Package 'fpsim'

May 8, 2026

Title Compute Measures of Foreign Policy Similarity/Agreement

Version 0.1.0

Maintainer Steven Miller <steve@svmiller.com>

Description Provides functions for calculating various measures of foreign policy similarity or association commonly used in the study of international relations. These include Signorino and Ritter's S statistic (weighted and unweighted), Cohen's weighted kappa, Scott's pi, and Kendall's tau-b. The package facilitates the generation of dyadic similarity scores for empirical analyses and can also serve as an educational resource for understanding how such measures are derived.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 3.6.0)

LazyData true

Suggests peacesciencer

NeedsCompilation no

Author Steven Miller [aut, cre] (ORCID: <https://orcid.org/0000-0003-4072-6263>)

Repository CRAN

Date/Publication 2025-08-18 14:20:28 UTC

Contents

cohenk *Calculate Cohen's (1960, 1968) weighted kappa*

Description

cohenk() takes two vectors and returns Cohen's kappa as an estimate of chance-corrected agreement.

Usage

```
cohenk(x1, x2, w_exp = 2, levels = NULL)
```

Arguments

x1	a vector, and one assumes an integer
x2	a vector, and one assumes an integer
w_exp	an exponent to apply to the weight matrix. Default is 2 for squared distances in the weight matrix. Supplying a 1 would make for linear distances.
levels	defaults to NULL, but an optional vector that defines the full sequence of values that could be observed in x1 and x2. If NULL, the function looks for observed values.

Details

The function subsets to complete cases of the two vectors for which you want Cohen's kappa.

The function implicitly assumes that x1 and x2 are columns in a data frame. One indirect check for this looks at whether x1 and x2 are the same length. The function will stop if they're not.

There will sometimes be instances, assuredly with alliances, where not all categories are observed. For example, the toy example I provide of Germany and Russia in 1914 includes no 2s. In the language of "ratings", the "rating" of 2 was available for Germany and Russia in 1914 but neither side used it. The levels argument allows you to specify the full sequence of values that could be observed, even if none were. It probably makes the most sense to always use this argument, even if the default behavior operates as if you won't.

Value

cohenk() takes two vectors and returns Cohen's kappa as an estimate of chance-corrected agreement.

References

Cohen, Jacob. 1960. "A Coefficient of Agreement for Nominal Scales." *Educational and Psychological Measurement* 20(1): 37-46.

Cohen, Jacob. 1968. "Weighted Kappa: Nominal Scale Agreement with Provision for Scaled Disagreement or Partial Credit." *Psychological Bulletin* 70(4): 213–220.

Examples

```
cohenk(gmyrus14$gmy, gmyrus14$rus, levels = 0:3) # with levels argument
cohenk(usamex46$vote1, usamex46$vote2) # levels argument not necessary here.
```

`gmyrus14` *Select Alliance Portfolios of Germany and Russia, 1914*

Description

A simple example of alliance portfolios of Germany and Russia, by way of Signorino and Ritter's (1999) example.

Usage

```
gmyrus14
```

Format

A data frame with 20 observations on the following 4 variables.

`state` a three-character code indicating a state

`syscap` the capabilities of the state, as a potential weight

`gmy` the alliance commitment for Germany with the state identified in the `state` column

`rus` the alliance commitment for Russia with the state identified in the `state` column

Details

The data come by way of Signorino and Ritter (1999, Table 6).

References

Signorino, Curtis S. and Jeffrey M. Ritter. "Tau-b or Not Tau-B: Measuring the Similarity of Foreign Policy Positions." *International Studies Quarterly* 43(1): 115–44.

spi *Calculate Scott's (1955) pi*

Description

spi() takes two vectors and returns Scott's (1955) pi coefficient, communicating extent of inter-observer reliability.

Usage

```
spi(x1, x2, levels = NULL)
```

Arguments

x1	a vector, and one assumes an integer
x2	a vector, and one assumes an integer
levels	defaults to NULL, but an optional vector that defines the full sequence of values that could be observed in x1 and x2. If NULL, the function looks for observed values.

Details

The function subsets to complete cases of the two vectors for which you want Scott's pi.

The function implicitly assumes that x1 and x2 are columns in a data frame. One indirect check for this looks at whether x1 and x2 are the same length. The function will stop if they're not.

There will sometimes be instances, assuredly with alliances, where not all categories are observed. For example, the toy example I provide of Germany and Russia in 1914 includes no 2s. In the language of "ratings", the "rating" of 2 was available for Germany and Russia in 1914 but neither side used it. The levels argument allows you to specify the full sequence of values that could be observed, even if none were. It probably makes the most sense to always use this argument, even if the default behavior operates as if you won't.

Value

spi() takes two vectors and returns Scott's (1955) pi coefficient, communicating extent of inter-observer reliability.

References

Scott, William A. 1955. "Reliability of Content Analysis: The Case of Nominal Scale Coding." *Public Opinion Quarterly* 19(3): 321–5.

Examples

```
spi(gmyrus14$gmy, gmyrus14$rus, levels = 0:3) # with levels argument
spi(usamex46$vote1, usamex46$vote2) # levels argument not necessary here.
```

srs *Calculate Signorino and Ritter's (1999) S for Similarity*

Description

srs() takes two vectors and returns Signorino and Ritter's S statistic communicating broadly understood "similarity" of interests or ratings.

Usage

```
srs(x1, x2, distances = "absolute", weights = NULL, range = NULL)
```

Arguments

x1	a vector, and one assumes an integer
x2	a vector, and one assumes an integer
distances	the type of distances between ratings/attachments to estimate. Can be either "absolute" or "squared". Defaults to "absolute", but see note in details section.
weights	a vector of weights. Defaults to NULL for creating unweighted S statistics
range	defaults to NULL, but an optional vector that forces the range to be a certain value. If NULL, the function calculates a range based on the maximum and minimum values observed across both x1 and x2. See details section for more.

Details

Be advised that Signorino and Ritter's (1999) treatment of the S statistic used absolute distances when squared distances are more commonly used in the world of distance and association metrics.

There are potentially instances in which the conceivable range of ratings/attachments (i.e. your two vectors) are not observed. In the case of applications to alliance data, this is almost an impossibility. Every state, by assumption, is maximally committed to defending itself. There will assuredly be cases in which there is no commitment to another state in the data (either for reasons of disinterest or enmity, though the first calls into question what a 0 should communicate and the latter betrays the interesting complexity of alliances). Thus, the minimum and maximum, one assumes, will always be observed in the alliance data. Perhaps the same could be said for UN voting data, though I couldn't rule out the possibility that there is a dyad out there for which both states never voted "yes" or "no". That would have implications for the range in the denominator of the formula. You can override that by hard-setting the range in the range argument.

The function subsets to complete cases of the two vectors for which you want an S score. If weights are included, the function further subsets to complete cases including the weights as well.

The function implicitly assumes that x1 and x2 are columns in a data frame. One indirect check for this looks at whether x1 and x2 are the same length. The function will stop if they're not.

Several Comments on Weighting:

If it were my call to make, I'd caution against the IR standard of using the composite index of national capabilities (CINC) as a weight on the calculation of the S statistic. Conceptually,

weighting by capabilities tries to capture some kind of "importance" quantity. Related to the familiar application of alliances, this would prioritize those states that could conceivably bring more to the battlefield. In practice, this adds one anachronism to another. Capabilities, as measured, are basically a nineteenth century measurement for which estimates of energy consumption, iron and steel production, and urban population size are given equal weight in composition of the measure to military expenditures and military size. Alliances themselves are somewhat antiquarian, certainly in what we want them to do for this measure. If the question is "why must alliances be measures of foreign policy similarity", the answer kind of reduces to "we have historical data on them." If you want estimates for the 19th century, you have this, but then are implicitly confessing your measure of foreign policy similarity is an anachronism.

There are other peculiarities too. The data on capabilities has always been historically skewed to the right. Very few states have proportionally that much weight. As the state system has expanded in size (i.e. as empires ended), the relative weight at the top necessarily decreases. For example, the top 3 states in capabilities in 1816 (the United Kingdom, Russia, and France) combined for 61.8% of capabilities in a system of just 23 states. In 2016, the top three states (China, the United States, and India) combined for 45% of capabilities in a system of 195 states. New states are almost always small states that possess almost no capabilities. 11 of 23 states in 1816 had less than 1% of capabilities. That's about 48% of the system. In 2016, 176 of 195 states have less than 1% of capabilities. That's over 90% of the system. If the idea is to identify the "important" foreign policy ties, I echo Haege's (2011) contention that this approach is a second-best solution. It's second-best to other metrics that better model chance-corrected agreement. It just discards too much information and gives too much weight to great powers and/or states that are conspicuously high on capabilities (e.g. India).

Faithfully calculating a weighted S statistic (by system capabilities) requires a weight that sums to 1. In the most literal sense of 1, there is no year in the National Material Capabilities data (v. 2016) in which system capabilities in a given year sum to 1. In almost 60% of cases/years, the discrepancy doesn't look like a rounding error either. In 1860, all capabilities sum to over 1.07! In the context of applications with Correlates of War's CINC scores, you can still use the raw data because the function doesn't assume the weights sum to 1. You'll see how in the denominator of the formula.

Weights are only applicable to absolute distances. If you specify a weight variable with `distances = 'squared'`, the function will ignore your weights.

In applications to the Correlates of War system, as far as I am aware, there are no CoW states for which there isn't a CINC estimate. If, for some reason, a CINC score (or some other weight) is missing, the cases are dropped *before* weights are applied.

If weights are supplied, the weights must match the length of either `x1` or `x2`. The function builds in an implicit assumption that the weights are a column in the data frame you're using.

Value

`srs()` takes two vectors and returns Signorino and Ritter's S statistic communicating broadly understood "similarity" of interests or ratings.

References

Signorino, Curtis S. and Jeffrey M. Ritter. "Tau-b or Not Tau-B: Measuring the Similarity of Foreign Policy Positions." *International Studies Quarterly* 43(1): 115–44.

Examples

```
srs(gmyrus14$gmy, gmyrus14$rus, distances = 'absolute')
srs(gmyrus14$gmy, gmyrus14$rus, distances = 'squared')
srs(gmyrus14$gmy, gmyrus14$rus, distances = 'absolute', weights = gmyrus14$syscap)
```

---

taub *Calculate Kendall's (1938) Tau-B*

Description

`taub()` takes two vectors and returns Kendall's Tau-b as a measure of rank correlation.

Usage

```
taub(x1, x2)
```

Arguments

x1	a vector, and one assumes an integer
x2	a vector, and one assumes an integer

Details

I'll be honest that I wrote this just to say that I did write this and that my workflow would still lean on using the `cor()` function in base R.

Value

`taub()` takes two vectors and returns Kendall's Tau-b as a measure of rank correlation.

References

Kendall, Maurice G. 1938. "A New Measure of Rank Correlation". *Biometrika* 30(1/2): 81-93.

Examples

```
taub(usamex46$vote1, usamex46$vote2)
taub(gmyrus14$gmy, gmyrus14$rus)

# Compare with...

cor(usamex46$vote1, usamex46$vote2, method = 'kendall')
cor(gmyrus14$gmy, gmyrus14$rus, method = 'kendall')
```

`usamex46` *American-Mexican Dyadic Voting Patterns in the United Nations, 1946*

Description

A simple example of voting patterns for the United States and Mexico in the United Nations in 1946.

Usage

```
usamex46
```

Format

A data frame with 38 observations on the following 6 variables.

`resid` an identifier for a roll-call vote ID

`ccode1` the Correlates of War state code for the United States (2)

`ccode2` the Correlates of War state code for Mexico (70)

`year` a numeric constant for the year (1946)

`vote1` an integer for how the United States voted on the resolution identified in the `resid` column

`vote2` an integer for how Mexico voted on the resolution identified in the `resid` column

Details

Data are from a June 2024 of the United Nations voting data provided by Erik Voeten on his Dataverse for the project.

Valid vote values identified are 1 (yes), 2 (abstain), and 3 (no).

References

Bailey, Michael A., Anton Strezhnev, and Erik Voeten. 2017. "Estimating Dynamic State Preferences from United Nations Voting Data." *Journal of Conflict Resolution* 61(2): 430-56.

Index