Package ‘circhelp’

July 4, 2024

Type Package

Title Circular Analyses Helper Functions

Version 1.1

Description Light-weight functions for computing descriptive statistics in different circular spaces (e.g., 2pi, 180, or 360 degrees), to handle angle-dependent biases, pad circular data, and more. Specifically aimed for psychologists and neuroscientists analyzing circular data. Basic methods are based on Jammalamadaka and SenGupta (2001) <doi:10.1142/4031>, removal of cardinal biases is based on the approach introduced in van Bergen, Ma, Pratte, & Jehee (2015) <doi:10.1038/nn.4150> and Chetverikov and Jehee (2023) <doi:10.1038/s41467-023-43251-w>.

URL `https://achetverikov.github.io/circhelp/index.html`,
`https://github.com/achetverikov/circhelp`

BugReports `https://github.com/achetverikov/circhelp/issues`

License CC0

Encoding UTF-8

LazyData true

Suggests testthat (>= 3.0.0), circular, knitr, rmarkdown, BAMBI, mgcv, ragg

Imports gamlss, MASS, mathjaxr, patchwork

Depends stats, data.table, ggplot2, R (>= 2.10)

Config/testthat/edition 3

RoxygenNote 7.3.2

VignetteBuilder knitr, patchwork, ggplot2, MASS, gamlss, mgcv, ragg

RdMacros mathjaxr

NeedsCompilation no

Author Andrey Chetverikov [aut, cre] (<`https://orcid.org/0000-0003-2767-6310`>),
Eline Van Geert [ctb] (<`https://orcid.org/0000-0002-7848-5998`>)

Maintainer Andrey Chetverikov <`andrey.chetverikov@uib.no`>

Repository CRAN

Date/Publication 2024-07-04 17:10:02 UTC

Contents

`angle_diff_rad` *Differences between angles in different circular spaces*

Description

Differences between angles in different circular spaces

Usage

```
angle_diff_rad(a, b)

angle_diff_360(a, b)

angle_diff_180(a, b)

angle_diff_90(a, b)

angle_diff_180_45(a, b)

angle_diff_360_90(a, b)
```

Arguments

a	first angle
b	second angle

Details

By default, all functions return values in ± half-range space (e.g., -pi to pi for 2pi radian space used by `angle_diff_rad()`) but `angle_diff_180_45()` and `angle_diff_360_90()` return values in [-1/4 range, 3/4 range] space

Value

difference between a and b

Functions

- `angle_diff_rad()`: angle difference in radians
- `angle_diff_360()`: angle difference in 360 degree space
- `angle_diff_180()`: angle difference in 180 degree space (e.g., line orientation)
- `angle_diff_90()`: angle difference in 90 degree space
- `angle_diff_180_45()`: angle difference in 180 degree space from -45 to 135
- `angle_diff_360_90()`: angle difference in 360 degree space from -90 to 270

Examples

```
angle_diff_180(5, 175)
angle_diff_360(5, 175)
angle_diff_90(5, 175)
angle_diff_rad(5, 175)

angle_diff_360(300, 0)
angle_diff_360_90(300, 0)
```

`Bae_Luck_2018_data` *Data from a motion estimation task*

Description

A dataset with the motion estimation results from Bae & Luck (2018) available from https://osf.io/4m2kb/ (some variables are removed, see the link for the full dataset).

Usage

```
Bae_Luck_2018_data
```

Format

A data frame with 20480 rows and 8 variables:

subject_Num observer ID

trial_Num trial number

TargetDirection true motion direction

RespAngle reported motion direction

motionCoh motion coherence

Block block number

Session session number

err estimation error

Source

`https://osf.io/4m2kb/download`

References

Bae, G.-Y., & Luck, S. J. (2018). Decoding motion direction using the topography of sustained ERPs and alpha oscillations. NeuroImage, 184(August 2018), 242–255. doi:10.1016/J.NEUROIMAGE.2018.09.029

`circ_corr` *Circular correlation coefficient*

Description

Computes a circular correlation coefficient as defined in Jammalamadaka & SenGupta (2001).

Usage

```
circ_corr(a, b, ill_defined = FALSE, mu = NULL, na.rm = FALSE)
```

Arguments

`a`	first variable
`b`	second variable
`ill_defined`	is one of the variables mean is not well-defined (e.g., it is uniformly distributed)?
`mu`	fix the mean parameter of both vectors to a certain value
`na.rm`	a logical value indicating whether NA values should be removed before the computation proceeds

Value

correlation coefficient

References

Jammalamadaka, S. R., & SenGupta, A. (2001). Topics in Circular Statistics. WORLD SCIENTIFIC. doi:10.1142/4031

Examples

```
requireNamespace("mgcv")
data <- mgcv::rmvn(10000, c(0, 0), V = matrix(c(1, 0.5, 0.5, 1), ncol = 2))
circ_corr(data[, 1], data[, 2])
```

---

circ_descr — *A set of descriptive statistics for circular data*

Description

A set of descriptive statistics for circular data

Usage

```
circ_descr(x, w = NULL, d = NULL, na.rm = FALSE)
```

Arguments

`x`	vector of angles
`w`	weights for the values in the vector
`d`	correction for the bias for data with known spacing
`na.rm`	a logical value indicating whether NA values should be removed before the computation proceeds

Value

a list with descriptive statistics

- mu - mean
- sigma - standard deviation
- skew_pewsey - skewness as defined by Pewsey
- skew_fischer - skewness as defined by Fischer
- rho - mean resultant length
- skew_rel_to_zero - skewness relative to zero

Examples

```
x <- c(rnorm(50, 0, 0.5), rnorm(20, 1, 0.5))
circ_descr(x)
```

circ_lin_corr *Circular-linear correlation*

Description

Implementation of the circular-linear correlation measure introduced by Mardia (1976) and Johnson and Wehrly (1977) as cited in Jammalamadaka & Sengupta (2001).

Usage

```
circ_lin_corr(circ_x, lin_x, na.rm = FALSE)
```

Arguments

circ_x	circular variable
lin_x	linear variable
na.rm	a logical value indicating whether NA values should be removed before the computation proceeds

Details

This measure is computed as

$$r^2 = (r_{xc}^2 + r_{xs}^2 - 2r_{xc}r_{xs}r_{cs})/(1 - r_{cs}^2)$$

where $r_{xc} = corr(x, cos(\alpha))$, $r_{xs} = corr(x, sin(\alpha))$, $r_{cs} = corr(cos(\alpha), sin(\alpha))$, and α and x are the circular and linear variables, respectively.

Value

circular-linear correlation measure

References

Jammalamadaka, S. R., & SenGupta, A. (2001). Topics in Circular Statistics. WORLD SCIENTIFIC. doi:10.1142/4031

Examples

```
x <- rnorm(50)
a <- as.vector(circular::rvonmises(50, 0, 5))
circ_lin_corr(x + a, x)
```

`circ_loess` *An implementation of circular-linear locally-weighted regression (LOESS)*

Description

Provides an locally-weighted average when the independent variable is circular and depended variable is linear. Mainly to use with ggplot2.

Usage

```
circ_loess(
  formula = NULL,
  data = NULL,
  angle = NULL,
  y = NULL,
  xseq = NULL,
  circ_space = NULL,
  span = 0.75,
  ...
)
```

Arguments

`formula`	the formula, e.g., y ~ x
`data`	data to use
`angle`	a vector of angles (not used if a formula is provided)
`y`	dependent variable vector (not used if a formula is provided)
`xseq`	a grid to compute predictions on (optional, the default is to use 500 points spanning the circle)
`circ_space`	circular space to use (90, 180, 360, or 2*pi)
`span`	a span to adjust the degree of smoothing
`...`	other arguments (ignored)

Details

Weights for the regression are computed as

$$w = (1 - (d/d_{max})^3)^3$$

where *d* is the angular difference between the point at which the estimate is computed and the angles in the data, and d_{max} is the maximum possible distance. If `span` is above 1, all points are included and $d_{max} = circ\_space/(4 * span)$. Otherwise, a proportion α of the points included based on their distance to the point at which the estimate is computed and d_{max} is the corresponding maximal distance.

Value

an object of `circ_loess` class with the following parameters:

- `angle` the angles in the data
- `y` the dependent variable vales in the data
- `xseq` the grid on which the loess values are estimated
- `y_est` the estimated loess values
- `y_se` standard errors
- `w` weights
- `circ_space` circular space used
- `span` span used

See Also

`stats::loess()`

Examples

```
p <- ggplot(Pascucci_et_al_2019_data, aes(x = orientation, y = err)) +
  geom_point(alpha = 0.05) +
  labs(x = "Orientation, deg.", y = "Error, deg.")
p1 <- p + geom_smooth(method = "loess") + ggtitle("Standard LOESS")
p2 <- p + geom_smooth(method = "circ_loess", method.args = list(circ_space = 180, span = 0.5)) +
  ggtitle("Circular LOESS, span = 0.5")
p3 <- p + geom_smooth(method = "circ_loess", method.args = list(circ_space = 180, span = 0.2)) +
  ggtitle("Circular LOESS, span = 0.2")
(p1 + p2 + p3)
```

`circ_mean_rad` *Circular mean*

Description

Circular mean

Usage

```
circ_mean_rad(x, na.rm = FALSE)

circ_mean_180(x, na.rm = FALSE)

circ_mean_360(x, na.rm = FALSE)
```

Arguments

`x`	vector of values
`na.rm`	a logical value indicating whether NA values should be removed before the computation proceeds

Value

mean of values in the vector

Functions

- `circ_mean_rad()`: circular mean in 2pi space
- `circ_mean_180()`: circular mean in 180° space (e.g., line orientation)
- `circ_mean_360()`: circular mean in 360° space

Examples

```
x <- runif(1000, -pi, pi)
mean(x)
circ_mean_rad(x)
```

`circ_sd_rad` *Circular standard deviation*

Description

Circular standard deviation

Usage

```
circ_sd_rad(x, na.rm = FALSE)

circ_sd_360(x, na.rm = FALSE)

circ_sd_180(x, na.rm = FALSE)
```

Arguments

`x`	vector of angles
`na.rm`	a logical value indicating whether NA values should be removed before the computation proceeds

Value

standard deviation of values in the vector

Functions

- `circ_sd_rad()`: SD of angles in radians
- `circ_sd_360()`: SD of angles in 360 degree space
- `circ_sd_180()`: SD of angles in 180 degree space

Examples

```
circ_sd_rad(rnorm(50))
circ_sd_180(rnorm(50))
```

`correct_angle_rad` *Get angle value in [-pi, pi] space*

Description

Get angle value in [-pi, pi] space

Usage

```
correct_angle_rad(x)
```

Arguments

`x`	angle

Value

angle in [-pi, pi] space

Examples

```
correct_angle_rad(4 * pi)
```

pad_circ — Pad circular data on both ends

Description

Pad circular data on both ends

Usage

```
pad_circ(
  data,
  circ_var,
  circ_borders = c(-90, 90),
  circ_part = 1/6,
  verbose = FALSE
)
```

Arguments

data	data.table to pad
circ_var	circular variable
circ_borders	range of the circular variable
circ_part	padding proportion
verbose	print extra info

Details

Pads the data by adding a part of the data (default: 1/6th) from one end to another end. Useful to roughly account for circularity when using non-circular methods.

Value

a padded data.table

Examples

```
dt <- data.table(x = runif(1000, -90, 90), y = rnorm(1000))
pad_circ(dt, "x", verbose = TRUE)
```

`Pascucci_et_al_2019_data` *Data from an orientation estimation task*

Description

A dataset with the orientation estimation results from Experiment 2 in Pascucci et al. (2019) available from https://doi.org/10.5281/zenodo.2544946.

Usage

```
Pascucci_et_al_2019_data
```

Format

A data frame with 4400 rows and 5 variables:

observer observer ID

orientation true orientation

reported reported orientation

rt response time

err estimation error

Source

`https://zenodo.org/record/2544946/files/Experiment2_rawdata.csv?download=1`

References

Pascucci, D., Mancuso, G., Santandrea, E., Libera, C. D., Plomp, G., & Chelazzi, L. (2019). Laws of concatenated perception: Vision goes for novelty, decisions for perseverance. PLoS Biology, 17(3). doi:10.1371/journal.pbio.3000144

`remove_cardinal_biases` *Remove cardinal biases*

Description

Remove cardinal biases

Usage

```
remove_cardinal_biases(
  err,
  x,
  space = "180",
  bias_type = "fit",
  plots = "hide",
  poly_deg = 4,
  var_sigma = TRUE,
  var_sigma_poly_deg = 4,
  reassign_at_boundaries = TRUE,
  reassign_range = 2,
  break_points = NULL,
  init_outliers = NULL,
  debug = FALSE,
  do_plots = NULL
)
```

Arguments

`err`	a vector of errors, deviations of response from the true stimuli
`x`	a vector of true stimuli in degrees (see space)
`space`	circular space to use (a string: `180` or `360`)
`bias_type`	bias type to use (`fit`, `card`, `obl`, or `custom`, see details)
`plots`	a string `hide`, `show`, or `return` to hide, show, or return plots (default: `hide`)
`poly_deg`	degree of the fitted polynomials for each bin (default: 4)
`var_sigma`	allow standard deviation (width) of the fitted response distribution to vary as a function of distance to the nearest cardinal (default: True)
`var_sigma_poly_deg`	degree of the fitted polynomials for each bin for the first approximation for the response distribution to select the best fitting model (default: 4)
`reassign_at_boundaries`	select the bin for the observations at the boundaries between bins based on the best-fitting polynomial (default: True)
`reassign_range`	maximum distance to the boundary at which reassignment can occur (default: 2 degrees)
`break_points`	can be used to assign custom break points instead of cardinal/oblique ones with `bias_type` set to `custom` (default: NULL)
`init_outliers`	a vector determining which errors are initially assumed to be outliers (default: NULL)
`debug`	print some extra info (default: False)
`do_plots`	deprecated, use the parameter `plots` instead

Details

If the `bias_type` is set to `fit`, the function computes the cardinal biases in the following way:

1. Create two sets of bins, splitting the stimuli vector into bins centered at cardinal and at oblique directions.
2. For each set of bins, fit a nth-degree polynomial for the responses in each bin, optionally allowing the distribution of responses to vary in width as a function of distance to the nearest cardinal (regardless of whether the bins are centered at the cardinal or at the oblique, the width of the response distribution usually increases as the distance to cardinals increase).
3. Choose the best-fitting model between the one using cardinal and the one using oblique bins.
4. Compute the residuals of the best-fitting model - that's your bias-corrected error - and the biases (see below).

The bias is computed by flipping the sign of errors when the average predicted error is negative, so, that, for example, if on average the responses are shifted clockwise relative to the true values, the trial-by-trial error would count as bias when it is also shifted clockwise.

If `bias_type` is set to `obl` or `card`, only one set of bins is used, centred at cardinal or oblique angles, respectively.

Value

If `plots=='return'`, returns the three plots showing the biases (combined together with `patchwork::wrap_plots()`). Otherwise, returns a list with the following elements:

- is_outlier - 0 for outliers (defined as `±3*pred_sigma` for the model with varying sigma or as `±3\*SD` for the simple model)
- pred predicted error
- be_c error corrected for biases (`be_c = observed error - pred`)
- which_bin the numeric ID of the bin that the stimulus belong to
- bias the bias computed as described above
- bias_typ bias type (cardinal or oblique)
- pred_lin predicted error for a simple linear model for comparison
- pred_sigma predicted SD of the error distribution
- coef_sigma_int, coef_sigma_slope intercept and slope for the sigma prediction

References

- Chetverikov, A., & Jehee, J. F. M. (2023). Motion direction is represented as a bimodal probability distribution in the human visual cortex. Nature Communications, 14(7634). doi:10.1038/s4146702343251w
- van Bergen, R. S., Ma, W. J., Pratte, M. S., & Jehee, J. F. M. (2015). Sensory uncertainty decoded from visual cortex predicts behavior. Nature Neuroscience, 18(12), 1728–1730. doi:10.1038/nn.4150

Examples

```
# Data in orientation domain from Pascucci et al. (2019, PLOS Bio),
# https://doi.org/10.5281/zenodo.2544946

ex_data <- Pascucci_et_al_2019_data[observer == 4, ]
remove_cardinal_biases(ex_data$err, ex_data$orientation, plots = "show")

# Data in motion domain from Bae & Luck (2018, Neuroimage),
# https://osf.io/2h6w9/
ex_data_bae <- Bae_Luck_2018_data[subject_Num == unique(subject_Num)[5], ]
remove_cardinal_biases(ex_data_bae$err, ex_data_bae$TargetDirection,
  space = "360", plots = "show"
)

# Using a stricter initial outlier boundary

remove_cardinal_biases(ex_data_bae$err, ex_data_bae$TargetDirection,
  space = "360", plots = "show",
  init_outliers = abs(ex_data_bae$err) > 60
)

# We can also use just one bin by setting `bias_type` to custom
# and setting the `break_points` at the ends of the range for x

remove_cardinal_biases(ex_data_bae$err, ex_data_bae$TargetDirection,
  space = "360", bias_type = "custom",
  break_points = c(-180, 180), plots = "show",
  reassign_at_boundaries = FALSE, poly_deg = 8,
  init_outliers = abs(ex_data_bae$err) > 60
)
```

---

remove_cardinal_biases_discrete

Remove cardinal biases for data with orientation (color, motion, ...) set in discrete steps

---

Description

Remove cardinal biases for data with orientation (color, motion, ...) set in discrete steps

Usage

```
remove_cardinal_biases_discrete(err, x, space, init_outliers = NULL)
```

Arguments

err	a vector of errors, deviations of response from the true stimuli
x	a vector of true stimuli in degrees (see space)

space	circular space to use (a string: 180 or 360)
init_outliers	a vector determining which errors are initially assumed to be outliers (default: NULL)

Value

returns a data.table with the following columns:

- is_outlier - 0 for outliers (defined as ±3*predicted SD, where SD and mean are computed after excluding initially assumed outliers)
- be_c error corrected for biases (`be_c = observed error - pred`)

`vm_kappa_to_circ_sd` *Conversion between the circular SD and kappa of von Mises*

Description

Conversion between the circular SD and kappa of von Mises

Usage

```
vm_kappa_to_circ_sd(kappa)

vm_kappa_to_circ_sd_deg(kappa)

vm_circ_sd_to_kappa(sd)

vm_circ_sd_deg_to_kappa(sd_deg)
```

Arguments

kappa	von Mises kappa parameter
sd	circular SD of von Mises (radians)
sd_deg	circular SD of von Mises (degrees)

Value

`vm_kappa_to_circ_sd` and `vm_kappa_to_circ_sd_deg` return circular SD (in radians or degrees, respectively) corresponding to a given kappa. `vm_circ_sd_to_kappa` and `vm_circ_sd_deg_to_kappa` return kappa corresponding to a given circular SD (in radians or degrees, respectively).

Functions

- `vm_kappa_to_circ_sd_deg()`: get circular SD (in degrees) from kappa
- `vm_circ_sd_to_kappa()`: get kappa from circular SD (in radians)
- `vm_circ_sd_deg_to_kappa()`: get kappa from circular SD (in degrees)

Examples

```
vm_kappa <- 5
vm_sd <- vm_kappa_to_circ_sd(vm_kappa)

vm_circ_sd_to_kappa(vm_sd)

x <- circular::rvonmises(10000, mu = circular::circular(0), kappa = vm_kappa)

sprintf("Expected SD: %.2f, actual SD: %.2f", vm_sd, circ_sd_rad(x))
```

`weighted_circ_mean` *Weighted circular parameters*

Description

Weighted circular parameters

Usage

```
weighted_circ_mean(x, w, na.rm = FALSE)

weighted_circ_mean2(x, w, na.rm = FALSE)

weighted_circ_sd(x, w, na.rm = FALSE)

weighted_circ_rho(x, w, na.rm = FALSE)
```

Arguments

`x`	vector of values (in radians)
`w`	vector of weights
`na.rm`	a logical value indicating whether NA values should be removed before the computation proceeds

Value

weighted mean of values in the vector

Functions

- `weighted_circ_mean()`: weighted circular mean
- `weighted_circ_mean2()`: an alternative way to compute weighted circular mean (the results are the same)
- `weighted_circ_sd()`: weighted circular SD
- `weighted_circ_rho()`: weighted mean resultant length

Examples

```
x <- rnorm(1000, 0, 0.5)
w <- runif(1000, 0, 1)
weighted.mean(x, w)
weighted_circ_mean(x, w)
```

---

weighted_sem *Weighted standard error of the mean (SEM_w)*

Description

Computes the variance of a weighted mean following the definitions given by Kirchner (2006).

Usage

```
weighted_sem(x, w, na.rm = FALSE)
```

Arguments

x	variable to compute the SEM for
w	weights
na.rm	should NAs be removed

Details

James Kirchner describes two different cases when the weighted variance is computed. The code here implements Case I where "one wants to give more weight to some points than to others, because they are considered to be more important" and "the weights differ but the uncertainties associated with the individual xi are assumed to be the same" (Kirchner, 2006, p. 1). The formula used is:

$$SEM_w = \sqrt{\left(\sum_{i=1}^{N}(w_i x_i^2) - \bar{x}^2\right) \frac{\sum_{i=1}^{N} w_i^2}{1 - \sum_{i=1}^{N} w_i^2}}$$

The expected error is within 5% of the bootstrapped SEM (at larger sample sizes).

Value

weighted standard error of the mean

References

- Kirchner, J. 2006. Data Analysis Toolkit #12: Weighted averages and their uncertainties. https://seismo.berkeley.edu/~kirchner/Toolkits/Toolkit_12.pdf. Retrieved on 04.07.2024.
- Bevington, P. R. 1969. Data Reduction and Error Analysis for the Physical Sciences. McGraw-Hill, 336 pp.

Examples

```
set.seed(1)
n_obs <- 200
w <- runif(n_obs)
w <- w/sum(w)
x <- rnorm(n_obs, sd = 5)
weighted_sem(x, w)
```

Index