Package 'fpsim'

July 5, 2026

Title Compute Measures of Foreign Policy Similarity/Agreement

Version 0.2.0

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), Benati and Capurri's chance-corrected S statistic (A), 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 calculated.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 3.6.0)

LazyData true

Suggests peacesciencer

Config/Needs/website rmarkdown

NeedsCompilation no

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

Repository CRAN

Date/Publication 2026-07-04 23:10:08 UTC

Contents

`bcai` *Calculate Benati and Capurri's (2026) alignment index (A)*

Description

`bcai()` takes two vectors and returns Benati and Capurri's (2026) alignment index (A).

Usage

```
bcai(x1, x2, distances = "absolute", weights = NULL, levels = 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 A index values.
`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

You can think of the alignment index that Benati and Capurri (2026) describe as an *S* corollary to the chance-corrected measures that Häge (2011) offers as substitutes for *S*. It takes the (unweighted, absolute distances) *S* score proposed by Signorino and Ritter (1999) and subtracts from it the *S* score that would follow under the assumption of independent voting.

The function subsets to complete cases of the two vectors for which you want an alignment score.

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.

A Few Caveats on Weighting:

You can weight this measure if you want. Please be mindful about what you're doing, especially if the weights are CINC scores. See here:

https://svmiller.com/blog/2026/06/alliances-weighting-foreign-policy-similarity/

The function will proportionalize your weights to sum to 1 if they do not sum to 1 already.

Value

`bcai()` takes two vectors and returns Benati and Capurri's (2026) alignment index (A).

References

Benati, Stefano, and Agnese Capurri. 2026. "The Alignment index and its application to voting at the United Nations General Assembly." *Quality & Quantity*. doi:10.1007/s1113502602814x

Examples

```
# with levels argument
bcai(gmyrus14$gmy, gmyrus14$rus, levels = 0:3)
# levels argument not necessary here.
bcai(bencapex$rowv, bencapex$colv)
# squared, with levels argument
bcai(gmyrus14$gmy, gmyrus14$rus, distances = 'squared', levels = 0:3)
```

---

bencapex *A Worked Example from Benati and Capurri (2026)*

Description

A simple worked example illustrating what the authors call an "alignment index." You can think of this as a kind of chance-corrected *S* score.

Usage

```
bencapex
```

Format

A data frame with 240 observations on the following 2 variables.

`rowv` how the row voter voted on a resolution

`colv` how the column voter voted on a resolution

Details

This is an expansion of Table 1 in their article. Valid vote values identified are 1 (yes), 2 (abstain), and 3 (no).

References

Benati, Stefano, and Agnese Capurri. 2026. "The Alignment index and its application to voting at the United Nations General Assembly." *Quality & Quantity*. doi:10.1007/s1113502602814x

---

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.
```

---

download.fpsim.data *Download FPSIM Data*

Description

`download.fpsim.data()` will download available pre-made dyadic foreign policy similarity data and place its contents the package's `extdata/` directory. It leverages R's `inst/` directory flexibility.

Usage

```
download.fpsim.data(confirm = FALSE, warning = TRUE, format = "rds")
```

Arguments

`confirm`	logical, defaults to FALSE. If FALSE, the function does not actually download the data. Set this to TRUE to confirm your intentions to download the data.
`warning`	logical, defaults to TRUE. If TRUE, the function returns a message advising you about the total size of the files you'll be downloading. If FALSE, no message is returned about the total file sizes.
`format`	a character vector determining what format to download. Defaults to "rds" for the R serialized data frame format native to R. "qs" downloads the data in a `.qs` file for more file compression, though reading it will depend on the **qs2** package.

Details

This function is named in such a way to avoid a function clash with the `download_extdata()` function in **peacesciencer**. It also comes from my aversion to the use of underscores in R function. There is nothing wrong with this convention, but the chatbots have ruined it for me.

This function will not download `FPSIM.rds`. The `download_extdata()` function in **peacesciencer** will do that.

Examples

```
download.fpsim.data()

# If you use this function, please inspect the extdata/ directory of the R
# package on your system. This function's output will tell you where it is.
# Thereafter, you can manually read a particular file into R. Something like
# this would work, though, if you had no idea where anything is.
#
# a <- paste0(system.file("extdata", package = "fpsim"),"/fpsim-votes-a.rds")
# Data <- readRDS(a)
# Data
```

`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, levels = 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
`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

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 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.

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. I expand a bit on this line of thinking on my blog here:

https://svmiller.com/blog/2026/06/alliances-weighting-foreign-policy-similarity/

The following was my original entry into this documentation file and I'll keep it as is for posterity. 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.

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.

The function will proportionalize your weights to sum to 1 if they do not sum to 1 already.

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', levels = c(0:3))
srs(gmyrus14$gmy, gmyrus14$rus, distances = 'squared', levels = c(0:3))
srs(gmyrus14$gmy, gmyrus14$rus, distances = 'absolute', weights = gmyrus14$syscap, levels = c(0:3))
```

---

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