that case, while the Gini, S-Gini, Kolm, Wolfson, Palma, Hoover and percentile-ratio rows are computed using the formulas appropriate for that input.

Value

An S3 object of class `"iq_comparison"` with elements:

table data.frame with columns `measure`, `value`, and (when `ci = TRUE`) `ci_lower` and `ci_upper`.

n Integer. Number of observations.

level Numeric or `NULL`. Confidence level.

Examples

```
d <- iq_sample_data("income")
iq_compare(d$income)

# CIs for every measure in the table (one bootstrap loop, all rows)
iq_compare(d$income, ci = TRUE, R = 200)

# Wealth distributions can include negatives
wealth <- c(-5000, 0, 5000, 20000, 80000, 250000, 1e6)
iq_compare(wealth, negatives = "keep")
```

---

iq_concentration *Concentration index*

Description

Computes the concentration index, which measures inequality in a health (or other) variable across the income distribution. Unlike the Gini coefficient, the ranking variable and the outcome variable are different.

Usage

```
iq_concentration(
  x,
  rank,
  weights = NULL,
  correction = c("none", "erreygers", "wagstaff"),
  bounds = c(0, 1),
```

```
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95
)
```

Arguments

x	Numeric vector of outcome values (e.g. health expenditure).
rank	Numeric vector of ranking values (e.g. income). Must be the same length as x.
weights	Optional numeric vector of survey weights.
correction	Character. "none" (default) for the standard index; "erreygers" for the Erreygers (2009) bounded-variable correction; "wagstaff" for the Wagstaff (2005) normalised index.
bounds	Numeric vector of length 2 giving the lower and upper bounds of x. Required when correction = "erreygers". Default c(0, 1) (suitable for binary or proportion variables).
na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute bootstrap confidence intervals? Default FALSE.
R	Integer. Number of bootstrap replicates. Default 1000.
level	Numeric. Confidence level. Default 0.95.

Details

A positive value indicates the outcome is concentrated among the better-off; a negative value indicates concentration among the worse-off.

For bounded variables (e.g. binary health indicators), the standard concentration index has bounds that depend on the mean. Two corrections are available:

- correction = "erreygers": the Erreygers (2009) corrected index, E = 4 * mu / (b - a) * C, which has fixed bounds of -1 to 1.
- correction = "wagstaff": the Wagstaff (2005) normalised index, W = C / (1 - mu / b) for variables bounded above at b, which is the standard normalisation in much of the health-economics literature.

Value

An S3 object of class "iq_concentration" with elements:

value Numeric. The concentration index.

correction Character. The correction applied.

n Integer. Number of observations.

se, ci_lower, ci_upper, level Bootstrap CI fields, NULL unless ci = TRUE.

References

Wagstaff, A., Paci, P. and van Doorslaer, E. (1991). "On the Measurement of Inequalities in Health." *Social Science and Medicine*, 33(5), 545–557.

Erreygers, G. (2009). "Correcting the Concentration Index." *Journal of Health Economics*, 28(2), 504–515.

Wagstaff, A. (2005). "The Bounds of the Concentration Index when the Variable of Interest is Binary, with an Application to Immunization Inequality." *Health Economics*, 14(4), 429–432.

Examples

```
set.seed(1)
income <- rlnorm(200, 10, 0.8)
health_exp <- income * 0.05 + rnorm(200, 500, 100)
iq_concentration(health_exp, rank = income)

# With bootstrap CIs
iq_concentration(health_exp, rank = income, ci = TRUE, R = 200)

# Binary outcome with Erreygers correction
sick <- as.numeric(income < median(income)) + rbinom(200, 1, 0.1)
sick <- pmin(sick, 1)
iq_concentration(sick, rank = income, correction = "erreygers")
```

---

iq_decompose *Between-within group decomposition*

Description

Decomposes a generalised entropy index into a between-group component (inequality due to differences in group means) and a within-group component (inequality within each group). The decomposition is exact: between + within = total.

Usage

```
iq_decompose(x, group, weights = NULL, index = "T", na.rm = FALSE)
```

Arguments

x	Numeric vector of incomes (strictly positive).
group	Factor or character vector identifying group membership.
weights	Optional numeric vector of survey weights.
index	Character or numeric. "T" for Theil T (GE(1)), "L" for mean log deviation (GE(0)), or a numeric alpha. Default "T".
na.rm	Logical. Remove NA values? Default FALSE.

Value

An S3 object of class `"iq_decomposition"` with elements:

total Numeric. The total GE index.

between Numeric. The between-group component.

within Numeric. The within-group component.

groups data.frame with columns `group`, `n`, `mean_income`, `pop_share`, `income_share`, `within_ge`.

index_name Character. Name of the index used.

References

Bourguignon, F. (1979). "Decomposable Income Inequality Measures." *Econometrica*, 47(4), 901–920.

Examples

```
d <- iq_sample_data("grouped")
iq_decompose(d$income, d$group)
```

---

iq_gini *Gini coefficient*

Description

Computes the Gini coefficient of a distribution, with optional survey weights and confidence intervals (bootstrap or asymptotic).

Usage

```
iq_gini(
  x,
  weights = NULL,
  na.rm = FALSE,
  ci = FALSE,
  method = c("bootstrap", "asymptotic"),
  R = 1000L,
  level = 0.95,
  negatives = c("error", "keep"),
  normalised = FALSE
)
```

Arguments

x	Numeric vector of incomes or values.
weights	Optional numeric vector of survey weights.
na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute confidence intervals? Default FALSE.
method	Character. CI method: "bootstrap" (default) or "asymptotic" (jackknife-based, faster for large samples).
R	Integer. Number of bootstrap replicates (ignored for asymptotic). Default 1000.
level	Numeric. Confidence level. Default 0.95.
negatives	Character. "error" (default) aborts when x contains negative values; "keep" permits negatives.
normalised	Logical. Use the Raffinetti, Siletti and Vernizzi (2017) normalised Gini? Default FALSE. Only meaningful when negatives = "keep".

Details

For a strictly non-negative distribution the Gini ranges from 0 (perfect equality) to 1 (perfect inequality) and equals twice the area between the Lorenz curve and the 45-degree line.

Following feedback from Cowell and Flachaire (personal communication, 2026) the package permits negative values via negatives = "keep". Two policies are then available:

- normalised = FALSE (default): the standard formula is applied. With negatives present the index is no longer bounded in the unit interval. When the population mean is non-positive the Gini has no inequality interpretation and the function returns NA with a warning.
- normalised = TRUE: the Raffinetti, Siletti and Vernizzi (2017) normalised Gini, which rescales the index back into the unit interval for distributions containing negatives. The denominator is replaced by mean(|x|) so the index is well-defined whenever any observation is non-zero.

Value

An S3 object of class "iq_gini" with elements:

gini Numeric. The Gini coefficient (or NA when undefined).

n Integer. Number of observations.

se Numeric or NULL. Standard error.

ci_lower Numeric or NULL. Lower bound of the CI.

ci_upper Numeric or NULL. Upper bound of the CI.

level Numeric or NULL. Confidence level.

method Character or NULL. CI method used.

has_negatives Logical. Whether the input contained negatives.

normalised Logical. Whether the Raffinetti et al. normalisation was applied.

References

Gini, C. (1912). "Variabilita e mutabilita." Reprinted in *Memorie di metodologica statistica* (Ed. Pizetti E, Salvemini, T). Rome: Libreria Eredi Virgilio Veschi.

Davidson, R. (2009). "Reliable Inference for the Gini Index." *Journal of Econometrics*, 150(1), 30–40.

Raffinetti, E., Siletti, E. and Vernizzi, A. (2017). "Analyzing the Effects of Negative and Non-negative Values on Income Inequality: Evidence from the Survey of Household Income and Wealth of the Bank of Italy (2012)." *Social Indicators Research*, 133(1), 185–207.

Examples

```
d <- iq_sample_data("income")
iq_gini(d$income)

# Bootstrap CIs
iq_gini(d$income, ci = TRUE, R = 500)

# Asymptotic CIs (faster for large samples)
iq_gini(d$income, ci = TRUE, method = "asymptotic")

# Wealth distributions can include negative net worth
wealth <- c(-5000, -1000, 0, 5000, 20000, 80000, 250000)
iq_gini(wealth, negatives = "keep")

# Same data with the Raffinetti et al. (2017) normalisation
iq_gini(wealth, negatives = "keep", normalised = TRUE)

# Perfect equality
iq_gini(rep(100, 50))
```

---

iq_growth_incidence *Growth incidence curve*

Description

Computes the growth incidence curve (GIC), showing the annualised or total growth rate at each quantile of the distribution between two time periods.

Usage

```
iq_growth_incidence(
  x_t0,
  x_t1,
  weights_t0 = NULL,
  weights_t1 = NULL,
  n_quantiles = 20L,
  na.rm = FALSE
)
```

Arguments

x_t0	Numeric vector of incomes in period 0.
x_t1	Numeric vector of incomes in period 1. Must be the same length as x_t0.
weights_t0	Optional weights for period 0.
weights_t1	Optional weights for period 1.
n_quantiles	Integer. Number of quantile bins. Default 20 (ventiles).
na.rm	Logical. Remove NA values? Default FALSE.

Details

If the GIC is upward-sloping, the rich grew faster and inequality increased. If downward-sloping, growth was pro-poor.

Value

An S3 object of class "iq_growth_incidence" with elements:

gic data.frame with columns quantile (midpoint), growth (proportional growth rate at that quantile).

mean_growth Numeric. Mean growth across all quantiles.

median_growth Numeric. Median growth rate.

n_quantiles Integer.

References

Ravallion, M. and Chen, S. (2003). "Measuring Pro-Poor Growth." *Economics Letters*, 78(1), 93–99.

Examples

```
d <- iq_sample_data("panel")
gic <- iq_growth_incidence(d$income_t0, d$income_t1)
plot(gic)
```

---

iq_hoover *Hoover index (Robin Hood index)*

Description

Computes the Hoover index, also known as the Robin Hood index or the Schutz coefficient. It equals the maximum proportion of total income that would need to be redistributed to achieve perfect equality, or equivalently, half the mean absolute deviation divided by the mean.

Usage

```
iq_hoover(
  x,
  weights = NULL,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95,
  negatives = c("error", "keep")
)
```

Arguments

x	Numeric vector of incomes.
weights	Optional numeric vector of survey weights.
na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute bootstrap confidence intervals? Default FALSE.
R	Integer. Number of bootstrap replicates. Default 1000.
level	Numeric. Confidence level. Default 0.95.
negatives	Character. "error" (default) aborts on negatives; "keep" permits them.

Value

An S3 object of class "iq_hoover" with elements:

value Numeric. The Hoover index (0 to 1 with non-negative input).

n Integer. Number of observations.

se, ci_lower, ci_upper, level Bootstrap CI fields, NULL unless ci = TRUE.

Examples

```
d <- iq_sample_data("income")
iq_hoover(d$income)

# With bootstrap CIs
iq_hoover(d$income, ci = TRUE, R = 200)

# Perfect equality
iq_hoover(rep(100, 50))
```

`iq_kakwani` *Kakwani progressivity index*

Description

Measures the progressivity of a tax or transfer system. A positive value indicates progressivity (the rich pay a larger share than their income share); a negative value indicates regressivity. Zero means proportional.

Usage

```
iq_kakwani(
  pre_tax,
  tax,
  weights = NULL,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95,
  negatives = c("error", "keep")
)
```

Arguments

`pre_tax`	Numeric vector of pre-tax incomes (non-negative by default).
`tax`	Numeric vector of tax payments (same length as `pre_tax`). Positive values are taxes paid; negative values are transfers received.
`weights`	Optional numeric vector of survey weights.
`na.rm`	Logical. Remove `NA` values? Default `FALSE`.
`ci`	Logical. Compute bootstrap confidence intervals on the Kakwani and Reynolds-Smolensky indices? Default `FALSE`.
`R`	Integer. Number of bootstrap replicates. Default `1000`.
`level`	Numeric. Confidence level. Default `0.95`.
`negatives`	Character. `"error"` (default) aborts on negative pre-tax incomes; `"keep"` permits them.

Details

The Kakwani index equals the concentration coefficient of the tax minus the pre-tax Gini coefficient: K = C_T - G_pre.

The post-tax Gini is computed on `pre_tax - tax` directly. Households whose post-tax income is negative are kept as-is, so the post-tax Gini may exceed 1 in distributions with extreme tax burdens. Pass `negatives = "error"` to abort on negative pre-tax incomes.

Value

An S3 object of class `"iq_kakwani"` with elements:

kakwani Numeric. The Kakwani index (-1 to 1).

gini_pre Numeric. The pre-tax Gini coefficient.

concentration_tax Numeric. The concentration coefficient of taxes.

reynolds_smolensky Numeric. The Reynolds-Smolensky index (pre-tax Gini minus post-tax Gini).

gini_post Numeric. The post-tax Gini coefficient.

avg_tax_rate Numeric. Average effective tax rate.

n Integer. Number of observations.

kakwani_ci, rs_ci Lists with `lower` and `upper` (or `NULL`).

References

Kakwani, N. C. (1977). "Measurement of Tax Progressivity: An International Comparison." *The Economic Journal*, 87(345), 71–80.

Reynolds, M. and Smolensky, E. (1977). *Public Expenditures, Taxes, and the Distribution of Income*. New York: Academic Press.

Examples

```
set.seed(1)
pre <- iq_sample_data("income")$income
# Progressive tax: higher rate for higher incomes
tax <- pre * (0.10 + 0.15 * (pre / max(pre)))
iq_kakwani(pre, tax)

# With bootstrap CIs
iq_kakwani(pre, tax, ci = TRUE, R = 200)
```

---

iq_kolm *Kolm index (absolute inequality)*

Description

Computes the Kolm index, the only standard inequality measure that is translation-invariant (absolute). Adding the same amount to every income leaves the index unchanged. All other indices in this package are scale-invariant (relative): multiplying every income by the same factor leaves them unchanged.

Usage

```
iq_kolm(
  x,
  weights = NULL,
  alpha = 1,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95
)
```

Arguments

x	Numeric vector of incomes.
weights	Optional numeric vector of survey weights.
alpha	Numeric. Inequality aversion parameter (> 0). Default 1.
na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute bootstrap confidence intervals? Default FALSE.
R	Integer. Number of bootstrap replicates. Default 1000.
level	Numeric. Confidence level. Default 0.95.

Details

Higher alpha gives more weight to inequality at the bottom of the distribution. The index is always non-negative and equals zero only under perfect equality. The Kolm index is well-defined for any real values, including negatives.

Value

An S3 object of class "iq_kolm" with elements:

value Numeric. The Kolm index.

alpha Numeric. The inequality aversion parameter used.

n Integer. Number of observations.

se, ci_lower, ci_upper, level Bootstrap CI fields, NULL unless ci = TRUE.

References

Kolm, S.-C. (1976). "Unequal Inequalities II." *Journal of Economic Theory*, 13(1), 82–111.

Examples

```
d <- iq_sample_data("income")
iq_kolm(d$income, alpha = 1)

# With bootstrap CIs
iq_kolm(d$income, alpha = 1, ci = TRUE, R = 200)
```

```
# Higher aversion to inequality at the bottom
iq_kolm(d$income, alpha = 2)
```

`iq_lorenz` *Lorenz curve*

Description

Computes the Lorenz curve: the cumulative share of income held by the cumulative share of the population, ordered from poorest to richest. The result can be plotted with `plot()`.

Usage

```
iq_lorenz(x, weights = NULL, na.rm = FALSE)
```

Arguments

`x`	Numeric vector of incomes (non-negative).
`weights`	Optional numeric vector of survey weights.
`na.rm`	Logical. Remove `NA` values? Default `FALSE`.

Value

An S3 object of class `"iq_lorenz"` with elements:

curve data.frame with columns `cum_pop` and `cum_income` (both 0 to 1). Starts at (0, 0) and ends at (1, 1).

gini Numeric. The Gini coefficient (twice the area between the curve and the diagonal).

n Integer. Number of observations.

References

Lorenz, M. O. (1905). "Methods of Measuring the Concentration of Wealth." *Publications of the American Statistical Association*, 9(70), 209–219.

Examples

```
d <- iq_sample_data("income")
lc <- iq_lorenz(d$income)
plot(lc)
```

`iq_palma` *Palma ratio*

Description

Computes the Palma ratio: the share of total income received by the top 10 percent divided by the share received by the bottom 40 percent.

Usage

```
iq_palma(
  x,
  weights = NULL,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95,
  negatives = c("error", "keep")
)
```

Arguments

`x`	Numeric vector of incomes.
`weights`	Optional numeric vector of survey weights.
`na.rm`	Logical. Remove `NA` values? Default `FALSE`.
`ci`	Logical. Compute bootstrap confidence intervals? Default `FALSE`.
`R`	Integer. Number of bootstrap replicates. Default `1000`.
`level`	Numeric. Confidence level. Default `0.95`.
`negatives`	Character. `"error"` (default) aborts on negatives; `"keep"` permits them.

Details

The Palma ratio is motivated by Palma's (2011) observation that the "middle" 50 percent (deciles 5–9) tends to capture a remarkably stable share of income across countries, so inequality is driven by what happens at the tails. A Palma ratio of 1 means the top 10 percent and bottom 40 percent receive equal shares.

Distributions containing negative values may produce a non-positive bottom-40 share, in which case the Palma ratio is undefined. The function returns `NA` with a warning rather than aborting.

Value

An S3 object of class `"iq_palma"` with elements:

palma Numeric. The Palma ratio.

top10_share Numeric. Share of income held by the top 10 percent.

bottom40_share Numeric. Share of income held by the bottom 40 percent.

n Integer. Number of observations.

se, ci_lower, ci_upper, level Bootstrap CI fields, `NULL` unless `ci = TRUE`.

References

Palma, J. G. (2011). "Homogeneous Middles vs. Heterogeneous Tails, and the End of the 'Inverted-U': It's All About the Share of the Rich." *Development and Change*, 42(1), 87–153.

Examples

```
d <- iq_sample_data("income")
iq_palma(d$income)

# With bootstrap CIs
iq_palma(d$income, ci = TRUE, R = 200)

# Equal distribution: Palma = 0.25/0.40 = 0.625
iq_palma(rep(100, 100))
```

---

iq_percentile_ratio *Percentile ratio*

Description

Computes the ratio of two percentiles of the distribution. Common choices include P90/P10 (inter-decile ratio), P80/P20, and P50/P10.

Usage

```
iq_percentile_ratio(
  x,
  weights = NULL,
  upper = 90,
  lower = 10,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95
)
```

Arguments

`x`	Numeric vector of incomes.
`weights`	Optional numeric vector of survey weights.
`upper`	Numeric. Upper percentile (0 to 100). Default `90`.
`lower`	Numeric. Lower percentile (0 to 100). Default `10`.

na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute bootstrap confidence intervals? Default FALSE.
R	Integer. Number of bootstrap replicates. Default 1000.
level	Numeric. Confidence level. Default 0.95.

Value

An S3 object of class "iq_percentile_ratio" with elements:

ratio Numeric. The percentile ratio.

upper_value Numeric. The value at the upper percentile.

lower_value Numeric. The value at the lower percentile.

upper Numeric. The upper percentile used.

lower Numeric. The lower percentile used.

n Integer. Number of observations.

se, ci_lower, ci_upper, level Bootstrap CI fields, NULL unless ci = TRUE.

Examples

```
d <- iq_sample_data("income")

# P90/P10 (interdecile ratio)
iq_percentile_ratio(d$income)

# With bootstrap CIs
iq_percentile_ratio(d$income, ci = TRUE, R = 200)

# P80/P20
iq_percentile_ratio(d$income, upper = 80, lower = 20)
```

---

iq_polarisation *Polarisation index*

Description

Computes the Wolfson bipolarisation index, which measures the extent to which a distribution is bimodal (clustering at the tails) rather than unimodal. Higher values indicate more polarisation.

Usage

```
iq_polarisation(
  x,
  weights = NULL,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95,
  negatives = c("error", "keep")
)
```

Arguments

x	Numeric vector of incomes.
weights	Optional numeric vector of survey weights.
na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute bootstrap confidence intervals? Default FALSE.
R	Integer. Number of bootstrap replicates. Default 1000.
level	Numeric. Confidence level. Default 0.95.
negatives	Character. "error" (default) aborts on negatives; "keep" permits them.

Value

An S3 object of class "iq_polarisation" with elements:

wolfson Numeric. The Wolfson polarisation index.

gini Numeric. The Gini coefficient.

median Numeric. The weighted median income.

mean Numeric. The weighted mean income.

n Integer. Number of observations.

se, ci_lower, ci_upper, level Bootstrap CI fields, NULL unless ci = TRUE.

References

Wolfson, M. C. (1994). "When Inequalities Diverge." *American Economic Review*, 84(2), 353–358.

Foster, J. E. and Wolfson, M. C. (2010). "Polarization and the Decline of the Middle Class: Canada and the US." *Journal of Economic Inequality*, 8(2), 247–273.

Examples

```
d <- iq_sample_data("income")
iq_polarisation(d$income)

# With bootstrap CIs
iq_polarisation(d$income, ci = TRUE, R = 200)
```

---

iq_poverty *Poverty measures*

Description

Computes the Foster-Greer-Thorbecke (FGT) family of poverty measures, plus the Sen index and the Watts index. All measures require a poverty line.

Usage

```
iq_poverty(
  x,
  line,
  weights = NULL,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95
)
```

Arguments

x	Numeric vector of incomes (non-negative).
line	Numeric. The poverty line. Required.
weights	Optional numeric vector of survey weights.
na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute bootstrap confidence intervals on the headcount, gap, severity, and Sen indices? Default FALSE.
R	Integer. Number of bootstrap replicates. Default 1000.
level	Numeric. Confidence level. Default 0.95.

Value

An S3 object of class "iq_poverty" with elements:

headcount Numeric. FGT(0): proportion below the poverty line.

gap Numeric. FGT(1): average normalised gap.

severity Numeric. FGT(2): average squared normalised gap.

sen Numeric. Sen index.

watts Numeric. Watts index.

line Numeric. The poverty line used.

n Integer. Number of observations.

n_poor Integer. Number of observations below the line.

ci Optional list of bootstrap CIs for the four standard FGT/Sen measures (each a list with lower and upper).

level Numeric or NULL. Confidence level.

References

Foster, J., Greer, J. and Thorbecke, E. (1984). "A Class of Decomposable Poverty Measures." *Econometrica*, 52(3), 761–766.

Sen, A. (1976). "Poverty: An Ordinal Approach to Measurement." *Econometrica*, 44(2), 219–231.

Examples

```
d <- iq_sample_data("income")
# Poverty line at the 20th percentile
p20 <- quantile(d$income, 0.20)
iq_poverty(d$income, line = p20)

# With bootstrap CIs
iq_poverty(d$income, line = p20, ci = TRUE, R = 200)
```

`iq_sample_data` *Generate sample inequality data*

Description

Creates synthetic data for testing and demonstrating inequalitykit functions. Three types are available: individual incomes, a two-period panel for growth incidence analysis, and grouped incomes for decomposition.

Usage

```
iq_sample_data(type = c("income", "panel", "grouped"))
```

Arguments

`type`	Character. One of `"income"`, `"panel"`, or `"grouped"`.

Value

A data.frame.

`"income"` 1000 rows with columns `income` and `weight`. Drawn from a lognormal distribution (mean log 10.5, sd log 0.8), producing realistic income-like data centred around 40,000.

`"panel"` 1000 rows with columns `income_t0`, `income_t1`, `weight`. Two periods with heterogeneous growth (bottom grows slower than top, mimicking rising inequality).

`"grouped"` 1000 rows with columns `income`, `group`, `weight`. Three groups (A, B, C) with different mean incomes for between/within decomposition.

Examples

```
d <- iq_sample_data("income")
head(d)

panel <- iq_sample_data("panel")
head(panel)

grouped <- iq_sample_data("grouped")
head(grouped)
```

`iq_sgini` *S-Gini (extended Gini family)*

Description

Computes the S-Gini coefficient, a one-parameter generalisation of the Gini that allows the user to specify how much weight to give different parts of the distribution. The standard Gini is the special case `delta = 2`.

Usage

```
iq_sgini(
  x,
  weights = NULL,
  delta = 2,
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95,
  negatives = c("error", "keep")
)
```

Arguments

`x`	Numeric vector of incomes.
`weights`	Optional numeric vector of survey weights.
`delta`	Numeric. Inequality aversion parameter (> 1). Default `2` (standard Gini).
`na.rm`	Logical. Remove `NA` values? Default `FALSE`.
`ci`	Logical. Compute bootstrap confidence intervals? Default `FALSE`.
`R`	Integer. Number of bootstrap replicates. Default `1000`.
`level`	Numeric. Confidence level. Default `0.95`.
`negatives`	Character. `"error"` (default) aborts on negatives; `"keep"` permits them.

Details

Lower delta (approaching 1) gives equal weight everywhere; higher delta gives more weight to the bottom of the distribution. The standard Gini (delta = 2) weights by rank position. Delta = 3 or 4 places even more emphasis on the poorest.

Like the standard Gini, the S-Gini is well-defined for distributions containing negative values via `negatives = "keep"`, though the resulting index is no longer bounded in the unit interval.

Value

An S3 object of class `"iq_sgini"` with elements:

value Numeric. The S-Gini coefficient.

delta Numeric. The inequality aversion parameter used.

n Integer. Number of observations.

se, ci_lower, ci_upper, level Bootstrap CI fields, `NULL` unless `ci = TRUE`.

has_negatives Logical. Whether the input contained negatives.

References

Donaldson, D. and Weymark, J. A. (1980). "A Single-Parameter Generalization of the Gini Indices of Inequality." *Journal of Economic Theory*, 22(1), 67–86.

Yitzhaki, S. (1983). "On an Extension of the Gini Inequality Index." *International Economic Review*, 24(3), 617–628.

Examples

```
d <- iq_sample_data("income")

# Standard Gini (delta = 2)
iq_sgini(d$income, delta = 2)

# More weight on the bottom of the distribution
iq_sgini(d$income, delta = 3)

# With bootstrap CIs
iq_sgini(d$income, delta = 3, ci = TRUE, R = 200)
```

---

iq_shares *Income shares by quantile*

Description

Computes the share of total income held by each segment of the distribution. Default segments: bottom 50%, middle 40%, top 10%, and top 1%.

Usage

```
iq_shares(
  x,
  weights = NULL,
  breaks = c(0.5, 0.9, 0.99, 1),
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95,
  negatives = c("error", "keep")
)
```

Arguments

x	Numeric vector of incomes.
weights	Optional numeric vector of survey weights.
breaks	Numeric vector of cumulative population thresholds defining the segments. Default c(0.50, 0.90, 0.99, 1.00).
na.rm	Logical. Remove NA values? Default FALSE.
ci	Logical. Compute bootstrap confidence intervals on each share? Default FALSE.
R	Integer. Number of bootstrap replicates. Default 1000.
level	Numeric. Confidence level. Default 0.95.
negatives	Character. "error" (default) aborts on negatives; "keep" permits them.

Details

Distributions containing negative values can produce shares that fall outside the unit interval: the bottom segment may have a negative share (it pulls total income down), and other segments may exceed 100% as a result. The function returns the raw shares and emits a warning when negatives are present so the user can interpret accordingly. If the population total income is non-positive (so shares are not well-defined at all), the function returns NA shares with a warning.

Value

An S3 object of class "iq_shares" with elements:

shares data.frame with columns segment, pop_share, income_share, and (when ci = TRUE) ci_lower and ci_upper.

n Integer. Number of observations.

level Numeric or NULL. Confidence level.

has_negatives Logical. Whether the input contained negatives.

Examples

```
d <- iq_sample_data("income")
iq_shares(d$income)

# With bootstrap CIs on each share
iq_shares(d$income, ci = TRUE, R = 200)

# Custom breaks: quintiles
iq_shares(d$income, breaks = c(0.20, 0.40, 0.60, 0.80, 1.00))

# Wealth distributions can include negative net worth
wealth <- c(-5000, -1000, 0, 5000, 20000, 80000, 250000, 1e6)
iq_shares(wealth, negatives = "keep")
```

`iq_theil` *Theil index and generalised entropy measures*

Description

Computes the Theil T index (GE(1)), Theil L / mean log deviation (GE(0)), or a generalised entropy index GE(alpha) for any non-negative alpha.

Usage

```
iq_theil(
  x,
  weights = NULL,
  index = "T",
  na.rm = FALSE,
  ci = FALSE,
  R = 1000L,
  level = 0.95
)
```

Arguments

`x`	Numeric vector of incomes (strictly positive).
`weights`	Optional numeric vector of survey weights.
`index`	Character or numeric. `"T"` for Theil T (GE(1)), `"L"` for mean log deviation (GE(0)), or a numeric value for GE(alpha). Default `"T"`.
`na.rm`	Logical. Remove `NA` values? Default `FALSE`.
`ci`	Logical. Compute bootstrap confidence intervals? Default `FALSE`.
`R`	Integer. Number of bootstrap replicates. Default `1000`.
`level`	Numeric. Confidence level. Default `0.95`.

Details

Generalised entropy indices are the only class of inequality measures that are both decomposable by population subgroups and satisfy the transfer principle. Higher values indicate more inequality.

Theil T (GE(1)) and Theil L (GE(0)) involve `log(x)` and so require strictly positive values. GE(alpha) for `alpha > 1` is well-defined for non-negative `x` but is highly sensitive to small or zero values. For wealth or income net of taxes/transfers (which can be zero or negative) use the Gini, S-Gini, or Kolm index instead.

Note on cross-validation against **ineq**: this package uses the textbook GE(alpha) convention, where `index = "T"` is GE(1) (Theil T) and `index = "L"` is GE(0) (mean log deviation). The legacy **ineq** package uses the opposite indexing, so `ineq::Theil(x, parameter = 0)` matches `iq_theil(x, "T")` and `ineq::Theil(x, parameter = 1)` matches `iq_theil(x, "L")`.

Value

An S3 object of class `"iq_theil"` with elements:

value Numeric. The index value.

alpha Numeric. The alpha parameter used.

index_name Character. Human-readable name of the index.

n Integer. Number of observations.

se Numeric or `NULL`. Bootstrap standard error.

ci_lower Numeric or `NULL`. Lower bound of the CI.

ci_upper Numeric or `NULL`. Upper bound of the CI.

level Numeric or `NULL`. Confidence level.

References

Theil, H. (1967). *Economics and Information Theory*. Amsterdam: North-Holland.

Cowell, F. A. (2011). *Measuring Inequality*. 3rd edition. Oxford University Press.

Shorrocks, A. F. (1980). "The Class of Additively Decomposable Inequality Measures." *Econometrica*, 48(3), 613–625.

Examples

```
d <- iq_sample_data("income")

# Theil T (GE(1))
iq_theil(d$income, index = "T")

# With bootstrap CIs
iq_theil(d$income, index = "T", ci = TRUE, R = 200)

# Mean log deviation (GE(0))
iq_theil(d$income, index = "L")

# GE(2): half the squared coefficient of variation
iq_theil(d$income, index = 2)
```

Index