# Package ‘icr’

October 31, 2024

**Type** Package

**Date** 2024-10-30

**Title** Compute Krippendorff's Alpha

**Version** 0.6.6

**Maintainer** Alexander Staudt <staudtlex@live.de>

**Description** Provides functions to compute and plot Krippendorff's inter-coder reliability coefficient alpha and bootstrapped uncertainty estimates (Krippendorff 2004, ISBN:0761915443). The bootstrap routines are set up to make use of parallel threads where supported.

**URL** https://github.com/staudtlex/icr

**BugReports** https://github.com/staudtlex/icr/issues

**License** GPL (>= 2)

**Encoding** UTF-8

**LazyData** true

**Imports** Rcpp (>= 0.12.9)

**LinkingTo** Rcpp

**Suggests** ggplot2, tinytest

**RoxygenNote** 7.3.2

**NeedsCompilation** yes

**Author** Alexander Staudt [aut, cre],
Pierre L'Ecuyer [ctb] (author of the C++ RNG code),
Chung-hong Chan [ctb]

**Repository** CRAN

**Date/Publication** 2024-10-31 17:00:02 UTC

## Contents

## `codings` *Example reliability data*

### Description

A matrix containing example codings of 12 units (e.g. newspaper articles) by four coders.

### Usage

```
codings
```

### Format

A matrix with 4 rows and 12 columns. Each column contains the coders' assessments of a coding unit (e.g. newspaper article)

### Source

Krippendorff, K. (1980). Content analysis: An introduction to its methodology. Beverly Hills, CA: Sage.

## `krippalpha` *Krippendorff's alpha*

### Description

`krippalpha` computes Krippendorff's reliability coefficient alpha.

### Usage

```
krippalpha(
  data,
  metric = "nominal",
  bootstrap = FALSE,
  bootnp = FALSE,
  nboot = 20000,
  nnp = 1000,
  cores = 1,
  seed = rep(12345, 6)
)
```

### Arguments

data — a matrix or data frame (coercible to a matrix) of reliability data. Data of type character are converted to numeric via as.factor().

metric — metric difference function to be applied to disagreements. Supports nominal, ordinal, interval, ratio, bipolar. Defaults to nominal.

bootstrap — logical indicating whether uncertainty estimates should be obtained using the bootstrap algorithm defined by Krippendorff. Defaults to FALSE.

bootnp — logical indicating whether non-parametric bootstrap uncertainty estimates should be computed. Defaults to FALSE.

nboot — number of bootstraps used in Krippendorff's algorithm. Defaults to 20000.

nnp — number of non-parametric bootstraps. Defaults to 1000.

cores — number of cores across which bootstrap-computations are distributed. Defaults to 1. If more cores are specified than available, the number will be set to the maximum number of available cores.

seed — numeric vector of length 6 for the internal L'Ecuyer-CMRG random number generator (see details). Defaults to c(12345, 12345, 12345, 12345, 12345, 12345).

### Details

krippalpha takes the seed vector to seed the internal random number generator of both bootstrap-routines. It does not advance R's RNG state.

When using the ratio metric with reliability data containing scales involving negative as well as positive values, krippalpha may return a value of NaN. The ratio metric difference function is defined as $\left(\frac{(c-k)}{(c+k)}\right)^2$. Hence, if for any two scale values $c = -k$, the fraction is not defined, resulting in $\alpha =$ NaN. In order to avoid this issue, shift your reliability data to have strictly positive values.

### Value

Returns a list of type icr with following elements:

alpha — value of inter-coder reliability coefficient

metric — integer representation of metric used to compute alpha: 1 nominal, 2 ordinal, 3 interval, 4 ratio, 6 bipolar

n_coders — number of coders

n_units — number of units to be coded

n_values — number of unique values in reliability data

coincidence_matrix — matrix containing coincidences within coder-value pairs

delta_matrix — matrix of metric differences depending on method

D_e — expected disagreement

D_o — observed disagreement

| | |
|---|---|
| `bootstrap` | `TRUE` if Krippendorff bootstrapping algorithm was run, `FALSE` otherwise |
| `nboot` | number of bootstraps |
| `bootnp` | `TRUE` if nonparametric bootstrap was run, `FALSE` otherwise |
| `nnp` | number of non-parametric bootstraps |
| `bootstraps` | vector of bootstrapped values of alpha (Krippendorff's algorithm) |
| `bootstrapsNP` | vector of non-parametrically bootstrapped values of alpha |

## Note

`krippalpha`'s bootstrap-routines use L'Ecuyer's CMRG random number generator (see L'Ecyuer et al. 2002) to create random numbers suitable for parallel computations. The routines interface to L'Ecuyer's C++ code, which can be found at https://www.iro.umontreal.ca/~lecuyer/myftp/streams00/c++/

## References

Krippendorff, K. (2004) *Content Analysis: An Introduction to Its Methodology*. Beverly Hills: Sage.

Krippendorff, K. (2011) *Computing Krippendorff's Alpha Reliability*. Departmental Papers (ASC) 43. https://web.archive.org/web/20220713195923/https://repository.upenn.edu/asc_papers/43/.

Krippendorff, K. (2016) *Bootstrapping Distributions for Krippendorff's Alpha*. https://www.asc.upenn.edu/sites/default/files/2021-03/Algorithm%20for%20Bootstrapping%20a%20Distribution%20of%20Alpha.pdf.

L'Ecuyer, P. (1999) Good Parameter Sets for Combined Multiple Recursive Random Number Generators. *Operations Research*, 47 (1), 159–164. https://www.iro.umontreal.ca/~lecuyer/myftp/streams00/opres-combmrg2-1999.pdf.

L'Ecuyer, P., Simard, R, Chen, E. J., and Kelton, W. D. (2002) An Objected-Oriented Random-Number Package with Many Long Streams and Substreams. *Operations Research*, 50 (6), 1073–1075. https://www.iro.umontreal.ca/~lecuyer/myftp/streams00/c++/streams4.pdf.

## Examples

```
data(codings)

# compute alpha, without uncertainty estimates
krippalpha(codings)

# additionally compute bootstrapped uncertainty estimates for alpha
alpha <- krippalpha(codings, metric = "nominal", bootstrap = TRUE, bootnp = TRUE)
alpha

# plot bootstrapped alphas
plot(alpha)

# alternatively, use ggplot2
df <- plot(alpha, return_data = TRUE)
```

```
library(ggplot2)
ggplot() +
  geom_line(data = df[df$ci_limit == FALSE, ], aes(x, y, color = type)) +
  geom_area(data = df[df$ci == TRUE, ], aes(x, y, fill = type), alpha = 0.4) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(legend.position = "bottom", legend.title = element_blank()) +
  ggtitle(expression(paste("Bootstrapped ", alpha))) +
  xlab("value") + ylab("density") +
  guides(fill = FALSE)
```

# Index