

# Package ‘segregation’

January 14, 2019

**Type** Package

**Title** Entropy-Based Segregation Indices

**Version** 0.2.0

**Description** Computes entropy-based segregation indices, as developed by Theil (1971) <isbn:978-0471858454>, with a focus on the Mutual Information Index (M) and Theil's Information Index (H). The M, further described by Mora and Ruiz-Castillo (2011) <doi:10.1111/j.1467-9531.2011.01237.x> and Frankel and Volij (2011) <doi:10.1016/j.jet.2010.10.008>, is a measure of segregation that is highly decomposable. The package provides tools to decompose the index by units and groups (local segregation), and by within and between terms. Includes standard error estimation by bootstrapping.

**License** MIT + file LICENSE

**Imports** data.table

**Encoding** UTF-8

**LazyData** true

**Suggests** testthat, covr, linter

**URL** <http://github.com/elbersb/segregation>

**BugReports** <http://github.com/elbersb/segregation/issues>

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Author** Benjamin Elbers [aut, cre] (<<https://orcid.org/0000-0001-5392-3448>>)

**Maintainer** Benjamin Elbers <be2239@columbia.edu>

**Repository** CRAN

**Date/Publication** 2019-01-14 07:10:03 UTC

## R topics documented:

entropy	2
ipf	3
mutual_difference	4
mutual_local	6
mutual_total	8
mutual_within	9
schools00	10
schools05	11
segregation	12

<b>Index</b>	<b>13</b>
--------------	-----------

---

entropy	<i>Calculates the entropy of a distribution</i>
---------	---

---

### Description

Returns the entropy of the distribution defined by group.

### Usage

```
entropy(data, group, weight = NULL, base = exp(1))
```

### Arguments

data	A data frame.
group	A categorical variable or a vector of variables contained in data.
weight	Numeric. Only frequency weights are allowed. (Default NULL)
base	Base of the logarithm that is used in the entropy calculation. Defaults to the natural logarithm.

### Value

A single number, the entropy.

### Examples

```
d <- data.frame(cat = c("A", "B"), n = c(25, 75))
entropy(d, "cat", weight = "n") # => .56
# this is equivalent to  $-.25 \times \log(.25) - .75 \times \log(.75)$ 

d <- data.frame(cat = c("A", "B"), n = c(50, 50))
# use base 2 for the logarithm, then entropy is maximized at 1
entropy(d, "cat", weight = "n", base = 2) # => 1
```

---

ipf	<i>Adjustment of marginal distributions using iterative proportional fitting</i>
-----	--

---

### Description

Adjusts the marginal distributions for `group` and `unit` in `source` to the respective marginal distributions in `target`, using the iterative proportional fitting algorithm (IPF).

### Usage

```
ipf(source, target, group, unit, weight = NULL, max_iterations = 100,
     precision = 0.001)
```

### Arguments

<code>source</code>	A "source" data frame. The marginals of this dataset are adjusted to the marginals of <code>target</code> .
<code>target</code>	A "target" data frame. The function returns a dataset where the marginal distributions of <code>group</code> and <code>unit</code> categories are approximated by those of <code>target</code> .
<code>group</code>	A categorical variable or a vector of variables contained in <code>source</code> and <code>target</code> . Defines the first distribution for adjustment.
<code>unit</code>	A categorical variable or a vector of variables contained in <code>source</code> and <code>target</code> . Defines the second distribution for adjustment.
<code>weight</code>	Numeric. Only frequency weights are allowed. (Default NULL)
<code>max_iterations</code>	Maximum number of iterations used for the IPF algorithm.
<code>precision</code>	Convergence criterion for the IPF algorithm. In every iteration, the ratio of the source and target marginals are calculated for every category of <code>group</code> and <code>unit</code> . The algorithm converges when all ratios are smaller than $1 + \text{precision}$ .

### Details

The algorithm works by scaling the marginal distribution of `group` in the source data frame towards the marginal distribution of `target`; then repeating this process for `unit`. The algorithm then keeps alternating between `group` and `unit` until the marginals of the adjusted data frame are within the allowed precision. This results in a dataset that retains the association structure of `source` while approximating the marginal distribution of `target`. If the number of `unit` and `group` categories is different in `source` and `target`, the data frame returns the combination of `unit` and `group` categories that occur in both datasets. Zero values are replaced by a small, non-zero number ( $1e-4$ ).

### Value

Returns a dataset that retains the association structure of `source` while approximating the marginal distributions for `group` and `unit` of `target`. The dataset identifies each combination of `group` and `unit`, and categories that only occur in either `source` or `target` are dropped. The adjusted frequency of each combination is given by the column `n`, while `n_target` and `n_source` contain the zero-adjusted frequencies in the target and source dataset, respectively.

## References

- W. E. Deming and F. F. Stephan. 1940. "On a Least Squares Adjustment of a Sampled Frequency Table When the Expected Marginal Totals are Known". *Annals of Mathematical Statistics*. 11 (4): 427–444.
- T. Karmel and M. Maclachlan. 1988. "Occupational Sex Segregation — Increasing or Decreasing?" *Economic Record* 64: 187-195.

## Examples

```
# adjusts the marginals of group and unit categories so that
# schools00 has similar marginals as schools05
adj <- ipf(schools00, schools05, "race", "school", weight = "n")

# check that the new "race" marginals are similar to the target marginals
# (the same could be done for schools)
aggregate(adj$n, list(adj$race), sum)
aggregate(adj$n_target, list(adj$race), sum)

# note that the adjusted dataset contains fewer
# schools than either the source or the target dataset,
# because the marginals are only defined for the overlap
# of schools
length(unique(schools00$school))
length(unique(schools05$school))
length(unique(adj$school))
```

---

mutual\_difference      *Decomposes the difference between two M indices*

---

## Description

Uses one of three methods to decompose the difference between two M indices: (1) "shapley": a method based on the Shapley decomposition with a few advantages over the Karmel-Maclachlan method (recommended and the default, Deutsch et al. 2006), (2) "km": the method based on Karmel-Maclachlan (1988), (3) "mrc": the method developed by Mora and Ruiz-Castillo (2009). All methods have been extended to account for missing units/groups in either data input.

## Usage

```
mutual_difference(data1, data2, group, unit, weight = NULL,
  method = "shapley", se = FALSE, n_bootstrap = 50, base = exp(1),
  ...)
```

## Arguments

data1                    A data frame with same structure as data2.

data2                    A data frame with same structure as data1.

group	A categorical variable or a vector of variables contained in data. Defines the first dimension over which segregation is computed.
unit	A categorical variable or a vector of variables contained in data. Defines the second dimension over which segregation is computed.
weight	Numeric. Only frequency weights are allowed. (Default NULL)
method	Either "shapley" (the default), "km" (Karmel and Maclachlan method), or "mrc" (Mora and Ruiz-Castillo method).
se	If TRUE, standard errors are estimated via bootstrap. (Default FALSE)
n_bootstrap	Number of bootstrap iterations. (Default 50)
base	Base of the logarithm that is used in the calculation. Defaults to the natural logarithm.
...	Only used for additional arguments when when method is set to shapley or km. See <a href="#">ipf</a> for details.

### Details

The Shapley method is an improvement over the Karmel-Maclachlan method (Deutsch et al. 2006). It is based on several margins-adjusted data inputs and yields symmetrical results (i.e. data1 and data2 can be switched).

The Karmel-Maclachlan method (Karmel and Maclachlan 1988) adjusts the margins of data1 to be similar to the margins of data2. This process is not symmetrical.

The Shapley and Karmel-Maclachlan methods are based on iterative proportional fitting (IPF), first introduced by Deming and Stephan (1940). Depending on the size of the dataset, this may take a few seconds (see [ipf](#) for details).

The method developed by Mora and Ruiz-Castillo (2009) uses an algebraic approach to estimate the size of the components. This will often yield substantively different results from the Shapley and Karmel-Maclachlan methods. Note that this method is not symmetric in terms of what is defined as group and unit categories, which may yield contradictory results.

A problem arises when there are group and/or unit categories in data1 that are not present in data2 (or vice versa). All methods estimate the difference only for categories that are present in both datasets, and report additionally the change in M that is induced by these cases as additions (present in data2, but not in data1) and removals (present in data1, but not in data2).

### Value

Returns a data frame with columns `stat` and `est`. The data frame contains the following rows defined by `stat`: `M1` contains the M for data1. `M2` contains the M for data2. `diff` is the difference between M2 and M1. The sum of all rows following `diff` equal `diff`.

`additions` contains the change in M induces by unit and group categories present in data2 but not data1, and `removals` the reverse.

All methods return the following three terms: `unit_marginal` is the contribution of unit composition differences. `group_marginal` is the contribution of group composition differences. `structural` is the contribution unexplained by the marginal changes, i.e. the structural difference. Note that the interpretation of these terms depend on the exact method used.

When using "km", one additional row is returned: interaction is the contribution of differences in the joint marginal distribution of unit and group.

If se is set to TRUE, an additional column se contains the associated bootstrapped standard errors, and the column est contains bootstrapped estimates.

## References

W. E. Deming, F. F. Stephan. 1940. "On a Least Squares Adjustment of a Sampled Frequency Table When the Expected Marginal Totals are Known." *The Annals of Mathematical Statistics* 11(4): 427-444.

T. Karmel and M. Maclachlan. 1988. "Occupational Sex Segregation — Increasing or Decreasing?" *Economic Record* 64: 187-195.

R. Mora and J. Ruiz-Castillo. 2009. "The Invariance Properties of the Mutual Information Index of Multigroup Segregation." *Research on Economic Inequality* 17: 33-53.

J. Deutsch, Y. Flückiger, and J. Silber. 2009. "Analyzing Changes in Occupational Segregation: The Case of Switzerland (1970–2000)." *Research on Economic Inequality* 17: 171–202.

## Examples

```
# decompose the difference in school segregation between 2000 and 2005,
# using the Shapley method
mutual_difference(schools00, schools05, group = "race", unit = "school",
  weight = "n", method = "shapley", precision = .1)
# => the structural component is close to zero, thus most change is in the marginals.
# This method gives identical results when we switch the unit and group definitions,
# and when we switch the data inputs.

# the Karmel-Maclachlan method is similar, but only adjust the data in the forward direction...
mutual_difference(schools00, schools05, group = "school", unit = "race",
  weight = "n", method = "km", precision = .1)

# ...this means that the results won't be identical when we switch the data inputs
mutual_difference(schools05, schools00, group = "school", unit = "race",
  weight = "n", method = "km", precision = .1)

# the MRC method indicates a much higher structural change...
mutual_difference(schools00, schools05, group = "race", unit = "school",
  weight = "n", method = "mrc")

# ...and is not symmetric
mutual_difference(schools00, schools05, group = "school", unit = "race",
  weight = "n", method = "mrc")
```

**Description**

Returns local segregation indices for each category defined by unit.

**Usage**

```
mutual_local(data, group, unit, weight = NULL, se = FALSE,
             n_bootstrap = 10, base = exp(1), wide = FALSE)
```

**Arguments**

data	A data frame.
group	A categorical variable or a vector of variables contained in data. Defines the dimension over which segregation is computed.
unit	A categorical variable or a vector of variables contained in data. Defines the group for which local segregation indices are calculated.
weight	Numeric. Only frequency weights are allowed. (Default NULL)
se	If TRUE, standard errors are estimated via bootstrap. (Default FALSE)
n_bootstrap	Number of bootstrap iterations. (Default 10)
base	Base of the logarithm that is used in the calculation. Defaults to the natural logarithm.
wide	Returns a wide dataframe instead of a long dataframe. (Default FALSE)

**Value**

Returns a data frame with two rows for each category defined by unit, for a total of  $2 \times (\text{number of units})$  rows. The column `est` contains two statistics that are provided for each unit: `ls`, the local segregation score, and `p`, the proportion of the unit from the total number of cases. If `se` is set to TRUE, an additional column `se` contains the associated bootstrapped standard errors, and the column `est` contains bootstrapped estimates. If `wide` is set to TRUE, returns instead a wide dataframe, with one row for each unit, and the associated statistics in separate columns.

**References**

Henri Theil. 1971. Principles of Econometrics. New York: Wiley.

Ricardo Mora and Javier Ruiz-Castillo. 2011. "Entropy-based Segregation Indices". Sociological Methodology 41(1): 159–194.

**Examples**

```
# which racial groups are most segregated?
(localseg = mutual_local(schools00, "school", "race",
                        weight="n", wide = TRUE))

sum(localseg$p) # => 1

# the sum of the weighted local segregation scores equals
# total segregation
```

```
sum(localseg$l1s * localseg$p) # => .425
mutual_total(schools00, "school", "race", weight="n") # M => .425
```

---

mutual_total	<i>Calculate total segregation for M and H</i>
--------------	--

---

### Description

Returns the total segregation between group and unit. If within is given, calculates segregation within each within category separately, and takes the weighted average. Also see [mutual\\_within](#) for detailed within calculations.

### Usage

```
mutual_total(data, group, unit, within = NULL, weight = NULL,
             se = FALSE, n_bootstrap = 10, base = exp(1))
```

### Arguments

data	A data frame.
group	A categorical variable or a vector of variables contained in data. Defines the first dimension over which segregation is computed.
unit	A categorical variable or a vector of variables contained in data. Defines the second dimension over which segregation is computed.
within	A categorical variable or a vector of variables contained in data. The variable(s) should be a superset of either the unit or the group for the calculation to be meaningful. If provided, segregation is computed within the groups defined by the variable, and then averaged. (Default NULL)
weight	Numeric. Only frequency weights are allowed. (Default NULL)
se	If TRUE, standard errors are estimated via bootstrap. (Default FALSE)
n_bootstrap	Number of bootstrap iterations. (Default 10)
base	Base of the logarithm that is used in the calculation. Defaults to the natural logarithm.

### Value

Returns a data frame with two rows. The column est contains the Mutual Information Index, M, and Theil's Entropy Index, H. The H is the the M divided by the group entropy. If within was given, M and H are weighted averages of the within-category segregation scores. If se is set to TRUE, an additional column se contains the associated bootstrapped standard errors, and the column est contains bootstrapped estimates.

### References

Henri Theil. 1971. *Principles of Econometrics*. New York: Wiley.

Ricardo Mora and Javier Ruiz-Castillo. 2011. "Entropy-based Segregation Indices". *Sociological Methodology* 41(1): 159–194.



**Examples**

```
# calculate school racial segregation
mutual_total(schools00, "school", "race", weight="n") # M => .425

# note that the definition of groups and units is arbitrary
mutual_total(schools00, "race", "school", weight="n") # M => .425

# if groups or units are defined by a combination of variables,
# vectors of variable names can be provided -
# here there is no difference, because schools
# are nested within districts
mutual_total(schools00, "race", c("district", "school"),
             weight="n") # M => .424

# estimate standard errors for M and H
mutual_total(schools00, "race", "school", weight="n", se=TRUE)

# estimate segregation within school districts
mutual_total(schools00, "race", "school",
             within="district", weight="n") # M => .087

# estimate between-district racial segregation
mutual_total(schools00, "race", "district", weight="n") # M => .338
# note that the sum of within-district and between-district
# segregation equals total school-race segregation;
# here, most segregation is between school districts
```

---

mutual\_within

---

*Calculate detailed within-category segregation scores for M and H*


---

**Description**

Calculates the segregation between group and unit within each category defined by within.

**Usage**

```
mutual_within(data, group, unit, within, weight = NULL, se = FALSE,
              n_bootstrap = 10, base = exp(1), wide = FALSE)
```

**Arguments**

data	A data frame.
group	A categorical variable or a vector of variables contained in data. Defines the first dimension over which segregation is computed.
unit	A categorical variable or a vector of variables contained in data. Defines the second dimension over which segregation is computed.
within	A categorical variable or a vector of variables contained in data that defines the within-segregation categories.

weight	Numeric. Only frequency weights are allowed. (Default NULL)
se	If TRUE, standard errors are estimated via bootstrap. (Default FALSE)
n_bootstrap	Number of bootstrap iterations. (Default 10)
base	Base of the logarithm that is used in the calculation. Defaults to the natural logarithm.
wide	Returns a wide dataframe instead of a long dataframe. (Default FALSE)

### Value

Returns a data frame with four rows for each category defined by `within`. The column `est` contains four statistics that are provided for each unit:  $M$  is the within-category  $M$ , and  $p$  is the proportion of the category. Multiplying  $M$  and  $p$  gives the contribution of each within-category towards the total  $M$ .  $H$  is the within-category  $H$ , and `h_weight` provides the weight. Multiplying  $H$  and `h_weight` gives the contribution of each within-category towards the total  $H$ . `h_weight` is defined as  $p * EW/E$ , where  $EW$  is the within-category entropy, and  $E$  is the overall entropy. If `se` is set to TRUE, an additional column `se` contains the associated bootstrapped standard errors, and the column `est` contains bootstrapped estimates. If `wide` is set to TRUE, returns instead a wide dataframe, with one row for each `within` category, and the associated statistics in separate columns.

### References

- Henri Theil. 1971. Principles of Econometrics. New York: Wiley.
- Ricardo Mora and Javier Ruiz-Castillo. 2011. "Entropy-based Segregation Indices". Sociological Methodology 41(1): 159–194.

### Examples

```
(within <- mutual_within(schools00, "race", "school", within = "state",
                        weight = "n", wide = TRUE))
# the M for "AL" is .409
# manual calculation
schools_AL <- schools00[schools00$state=="AL",]
mutual_total(schools_AL, "race", "school", weight = "n") # M => .409

# to recover the within M and H from the output, multiply
# p * M and h_weight * H, respectively
sum(within$p * within$M) # => .326
sum(within$H * within$h_weight) # => .321
# compare with:
mutual_total(schools00, "race", "school", within = "state", weight = "n")
```

**Description**

Fake dataset used for examples. Loosely based on data provided by the National Center for Education Statistics, Common Core of Data, with information on U.S. primary schools in three U.S. states. The original data can be downloaded at <https://nces.ed.gov/ccd/>.

**Usage**

schools00

**Format**

A data frame with 8,142 rows and 5 variables:

**state** either A, B, or C

**district** school agency/district ID

**school** school ID

**race** either native, asian, hispanic, black, or white

**n** n of students by school and race

---

schools05

*Ethnic/racial composition of schools for 2005/2006*

---

**Description**

Fake dataset used for examples. Loosely based on data provided by the National Center for Education Statistics, Common Core of Data, with information on U.S. primary schools in three U.S. states. The original data can be downloaded at <https://nces.ed.gov/ccd/>.

**Usage**

schools05

**Format**

A data frame with 8,013 rows and 5 variables:

**state** either A, B, or C

**district** school agency/district ID

**school** school ID

**race** either native, asian, hispanic, black, or white

**n** n of students by school and race

---

segregation

*segregation: Entropy-based segregation indices*

---

**Description**

Calculate and decompose entropy-based, multigroup segregation indices, with a focus on the Mutual Information Index (M) and Theil's Information Index (H). Provides tools to decompose the measures by groups and units, and by within and between terms. Includes standard error estimation by bootstrapping.

**See Also**

<https://elbersb.de/segregation>

# Index

## \*Topic **datasets**

schools00, [10](#)

schools05, [11](#)

entropy, [2](#)

ipf, [3](#), [5](#)

mutual\_difference, [4](#)

mutual\_local, [6](#)

mutual\_total, [8](#)

mutual\_within, [8](#), [9](#)

schools00, [10](#)

schools05, [11](#)

segregation, [12](#)

segregation-package (segregation), [12](#)