Package: ggcircular (via r-universe)

June 4, 2026

Title A 'ggplot2' Extension for Circular and Directional Data

Version 0.1.0

Description Provides a 'ggplot2' grammar for circular, axial and directional data, including rose diagrams, circular densities, mean directions, confidence arcs, theoretical circular distributions and movement data visualizations.

License MIT + file LICENSE

URL https://github.com/AurelienNicosiaULaval/ggcircular, https://aureliennicosiaulaval.github.io/ggcircular/

BugReports https://github.com/AurelienNicosiaULaval/ggcircular/issues

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports dplyr, ggplot2, grid, rlang, scales, stats, tibble, utils, vctrs

Suggests circular, knitr, momentuHMM, posterior, rmarkdown, testthat (>= 3.0.0)

Config/Needs/coverage covr

Config/Needs/website pkgdown

Config/testthat/edition 3

VignetteBuilder knitr

LazyData true

NeedsCompilation no

Author Aurélien Nicosia [aut, cre]

Maintainer Aurélien Nicosia <aurelien.nicosia@mat.ulaval.ca>

Repository https://cran.r-universe.dev

Date/Publication 2026-06-04 14:11:48 UTC

RemoteUrl https://github.com/cran/ggcircular

RemoteRef HEAD

RemoteSha 0b650586361d8e75abcde8c75349976d27597fb8

Contents

angular_difference . . . 3
angular_distance . . . 4
animal_steps . . . 4
as_circular_draws . . . 5
as_step_data . . . 6
augment_circular . . . 7
augment_momentuHMM_angles . . . 7
autoplot_circular . . . 8
autoplot_circular_draws . . . 9
axial_orientations . . . 10
cartesian_to_spherical . . . 10
check_angle . . . 11
circular_mean_ci . . . 11
circular_model_diagnostics . . . 12
circular_residuals . . . 13
circular_sd . . . 14
circular_summary . . . 14
circular_variance . . . 15
compass_to_rad . . . 16
compute_bearing . . . 16
compute_step_length . . . 17
compute_turn_angle . . . 17
coord_circular . . . 18
deg_to_rad . . . 19
estimate_kappa . . . 19
fit_vonmises_mixture . . . 20
geom_circular_density . . . 21
geom_circular_point . . . 23
geom_confidence_arc . . . 24
geom_direction_arrow . . . 25
geom_mean_direction . . . 26
geom_rose . . . 27
hour_to_rad . . . 28
hourly_activity . . . 29
is_angle . . . 30
mean_direction . . . 30
mean_resultant_length . . . 31
mutate_directional_features . . . 32
normalize_angle . . . 33
plot_state_angles . . . 33
rad_to_compass . . . 34

rad_to_deg . . . 35
rad_to_hour . . . 35
rayleigh_test . . . 36
resultant_length . . . 36
scale_x_circular_radians . . . 37
spherical_summary . . . 38
spherical_to_cartesian . . . 39
stat_circular_density . . . 39
stat_circular_test . . . 41
stat_mean_direction . . . 42
stat_rose . . . 43
stat_vonmises . . . 44
stat_vonmises_fit . . . 46
stat_vonmises_mixture . . . 47
summarise_circular_draws . . . 48
theme_circular . . . 49
watson_williams_test . . . 49
wind_directions . . . 50

Index **51**

`angular_difference` *Signed angular difference*

Description

Computes the signed difference `x - y` on a periodic scale. With the default period, values are returned in `[-pi, pi)`.

Usage

```
angular_difference(x, y, period = 2 * pi)
```

Arguments

`x, y`	Numeric vectors of angles.
`period`	Positive numeric period.

Value

A numeric vector following R recycling rules.

See Also

Other angle utilities: `angular_distance()`, `check_angle()`, `compass_to_rad()`, `deg_to_rad()`, `hour_to_rad()`, `is_angle()`, `normalize_angle()`, `rad_to_compass()`, `rad_to_deg()`, `rad_to_hour()`

Examples

```
angular_difference(0, 3 * pi / 2)
```

`angular_distance` *Circular angular distance*

Description

Computes the non-negative angular distance between `x` and `y`. With the default period, values are returned in `[0, pi]`.

Usage

```
angular_distance(x, y, period = 2 * pi)
```

Arguments

`x, y`	Numeric vectors of angles.
`period`	Positive numeric period.

Value

A non-negative numeric vector.

See Also

Other angle utilities: `angular_difference()`, `check_angle()`, `compass_to_rad()`, `deg_to_rad()`, `hour_to_rad()`, `is_angle()`, `normalize_angle()`, `rad_to_compass()`, `rad_to_deg()`, `rad_to_hour()`

Examples

```
angular_distance(0, 3 * pi / 2)
```

`animal_steps` *Simulated animal movement steps*

Description

Simulated tracks for three individuals with derived step length, bearing and turn angle features.

Usage

```
animal_steps
```

Format

A tibble with 600 rows and 8 variables:

id Animal identifier.

time Step index.

x, y Cartesian coordinates.

step_length Euclidean step length.

bearing Movement bearing in radians under the mathematical convention.

turn_angle Signed turn angle in radians.

state Latent movement state label.

Source

Simulated for package examples.

---

as_circular_draws *Convert posterior draws to circular draws*

Description

Converts objects supported by `posterior::as_draws_df()` into a long tibble of normalized angular draws.

Usage

```
as_circular_draws(draws, variables = NULL, period = 2 * pi, origin = 0)
```

Arguments

`draws`	Posterior draws object.
`variables`	Optional variables to keep.
`period`	Angular period.
`origin`	Lower bound of the normalized interval.

Value

A tibble with draw identifiers, `.variable` and `.angle`.

See Also

Other posterior helpers: `autoplot_circular_draws()`, `summarise_circular_draws()`

`as_step_data` *Coerce to step data*

Description

Thin wrapper around `mutate_directional_features()` for pipelines where a more explicit movement-data verb is useful.

Usage

```
as_step_data(
  data,
  x,
  y,
  id = NULL,
  time = NULL,
  angle_convention = c("mathematical", "bearing")
)
```

Arguments

`data`	A data frame.
`x, y`	Coordinate columns.
`id`	Optional individual identifier column.
`time`	Optional time column used for sorting within individual.
`angle_convention`	Angle convention passed to `compute_bearing()`.

Value

A tibble with movement features.

See Also

Other movement helpers: `augment_momentuHMM_angles()`, `compute_bearing()`, `compute_step_length()`, `compute_turn_angle()`, `geom_circular_point()`, `geom_direction_arrow()`, `mutate_directional_features()`, `plot_state_angles()`

augment_circular *Circular model helper generics*

Description

Lightweight generics reserved for future integration with angular regression packages. The default methods fail with an explicit message rather than silently returning incomplete output.

Usage

```
augment_circular(x, ...)

tidy_circular(x, ...)

glance_circular(x, ...)
```

Arguments

x	A model or circular object.
...	Additional arguments passed to methods.

Value

Method-dependent tibble output.

See Also

Other circular model helpers: `circular_model_diagnostics()`, `circular_residuals()`

augment_momentuHMM_angles *Augment momentuHMM fits with angular states*

Description

Extracts an angle column and inferred states from a fitted `momentuHMM` model. The function uses `momentuHMM::viterbi()` by default and adds state probabilities when `momentuHMM::stateProbs()` is available.

Usage

```
augment_momentuHMM_angles(
  object,
  data = NULL,
  angle = NULL,
  state_method = c("viterbi", "stateProbs"),
  ...
)
```

Arguments

object	A fitted momentuHMM object.
data	Optional data frame. If NULL, object$data is used.
angle	Optional name of the angle column.
state_method	State extraction method.
...	Reserved for future methods.

Value

A tibble with .angle, .state and optional state probabilities.

See Also

Other movement helpers: as_step_data(), compute_bearing(), compute_step_length(), compute_turn_angle(), geom_circular_point(), geom_direction_arrow(), mutate_directional_features(), plot_state_angles()

---

autoplot_circular *Autoplot circular data*

Description

Creates a quick diagnostic plot for a numeric vector of circular angles.

Usage

```
autoplot_circular(
  theta,
  bins = 24,
  density = TRUE,
  mean = TRUE,
  axial = FALSE,
  ...
)
```

Arguments

theta	Numeric vector of angles in radians.
bins	Number of rose diagram bins.
density	Should a circular density estimate be added?
mean	Should the mean direction be added?
axial	Should the data be treated as axial, modulo pi?
...	Additional arguments currently ignored.

Value

A ggplot object.

Examples

```
autoplot_circular(wind_directions$direction)
```

`autoplot_circular_draws` *Autoplot circular posterior draws*

Description

Autoplot circular posterior draws

Usage

```
autoplot_circular_draws(
  draws,
  variables = NULL,
  type = c("density", "interval"),
  axial = FALSE,
  ...
)
```

Arguments

`draws`	Circular draws or posterior draws.
`variables`	Optional variables to plot.
`type`	Plot type.
`axial`	Should draws be treated as axial, modulo `pi`?
`...`	Additional arguments passed to `as_circular_draws()` when needed.

Value

A ggplot object.

See Also

Other posterior helpers: `as_circular_draws()`, `summarise_circular_draws()`

axial_orientations *Simulated axial orientations*

Description

Simulated axial orientation data, such as fiber or fault orientations, stored modulo `pi`.

Usage

```
axial_orientations
```

Format

A tibble with 300 rows and 3 variables:

sample Sample identifier.

orientation Axial orientation in radians, modulo `pi`.

group Group label.

Source

Simulated for package examples.

cartesian_to_spherical *Convert Cartesian coordinates to spherical coordinates*

Description

Convert Cartesian coordinates to spherical coordinates

Usage

```
cartesian_to_spherical(
  x,
  y,
  z,
  convention = c("azimuth_colatitude", "azimuth_elevation")
)
```

Arguments

`x`, `y`, `z`	Cartesian coordinates.
`convention`	Output convention for `phi`.

Value

A tibble with `theta`, `phi` and `radius`.

See Also

Other spherical helpers: `spherical_summary()`, `spherical_to_cartesian()`

`check_angle` *Check an angle vector*

Description

Check an angle vector

Usage

```
check_angle(x, allow_na = TRUE)
```

Arguments

`x`	Object to check.
`allow_na`	Should missing values be allowed?

Value

Invisibly returns `x` if the check succeeds.

See Also

Other angle utilities: `angular_difference()`, `angular_distance()`, `compass_to_rad()`, `deg_to_rad()`, `hour_to_rad()`, `is_angle()`, `normalize_angle()`, `rad_to_compass()`, `rad_to_deg()`, `rad_to_hour()`

`circular_mean_ci` *Confidence interval for a circular mean*

Description

Computes an approximate confidence interval for a circular mean. The large sample method uses a normal approximation on the mean direction, while the bootstrap method resamples angles and forms an interval from angular deviations around the sample mean. These intervals are exploratory; they are not reliable when the mean resultant length is close to zero and the mean direction is weakly identified.

Usage

```
circular_mean_ci(
  x,
  level = 0.95,
  method = c("large_sample", "bootstrap"),
  R = 999,
  axial = FALSE,
  na.rm = TRUE,
  seed = NULL
)
```

Arguments

x	Numeric vector of angles in radians.
level	Confidence level.
method	Interval method.
R	Number of bootstrap resamples.
axial	Should data be treated as axial, modulo pi?
na.rm	Should missing values be removed?
seed	Optional random seed for the bootstrap.

Value

A tibble with mean, lower, upper, level, method, n and Rbar.

See Also

Other circular summaries: circular_sd(), circular_summary(), circular_variance(), estimate_kappa(), mean_direction(), mean_resultant_length(), resultant_length()

---

circular_model_diagnostics *Circular model diagnostics*

Description

Summarizes circular residual diagnostics for supported angular model objects.

Usage

```
circular_model_diagnostics(object, data = NULL, ...)
```

Arguments

object	A supported angular model object.
data	Optional data frame to bind to the diagnostic columns.
...	Reserved for future methods.

Value

A tibble with residual mean direction, resultant length, circular variance and maximum absolute circular residual.

See Also

Other circular model helpers: augment_circular(), circular_residuals()

---

circular_residuals *Circular residuals for angular models*

Description

Extracts observed angles, fitted angles and signed circular residuals from supported angular model objects. The function currently supports objects produced by the optional CircularRegression package when their fitted values are stored in a mui component.

Usage

```
circular_residuals(object, data = NULL, ...)
```

Arguments

object	A supported angular model object.
data	Optional data frame to bind to the diagnostic columns.
...	Reserved for future methods.

Value

A tibble with .observed, .fitted, .resid, .abs_resid, .index and .model_class.

See Also

Other circular model helpers: augment_circular(), circular_model_diagnostics()

Examples

```
fit <- structure(
  list(y = c(0, 0.2, 0.4), mui = c(0.05, 0.15, 0.5)),
  class = "angular"
)
circular_residuals(fit)
```

circular_sd *Circular standard deviation*

Description

Uses the common descriptive statistic `sqrt(-2 * log(Rbar))`.

Usage

```
circular_sd(x, axial = FALSE, na.rm = TRUE)
```

Arguments

`x`	Numeric vector of angles in radians.
`axial`	Should the data be treated as axial, modulo `pi`?
`na.rm`	Should missing values be removed?

Value

Circular standard deviation in radians.

See Also

Other circular summaries: `circular_mean_ci()`, `circular_summary()`, `circular_variance()`, `estimate_kappa()`, `mean_direction()`, `mean_resultant_length()`, `resultant_length()`

circular_summary *Summarize circular data*

Description

Computes grouped circular summaries for an angle column. Existing dplyr groups are respected, and additional grouping variables can be supplied in `...`.

Usage

```
circular_summary(data, angle, ..., axial = FALSE, na.rm = TRUE)
```

Arguments

`data`	A data frame or tibble.
`angle`	Angle column, in radians.
`...`	Optional grouping variables.
`axial`	Should the data be treated as axial, modulo `pi`?
`na.rm`	Should missing values be removed?

Value

A tibble with columns n, mean, R, Rbar, variance, sd and kappa. The returned object also has class ggcircular_summary.

See Also

Other circular summaries: circular_mean_ci(), circular_sd(), circular_variance(), estimate_kappa(), mean_direction(), mean_resultant_length(), resultant_length()

Examples

```
tibble::tibble(group = c("a", "a", "b"), theta = c(0, pi / 2, pi)) |>
  dplyr::group_by(group) |>
  circular_summary(theta)
```

---

circular_variance *Circular variance*

Description

Circular variance

Usage

```
circular_variance(x, axial = FALSE, na.rm = TRUE)
```

Arguments

x	Numeric vector of angles in radians.
axial	Should the data be treated as axial, modulo pi?
na.rm	Should missing values be removed?

Value

The circular variance 1 - Rbar.

See Also

Other circular summaries: circular_mean_ci(), circular_sd(), circular_summary(), estimate_kappa(), mean_direction(), mean_resultant_length(), resultant_length()

compass_to_rad *Convert compass labels to radians*

Description

Converts the eight standard compass labels `N`, `NE`, `E`, `SE`, `S`, `SW`, `W` and `NW` to bearing angles in radians, where zero is north and angles increase clockwise.

Usage

```
compass_to_rad(x)
```

Arguments

`x`	Character vector of compass labels.

Value

Numeric vector of bearing angles in radians.

See Also

Other angle utilities: `angular_difference()`, `angular_distance()`, `check_angle()`, `deg_to_rad()`, `hour_to_rad()`, `is_angle()`, `normalize_angle()`, `rad_to_compass()`, `rad_to_deg()`, `rad_to_hour()`

Examples

```
compass_to_rad(c("N", "E", "S", "W"))
```

compute_bearing *Compute bearings*

Description

Compute bearings

Usage

```
compute_bearing(x, y, angle_convention = c("mathematical", "bearing"))
```

Arguments

`x, y`	Numeric coordinate vectors.
`angle_convention`	Angle convention. `"mathematical"` means zero is east and angles increase counterclockwise. `"bearing"` means zero is north and angles increase clockwise.

Value

Numeric vector of bearings in radians. The first value is NA.

See Also

Other movement helpers: as_step_data(), augment_momentuHMM_angles(), compute_step_length(), compute_turn_angle(), geom_circular_point(), geom_direction_arrow(), mutate_directional_features(), plot_state_angles()

---

compute_step_length *Compute step lengths*

Description

Compute step lengths

Usage

```
compute_step_length(x, y)
```

Arguments

x, y	Numeric coordinate vectors.

Value

Numeric vector of Euclidean step lengths. The first value is NA.

See Also

Other movement helpers: as_step_data(), augment_momentuHMM_angles(), compute_bearing(), compute_turn_angle(), geom_circular_point(), geom_direction_arrow(), mutate_directional_features(), plot_state_angles()

---

compute_turn_angle *Compute turn angles*

Description

Compute turn angles

Usage

```
compute_turn_angle(bearing, period = 2 * pi)
```

Arguments

bearing	Numeric vector of bearings in radians.
period	Angular period.

Value

Numeric vector of signed turn angles. The first value is NA.

See Also

Other movement helpers: as_step_data(), augment_momentuHMM_angles(), compute_bearing(), compute_step_length(), geom_circular_point(), geom_direction_arrow(), mutate_directional_features(), plot_state_angles()

---

coord_circular *Circular coordinate system*

Description

Convenience wrapper around ggplot2::coord_polar() with arguments expressed in circular-data language.

Usage

```
coord_circular(
  zero = c("east", "north", "west", "south"),
  direction = c("counterclockwise", "clockwise"),
  start = NULL,
  clip = "on"
)
```

Arguments

zero	Direction corresponding to angle zero.
direction	Direction in which angles increase.
start	Optional start offset in radians. If supplied, it overrides zero.
clip	Should drawing be clipped to the plot panel?

Details

zero = "east" and direction = "counterclockwise" gives the usual mathematical convention: zero points east and positive angles rotate toward north. zero = "north" and direction = "clockwise" gives the usual bearing convention used for compass directions.

Value

A ggplot2 coordinate object.

See Also

Other circular scales: scale_x_circular_radians()

Examples

```
coord_circular()
coord_circular(zero = "north", direction = "clockwise")
```

---

deg_to_rad *Convert degrees to radians*

Description

Convert degrees to radians

Usage

```
deg_to_rad(x)
```

Arguments

x	Numeric vector in degrees.

Value

Numeric vector in radians.

See Also

Other angle utilities: angular_difference(), angular_distance(), check_angle(), compass_to_rad(), hour_to_rad(), is_angle(), normalize_angle(), rad_to_compass(), rad_to_deg(), rad_to_hour()

---

estimate_kappa *Estimate von Mises concentration*

Description

Estimates the von Mises concentration parameter from the mean resultant length using the standard piecewise approximation described by Fisher (1993). This is a descriptive approximation and does not apply small-sample bias corrections or uncertainty quantification.

Usage

```
estimate_kappa(x, axial = FALSE, na.rm = TRUE)
```

Arguments

x	Numeric vector of angles in radians.
axial	Should the data be treated as axial, modulo pi?
na.rm	Should missing values be removed?

Value

Estimated concentration parameter kappa.

References

Fisher, N. I. (1993). *Statistical Analysis of Circular Data*. Cambridge University Press.

See Also

Other circular summaries: circular_mean_ci(), circular_sd(), circular_summary(), circular_variance(), mean_direction(), mean_resultant_length(), resultant_length()

---

fit_vonmises_mixture *Fit a mixture of von Mises distributions*

Description

Fits a finite mixture of von Mises components using an expectation maximization algorithm. For axial data, the algorithm fits doubled angles and returns component means on the original modulo-pi scale.

Usage

```
fit_vonmises_mixture(
  x,
  k = 2,
  weights = NULL,
  axial = FALSE,
  init = c("kmeans", "spaced"),
  nstart = 1,
  init_params = NULL,
  kappa_max = 10000,
  min_component_weight = 1e-08,
  max_iter = 200,
  tol = 1e-08,
  na.rm = TRUE,
  seed = NULL
)
```

Arguments

x	Numeric vector of angles in radians.
k	Number of mixture components.
weights	Optional non-negative observation weights.
axial	Should data be treated as axial, modulo pi?
init	Initialization method, either "kmeans" or "spaced".
nstart	Number of EM starts. The fit with the largest log-likelihood is retained.
init_params	Optional list or data frame with initial proportions, mu and kappa values.
kappa_max	Maximum fitted concentration. This caps nearly degenerate components.
min_component_weight	Minimum relative component weight before a component is reinitialized.
max_iter	Maximum number of EM iterations.
tol	Convergence tolerance on the log-likelihood.
na.rm	Should missing values be removed?
seed	Optional random seed used for initialization.

Value

An object of class ggcircular_vonmises_mixture.

See Also

Other circular distributions: stat_vonmises(), stat_vonmises_fit(), stat_vonmises_mixture()

Examples

```
fit <- fit_vonmises_mixture(wind_directions$direction, k = 2)
tidy_circular(fit)
```

---

geom_circular_density *Circular density layer*

Description

Draws a circular density estimate as a line. This is a convenience wrapper around stat_circular_density().

Usage

```
geom_circular_density(
  mapping = NULL,
  data = NULL,
  stat = "circular_density",
  position = "identity",
  ...,
  method = c("kernel", "vonmises"),
  bw = NULL,
  adjust = 1,
  n = 512,
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `position`, `show.legend`, `inherit.aes`	Standard ggplot2 layer arguments.
`stat`	Statistical transformation, usually `"circular_density"`.
`...`	Additional arguments passed to the layer.
`method`	Density method. Currently `"kernel"` and `"vonmises"` both use a von Mises kernel estimator.
`bw`	Optional circular bandwidth. It is interpreted as `1 / sqrt(kappa)`.
`adjust`	Multiplicative adjustment applied to `bw` or to the automatic bandwidth scale.
`n`	Number of grid points.
`axial`	Should the data be treated as axial, modulo `pi`?
`na.rm`	Should missing values be silently removed?

Value

A ggplot2 layer.

See Also

Other circular density layers: `stat_circular_density()`

Examples

```
ggplot2::ggplot(wind_directions, ggplot2::aes(x = direction)) +
  geom_circular_density()
```

`geom_circular_point` *Circular point and rug helpers*

Description

Convenience layers for plotting angular observations at a fixed radius.

Usage

```
geom_circular_point(
  mapping = NULL,
  data = NULL,
  ...,
  radius = 1,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)

geom_circular_rug(
  mapping = NULL,
  data = NULL,
  ...,
  radius = 1,
  rug_length = 0.05,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `show.legend`, `inherit.aes`	Standard ggplot2 layer arguments.
`...`	Additional arguments passed to `ggplot2::geom_segment()`.
`radius`	Radius at which points or rugs are drawn.
`na.rm`	Should missing values be silently removed?
`rug_length`	Radial length of rug marks.

Value

A ggplot2 layer.

See Also

Other movement helpers: as_step_data(), augment_momentuHMM_angles(), compute_bearing(), compute_step_length(), compute_turn_angle(), geom_direction_arrow(), mutate_directional_features(), plot_state_angles()

---

geom_confidence_arc *Circular confidence arc*

Description

Draws angular intervals as arcs at a fixed radius. Intervals crossing zero are split into two path segments.

Usage

```
geom_confidence_arc(
  mapping = NULL,
  data = NULL,
  ...,
  radius = 1,
  n = 200,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)

geom_circular_interval(
  mapping = NULL,
  data = NULL,
  ...,
  radius = 1,
  n = 200,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, show.legend, inherit.aes	Standard ggplot2 layer arguments.
...	Additional arguments passed to the path geometry.
radius	Default radius used when no radius or y aesthetic is supplied.
n	Number of points used to discretize each interval.
na.rm	Should missing interval endpoints be silently removed?

Value

A ggplot2 layer.

Examples

```
tibble::tibble(lower = 5.5, upper = 0.5) |>
  ggplot2::ggplot(ggplot2::aes(xmin = lower, xmax = upper)) +
  geom_confidence_arc()
```

---

geom_direction_arrow *Direction arrows*

Description

Draws directional arrows from Cartesian coordinates and an angle.

Usage

```
geom_direction_arrow(
  mapping = NULL,
  data = NULL,
  ...,
  length = 1,
  arrow_length = grid::unit(0.15, "cm"),
  angle_convention = c("mathematical", "bearing"),
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `show.legend`, `inherit.aes`	Standard ggplot2 layer arguments.
`...`	Additional arguments passed to `ggplot2::geom_segment()`.
`length`	Default arrow length when no `length` aesthetic is supplied.
`arrow_length`	Grid unit controlling the arrow head length.
`angle_convention`	Angle convention. `"mathematical"` means zero is east and angles increase counterclockwise. `"bearing"` means zero is north and angles increase clockwise.
`na.rm`	Should missing values be silently removed?

Value

A ggplot2 layer.

See Also

Other movement helpers: as_step_data(), augment_momentuHMM_angles(), compute_bearing(), compute_step_length(), compute_turn_angle(), geom_circular_point(), mutate_directional_features(), plot_state_angles()

Examples

```
ggplot2::ggplot(animal_steps, ggplot2::aes(x = x, y = y, angle = bearing)) +
  geom_direction_arrow()
```

---

geom_mean_direction *Mean direction layer*

Description

Draws a radial segment at the circular mean direction. The segment length can be fixed or proportional to the mean resultant length.

Usage

```
geom_mean_direction(
  mapping = NULL,
  data = NULL,
  stat = "mean_direction",
  position = "identity",
  ...,
  length = c("resultant", "fixed"),
  radius = NULL,
  conf.int = FALSE,
  level = 0.95,
  axial = FALSE,
  arrow = TRUE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
stat	Statistical transformation, usually "mean_direction".
...	Additional arguments passed to the layer.
length	Should the displayed segment length be proportional to the mean resultant length ("resultant") or fixed ("fixed")?
radius	Optional maximum displayed radius.

conf.int	Should approximate confidence limits be computed?
level	Confidence level used when conf.int = TRUE.
axial	Should the data be treated as axial, modulo pi?
arrow	Should a small arrow head be drawn?
na.rm	Should missing values be silently removed?

Value

A ggplot2 layer.

See Also

Other mean direction layers: stat_mean_direction()

Examples

```
ggplot2::ggplot(wind_directions, ggplot2::aes(x = direction)) +
  geom_rose(bins = 16) +
  geom_mean_direction()
```

---

geom_rose *Rose diagram layer*

Description

geom_rose() is a convenience wrapper around stat_rose() using a bar geometry. It is designed to be used with coord_circular() or ggplot2::coord_polar().

Usage

```
geom_rose(
  mapping = NULL,
  data = NULL,
  stat = "rose",
  position = "identity",
  ...,
  bins = 30,
  binwidth = NULL,
  boundary = 0,
  closed = TRUE,
  area = FALSE,
  normalize = c("count", "density", "proportion"),
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
stat	Statistical transformation, usually "rose".
...	Additional arguments passed to the layer.
bins	Number of bins over the circular period.
binwidth	Optional bin width in radians. If supplied, bins is ignored after the number of bins is inferred from the period.
boundary	Lower boundary for the first bin.
closed	Included for API compatibility. Values on the upper period boundary are wrapped into the first bin.
area	If TRUE, radial heights are square-root transformed so that visual area is closer to the selected frequency scale.
normalize	Which scale should be used for the computed radial y value: counts, densities or proportions.
axial	Should angles be treated as axial, modulo pi?
na.rm	Should missing values be silently removed?

Value

A ggplot2 layer.

See Also

Other rose diagram layers: stat_rose()

Examples

```
ggplot2::ggplot(wind_directions, ggplot2::aes(x = direction)) +
  geom_rose(bins = 16) +
  coord_circular()
```

---

hour_to_rad *Convert hours to radians*

Description

Convert hours to radians

Usage

```
hour_to_rad(x)
```

Arguments

x	Numeric vector in hours on a 24-hour clock.

Value

Numeric vector in radians.

See Also

Other angle utilities: `angular_difference()`, `angular_distance()`, `check_angle()`, `compass_to_rad()`, `deg_to_rad()`, `is_angle()`, `normalize_angle()`, `rad_to_compass()`, `rad_to_deg()`, `rad_to_hour()`

`hourly_activity` *Simulated hourly activity*

Description

A reproducible simulated dataset of hourly activity converted to circular angles.

Usage

```
hourly_activity
```

Format

A tibble with 240 rows and 5 variables:

id Individual identifier.

hour Hour of day.

angle Hour converted to radians.

activity Activity level.

group Group label.

Source

Simulated for package examples.

is_angle *Test whether an object can represent angles*

Description

Test whether an object can represent angles

Usage

```
is_angle(x)
```

Arguments

x	Object to test.

Value

TRUE when x is numeric and contains only finite values or NA.

See Also

Other angle utilities: angular_difference(), angular_distance(), check_angle(), compass_to_rad(), deg_to_rad(), hour_to_rad(), normalize_angle(), rad_to_compass(), rad_to_deg(), rad_to_hour()

mean_direction *Circular mean direction*

Description

Computes the sample mean direction. For axial data, angles are doubled before computing the mean and the result is transformed back to the axial scale.

Usage

```
mean_direction(x, axial = FALSE, na.rm = TRUE)
```

Arguments

x	Numeric vector of angles in radians.
axial	Should the data be treated as axial, modulo pi?
na.rm	Should missing values be removed?

Value

A single angle in radians, or NA_real_ when the mean is undefined.

See Also

Other circular summaries: circular_mean_ci(), circular_sd(), circular_summary(), circular_variance(), estimate_kappa(), mean_resultant_length(), resultant_length()

Examples

```
mean_direction(c(0, pi / 4, pi / 2))
```

---

mean_resultant_length *Mean resultant length*

Description

Mean resultant length

Usage

```
mean_resultant_length(x, axial = FALSE, na.rm = TRUE)
```

Arguments

x	Numeric vector of angles in radians.
axial	Should the data be treated as axial, modulo pi?
na.rm	Should missing values be removed?

Value

The mean resultant length Rbar, between 0 and 1 when defined.

See Also

Other circular summaries: circular_mean_ci(), circular_sd(), circular_summary(), circular_variance(), estimate_kappa(), mean_direction(), resultant_length()

`mutate_directional_features` *Add directional movement features*

Description

Computes step length, bearing and turn angle from track coordinates. When `id` and `time` are supplied, records are sorted by individual and time before features are computed.

Usage

```
mutate_directional_features(
  data,
  x,
  y,
  id = NULL,
  time = NULL,
  angle_convention = c("mathematical", "bearing")
)
```

Arguments

`data`	A data frame.
`x, y`	Coordinate columns.
`id`	Optional individual identifier column.
`time`	Optional time column used for sorting within individual.
`angle_convention`	Angle convention passed to `compute_bearing()`.

Value

A tibble with added `step_length`, `bearing` and `turn_angle`.

See Also

Other movement helpers: `as_step_data()`, `augment_momentuHMM_angles()`, `compute_bearing()`, `compute_step_length()`, `compute_turn_angle()`, `geom_circular_point()`, `geom_direction_arrow()`, `plot_state_angles()`

Examples

```
tibble::tibble(id = 1, time = 1:3, x = 0:2, y = 0) |>
  mutate_directional_features(x = x, y = y, id = id, time = time)
```

normalize_angle *Normalize angles to a periodic interval*

Description

`normalize_angle()` maps numeric angles to `[origin, origin + period)`. The default period is `2 * pi`, which is appropriate for directional circular data measured in radians.

Usage

```
normalize_angle(x, period = 2 * pi, origin = 0)
```

Arguments

`x`	Numeric vector of angles.
`period`	Positive numeric period. Use `2 * pi` for directional data and `pi` for axial data.
`origin`	Lower bound of the target interval.

Value

A numeric vector with the same length as `x`.

See Also

Other angle utilities: `angular_difference()`, `angular_distance()`, `check_angle()`, `compass_to_rad()`, `deg_to_rad()`, `hour_to_rad()`, `is_angle()`, `rad_to_compass()`, `rad_to_deg()`, `rad_to_hour()`

Examples

```
normalize_angle(c(-pi, 0, 3 * pi))
```

plot_state_angles *Plot angular distributions by state*

Description

Convenience function for visualizing angles by observed or inferred states.

Usage

```
plot_state_angles(
  data,
  angle,
  state,
  type = c("rose", "density", "mean"),
  bins = 24,
  axial = FALSE
)
```

Arguments

data	A data frame.
angle	Angle column.
state	State or group column.
type	Plot type.
bins	Number of bins for rose diagrams.
axial	Should data be treated as axial, modulo pi?

Value

A ggplot object.

See Also

Other movement helpers: as_step_data(), augment_momentuHMM_angles(), compute_bearing(), compute_step_length(), compute_turn_angle(), geom_circular_point(), geom_direction_arrow(), mutate_directional_features()

---

rad_to_compass *Convert radians to compass labels*

Description

Converts angles to the nearest label among labels. Angles are interpreted as bearings by default: zero is north and angles increase clockwise.

Usage

```
rad_to_compass(x, labels = c("N", "NE", "E", "SE", "S", "SW", "W", "NW"))
```

Arguments

x	Numeric vector of angles in radians.
labels	Character vector of equally spaced labels.

Value

Character vector of labels.

See Also

Other angle utilities: angular_difference(), angular_distance(), check_angle(), compass_to_rad(), deg_to_rad(), hour_to_rad(), is_angle(), normalize_angle(), rad_to_deg(), rad_to_hour()

Examples

```
rad_to_compass(c(0, pi / 2, pi))
```

rad_to_deg *Convert radians to degrees*

Description

Convert radians to degrees

Usage

```
rad_to_deg(x)
```

Arguments

x	Numeric vector in radians.

Value

Numeric vector in degrees.

See Also

Other angle utilities: angular_difference(), angular_distance(), check_angle(), compass_to_rad(), deg_to_rad(), hour_to_rad(), is_angle(), normalize_angle(), rad_to_compass(), rad_to_hour()

rad_to_hour *Convert radians to hours*

Description

Convert radians to hours

Usage

```
rad_to_hour(x)
```

Arguments

x	Numeric vector in radians.

Value

Numeric vector in hours on a 24-hour clock.

See Also

Other angle utilities: angular_difference(), angular_distance(), check_angle(), compass_to_rad(), deg_to_rad(), hour_to_rad(), is_angle(), normalize_angle(), rad_to_compass(), rad_to_deg()

rayleigh_test *Rayleigh test for circular uniformity*

Description

Performs the one-sample Rayleigh test for non-uniformity. The test is most sensitive to unimodal departures from circular uniformity. The returned object follows the base `htest` structure.

Usage

```
rayleigh_test(x, axial = FALSE, na.rm = TRUE)
```

Arguments

`x`	Numeric vector of angles in radians.
`axial`	Should data be treated as axial, modulo `pi`?
`na.rm`	Should missing values be removed?

Value

An object of class `htest`.

See Also

Other circular tests: `stat_circular_test()`, `watson_williams_test()`

resultant_length *Resultant length*

Description

Resultant length

Usage

```
resultant_length(x, axial = FALSE, na.rm = TRUE)
```

Arguments

`x`	Numeric vector of angles in radians.
`axial`	Should the data be treated as axial, modulo `pi`?
`na.rm`	Should missing values be removed?

Value

The sample resultant length `R`.

See Also

Other circular summaries: circular_mean_ci(), circular_sd(), circular_summary(), circular_variance(), estimate_kappa(), mean_direction(), mean_resultant_length()

---

scale_x_circular_radians *Circular x scales*

Description

These scales label angular x axes in radians, degrees, hours or compass directions.

Usage

```
scale_x_circular_radians(
  breaks = ggplot2::waiver(),
  labels = ggplot2::waiver(),
  limits = c(0, 2 * pi),
  ...
)

scale_x_circular_degrees(
  breaks = ggplot2::waiver(),
  labels = ggplot2::waiver(),
  limits = c(0, 2 * pi),
  ...
)

scale_x_circular_hours(
  breaks = ggplot2::waiver(),
  labels = ggplot2::waiver(),
  limits = c(0, 2 * pi),
  ...
)

scale_x_circular_compass(
  breaks = ggplot2::waiver(),
  labels = ggplot2::waiver(),
  limits = c(0, 2 * pi),
  ...
)
```

Arguments

breaks	Break positions in radians.
labels	Break labels.
limits	Scale limits in radians.
...	Additional arguments passed to ggplot2::scale_x_continuous().

Value

A ggplot2 scale.

See Also

Other circular scales: `coord_circular()`

Examples

```
scale_x_circular_radians()
```

`spherical_summary` *Summarize spherical directions*

Description

Computes the mean direction vector and mean spherical coordinates.

Usage

```
spherical_summary(
  theta,
  phi,
  weights = NULL,
  convention = c("azimuth_colatitude", "azimuth_elevation"),
  na.rm = TRUE
)
```

Arguments

`theta`	Azimuth angle in radians.
`phi`	Colatitude or elevation angle in radians.
`weights`	Optional non-negative weights.
`convention`	Interpretation of `phi`.
`na.rm`	Should missing values be removed?

Value

A tibble with sample size, mean spherical coordinates and resultant length.

See Also

Other spherical helpers: `cartesian_to_spherical()`, `spherical_to_cartesian()`

spherical_to_cartesian *Convert spherical coordinates to Cartesian coordinates*

Description

Convert spherical coordinates to Cartesian coordinates

Usage

```
spherical_to_cartesian(
  theta,
  phi,
  radius = 1,
  convention = c("azimuth_colatitude", "azimuth_elevation")
)
```

Arguments

theta	Azimuth angle in radians.
phi	Colatitude or elevation angle in radians.
radius	Radius.
convention	Interpretation of phi.

Value

A tibble with x, y and z.

See Also

Other spherical helpers: cartesian_to_spherical(), spherical_summary()

stat_circular_density *Circular density statistic*

Description

Estimates a smooth circular density using a von Mises kernel. The density wraps around the origin, avoiding the boundary artifacts of a linear kernel density estimate. When bw is not supplied, the concentration is chosen from a simple resultant-length heuristic; it should be treated as an exploratory smoothing choice rather than an inferential bandwidth selector.

Usage

```
stat_circular_density(
  mapping = NULL,
  data = NULL,
  geom = "line",
  position = "identity",
  ...,
  method = c("kernel", "vonmises"),
  bw = NULL,
  adjust = 1,
  n = 512,
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, geom, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
...	Additional arguments passed to the layer.
method	Density method. Currently "kernel" and "vonmises" both use a von Mises kernel estimator.
bw	Optional circular bandwidth. It is interpreted as 1 / sqrt(kappa).
adjust	Multiplicative adjustment applied to bw or to the automatic bandwidth scale.
n	Number of grid points.
axial	Should the data be treated as axial, modulo pi?
na.rm	Should missing values be silently removed?

Value

A ggplot2 layer. Computed variables are x, density, scaled, count, n, bw and kappa.

See Also

Other circular density layers: geom_circular_density()

Examples

```
ggplot2::ggplot(wind_directions, ggplot2::aes(x = direction)) +
  stat_circular_density()
```

`stat_circular_test` *Annotate circular tests*

Description

Adds a text annotation with the p-value from a Rayleigh or Watson-Williams test.

Usage

```
stat_circular_test(
  mapping = NULL,
  data = NULL,
  geom = "text",
  position = "identity",
  ...,
  test = c("rayleigh", "watson_williams"),
  x = 0,
  y = 1,
  digits = 3,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `geom`, `position`, `show.legend`, `inherit.aes`	Standard ggplot2 layer arguments.
`...`	Additional arguments passed to the text geom.
`test`	Test to compute.
`x`, `y`	Text position.
`digits`	Number of digits used for p-value formatting.
`na.rm`	Should missing values be removed?

Value

A ggplot2 layer.

See Also

Other circular tests: `rayleigh_test()`, `watson_williams_test()`

`stat_mean_direction` *Mean direction statistic*

Description

Computes one mean direction per group, with resultant length and an optional approximate confidence arc.

Usage

```
stat_mean_direction(
  mapping = NULL,
  data = NULL,
  geom = "segment",
  position = "identity",
  ...,
  length = c("resultant", "fixed"),
  radius = NULL,
  conf.int = FALSE,
  level = 0.95,
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `geom`, `position`, `show.legend`, `inherit.aes`	Standard ggplot2 layer arguments.
`...`	Additional arguments passed to the layer.
`length`	Should the displayed segment length be proportional to the mean resultant length (`"resultant"`) or fixed (`"fixed"`)?
`radius`	Optional maximum displayed radius.
`conf.int`	Should approximate confidence limits be computed?
`level`	Confidence level used when `conf.int = TRUE`.
`axial`	Should the data be treated as axial, modulo `pi`?
`na.rm`	Should missing values be silently removed?

Value

A ggplot2 layer. Computed variables include `x`, `xend`, `y`, `yend`, `mean`, `R`, `Rbar`, `n`, `kappa`, `ci_low` and `ci_high`.

See Also

Other mean direction layers: geom_mean_direction()

Examples

```
ggplot2::ggplot(wind_directions, ggplot2::aes(x = direction)) +
  stat_mean_direction()
```

---

stat_rose *Rose diagram statistic*

Description

Bins circular angles over a full period and returns counts, densities and proportions for rose diagrams.

Usage

```
stat_rose(
  mapping = NULL,
  data = NULL,
  geom = "col",
  position = "identity",
  ...,
  bins = 30,
  binwidth = NULL,
  boundary = 0,
  closed = TRUE,
  area = FALSE,
  normalize = c("count", "density", "proportion"),
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, geom, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
...	Additional arguments passed to the layer.
bins	Number of bins over the circular period.
binwidth	Optional bin width in radians. If supplied, bins is ignored after the number of bins is inferred from the period.
boundary	Lower boundary for the first bin.

closed	Included for API compatibility. Values on the upper period boundary are wrapped into the first bin.
area	If TRUE, radial heights are square-root transformed so that visual area is closer to the selected frequency scale.
normalize	Which scale should be used for the computed radial y value: counts, densities or proportions.
axial	Should angles be treated as axial, modulo pi?
na.rm	Should missing values be silently removed?

Value

A ggplot2 layer. Computed variables are xmin, xmax, x, count, density, proportion, width and y.

See Also

Other rose diagram layers: geom_rose()

Examples

```
ggplot2::ggplot(wind_directions, ggplot2::aes(x = direction)) +
  stat_rose(bins = 16) +
  coord_circular()
```

---

stat_vonmises *Theoretical von Mises density*

Description

Adds a theoretical von Mises density to a circular plot.

Usage

```
stat_vonmises(
  mapping = NULL,
  data = NULL,
  geom = "line",
  position = "identity",
  ...,
  mu = 0,
  kappa = 1,
  n = 512,
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = FALSE
)
```

```
stat_wrapped_normal(
  mapping = NULL,
  data = NULL,
  geom = "line",
  position = "identity",
  ...,
  mu = 0,
  sigma = 1,
  terms = 5,
  n = 512,
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = FALSE
)

stat_uniform_circular(
  mapping = NULL,
  data = NULL,
  geom = "line",
  position = "identity",
  ...,
  n = 512,
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = FALSE
)
```

Arguments

mapping, data, geom, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
...	Additional arguments passed to the layer.
mu	Mean direction in radians.
kappa	Non-negative concentration parameter.
n	Number of grid points.
axial	Should the density be drawn over an axial period of pi?
na.rm	Included for ggplot2 layer compatibility.
sigma	Standard deviation of the wrapped normal distribution.
terms	Number of wrapping terms on each side of the origin.

Value

A ggplot2 layer.

See Also

Other circular distributions: `fit_vonmises_mixture()`, `stat_vonmises_fit()`, `stat_vonmises_mixture()`

Examples

```
ggplot2::ggplot(wind_directions, ggplot2::aes(x = direction)) +
  geom_rose(ggplot2::aes(y = ggplot2::after_stat(density))) +
  stat_vonmises(mu = pi / 2, kappa = 3)
```

---

stat_vonmises_fit *Fitted von Mises density*

Description

Estimates `mu` with `mean_direction()` and `kappa` with `estimate_kappa()`, then draws the fitted von Mises density.

Usage

```
stat_vonmises_fit(
  mapping = NULL,
  data = NULL,
  geom = "line",
  position = "identity",
  ...,
  n = 512,
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `geom`, `position`, `show.legend`, `inherit.aes`	Standard ggplot2 layer arguments.
`...`	Additional arguments passed to the layer.
`n`	Number of grid points.
`axial`	Should the data be treated as axial, modulo `pi`?
`na.rm`	Should missing values be silently removed?

Value

A ggplot2 layer.

See Also

Other circular distributions: `fit_vonmises_mixture()`, `stat_vonmises()`, `stat_vonmises_mixture()`

Examples

```
ggplot2::ggplot(wind_directions, ggplot2::aes(x = direction)) +
  geom_rose(ggplot2::aes(y = ggplot2::after_stat(density))) +
  stat_vonmises_fit()
```

---

stat_vonmises_mixture *Von Mises mixture density layer*

Description

Fits or draws a mixture of von Mises densities.

Usage

```
stat_vonmises_mixture(
  mapping = NULL,
  data = NULL,
  geom = "line",
  position = "identity",
  ...,
  fit = NULL,
  k = 2,
  nstart = 1,
  seed = NULL,
  kappa_max = 10000,
  n = 512,
  axial = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `geom`, `position`, `show.legend`, `inherit.aes`	Standard ggplot2 layer arguments.
`...`	Additional arguments passed to the layer.
`fit`	Optional `ggcircular_vonmises_mixture` object. If `NULL`, the mixture is fitted to the layer's `x` aesthetic.
`k`	Number of components when fitting inside the statistic.
`nstart`	Number of EM starts when fitting inside the statistic.
`seed`	Optional random seed when fitting inside the statistic.

kappa_max	Maximum fitted concentration when fitting inside the statistic.
n	Number of grid points.
axial	Should data be treated as axial, modulo pi?
na.rm	Should missing values be removed before fitting?

Value

A ggplot2 layer.

See Also

Other circular distributions: fit_vonmises_mixture(), stat_vonmises(), stat_vonmises_fit()

---

summarise_circular_draws

Summarize circular posterior draws

Description

Summarize circular posterior draws

Usage

```
summarise_circular_draws(
  draws,
  variables = NULL,
  level = 0.95,
  axial = FALSE,
  ...
)
```

Arguments

draws	Circular draws from as_circular_draws() or any object accepted by posterior::as_draws_df().
variables	Optional variables to summarize.
level	Credible interval level.
axial	Should draws be treated as axial, modulo pi?
...	Additional arguments passed to as_circular_draws() when needed.

Value

A tibble with posterior circular summaries.

See Also

Other posterior helpers: as_circular_draws(), autoplot_circular_draws()

theme_circular *Circular plot themes*

Description

Lightweight themes for circular plots. They keep a restrained grid and avoid imposing a strong visual identity.

Usage

```
theme_circular(base_size = 12, base_family = "")

theme_rose(base_size = 12, base_family = "")

theme_compass(base_size = 12, base_family = "")
```

Arguments

base_size	Base font size.
base_family	Base font family.

Value

A ggplot2 theme.

Examples

```
theme_circular()
```

watson_williams_test *Watson-Williams test for equal circular means*

Description

Wrapper around `circular::watson.williams.test()` with explicit optional dependency handling. The Watson-Williams test assumes von Mises-like groups with comparable concentrations and should be used cautiously for small samples or weakly concentrated data.

Usage

```
watson_williams_test(x, group, ...)
```

Arguments

x	Numeric vector of angles in radians.
group	Grouping variable.
...	Additional arguments passed to `circular::watson.williams.test()`.

Value

An object returned by `circular::watson.williams.test()`.

See Also

Other circular tests: `rayleigh_test()`, `stat_circular_test()`

`wind_directions` *Simulated wind directions*

Description

A reproducible simulated dataset of wind directions with station, speed and season variables. Angles are stored in radians.

Usage

```
wind_directions
```

Format

A tibble with 500 rows and 4 variables:

station Station identifier.

direction Wind direction in radians.

speed Wind speed in arbitrary units.

season Season label.

Source

Simulated for package examples.

Index

∗ **angle utilities**
angular_difference, 3
angular_distance, 4
check_angle, 11
compass_to_rad, 16
deg_to_rad, 19
hour_to_rad, 28
is_angle, 30
normalize_angle, 33
rad_to_compass, 34
rad_to_deg, 35
rad_to_hour, 35
∗ **autoplot helpers**
autoplot_circular, 8
∗ **circular density layers**
geom_circular_density, 21
stat_circular_density, 39
∗ **circular distributions**
fit_vonmises_mixture, 20
stat_vonmises, 44
stat_vonmises_fit, 46
stat_vonmises_mixture, 47
∗ **circular intervals**
geom_confidence_arc, 24
∗ **circular model helpers**
augment_circular, 7
circular_model_diagnostics, 12
circular_residuals, 13
∗ **circular scales**
coord_circular, 18
scale_x_circular_radians, 37
∗ **circular summaries**
circular_mean_ci, 11
circular_sd, 14
circular_summary, 14
circular_variance, 15
estimate_kappa, 19
mean_direction, 30
mean_resultant_length, 31
resultant_length, 36
∗ **circular tests**
rayleigh_test, 36
stat_circular_test, 41
watson_williams_test, 49
∗ **circular themes**
theme_circular, 49
∗ **datasets**
animal_steps, 4
axial_orientations, 10
hourly_activity, 29
wind_directions, 50
∗ **mean direction layers**
geom_mean_direction, 26
stat_mean_direction, 42
∗ **movement helpers**
as_step_data, 6
augment_momentuHMM_angles, 7
compute_bearing, 16
compute_step_length, 17
compute_turn_angle, 17
geom_circular_point, 23
geom_direction_arrow, 25
mutate_directional_features, 32
plot_state_angles, 33
∗ **posterior helpers**
as_circular_draws, 5
autoplot_circular_draws, 9
summarise_circular_draws, 48
∗ **rose diagram layers**
geom_rose, 27
stat_rose, 43
∗ **spherical helpers**
cartesian_to_spherical, 10
spherical_summary, 38
spherical_to_cartesian, 39

angular_difference, 3, *4*, *11*, *16*, *19*, *29, 30*, *33–35*

angular_distance, *3*, 4, *11*, *16*, *19*, *29, 30*, *33–35*
animal_steps, 4
as_circular_draws, 5, *9*, *48*
as_circular_draws(), *9*, *48*
as_step_data, 6, *8*, *17, 18*, *24*, *26*, *32*, *34*
augment_circular, 7, *13*
augment_momentuHMM_angles, *6*, 7, *17, 18*, *24*, *26*, *32*, *34*
autoplot_circular, 8
autoplot_circular_draws, *5*, 9, *48*
axial_orientations, 10

cartesian_to_spherical, 10, *38, 39*
check_angle, *3, 4*, 11, *16*, *19*, *29, 30*, *33–35*
circular_mean_ci, 11, *14, 15*, *20*, *31*, *37*
circular_model_diagnostics, *7*, 12, *13*
circular_residuals, *7*, 13, *13*
circular_sd, *12*, 14, *15*, *20*, *31*, *37*
circular_summary, *12*, 14, *14, 15*, *20*, *31*, *37*
circular_variance, *12*, *14*, 15, *15*, *20*, *31*, *37*
compass_to_rad, *3, 4*, *11*, 16, *19*, *29, 30*, *33–35*
compute_bearing, *6*, *8*, 16, *17, 18*, *24*, *26*, *32*, *34*
compute_bearing(), *6*, *32*
compute_step_length, *6*, *8*, 17, *17, 18*, *24*, *26*, *32*, *34*
compute_turn_angle, *6*, *8*, 17, *17*, *24*, *26*, *32*, *34*
coord_circular, 18, *38*
coord_circular(), *27*

deg_to_rad, *3, 4*, *11*, *16*, 19, *29, 30*, *33–35*

estimate_kappa, *12*, *14, 15*, 19, *31*, *37*
estimate_kappa(), *46*

fit_vonmises_mixture, 20, *46–48*

geom_circular_density, 21, *40*
geom_circular_interval *(*geom_confidence_arc*)*, 24
geom_circular_point, *6*, *8*, *17, 18*, 23, *26*, *32*, *34*
geom_circular_rug *(*geom_circular_point*)*, 23
geom_confidence_arc, 24
geom_direction_arrow, *6*, *8*, *17, 18*, *24*, 25, *32*, *34*
geom_mean_direction, 26, *43*
geom_rose, 27, *44*
glance_circular *(*augment_circular*)*, 7

hour_to_rad, *3, 4*, *11*, *16*, *19*, 28, *30*, *33–35*
hourly_activity, 29

is_angle, *3, 4*, *11*, *16*, *19*, *29*, 30, *33–35*

mean_direction, *12*, *14, 15*, *20*, 30, *31*, *37*
mean_direction(), *46*
mean_resultant_length, *12*, *14, 15*, *20*, 31, *31*, *37*
mutate_directional_features, *6*, *8*, *17, 18*, *24*, *26*, 32, *34*
mutate_directional_features(), *6*

normalize_angle, *3, 4*, *11*, *16*, *19*, *29, 30*, 33, *34, 35*

plot_state_angles, *6*, *8*, *17, 18*, *24*, *26*, *32*, 33

rad_to_compass, *3, 4*, *11*, *16*, *19*, *29, 30*, *33*, 34, *35*
rad_to_deg, *3, 4*, *11*, *16*, *19*, *29, 30*, *33, 34*, 35, *35*
rad_to_hour, *3, 4*, *11*, *16*, *19*, *29, 30*, *33, 34*, 35, *35*
rayleigh_test, 36, *41*, *50*
resultant_length, *12*, *14, 15*, *20*, *31*, 36

scale_x_circular_compass *(*scale_x_circular_radians*)*, 37
scale_x_circular_degrees *(*scale_x_circular_radians*)*, 37
scale_x_circular_hours *(*scale_x_circular_radians*)*, 37
scale_x_circular_radians, *19*, 37
spherical_summary, *11*, 38, *39*
spherical_to_cartesian, *11*, *38*, 39
stat_circular_density, *22*, 39
stat_circular_density(), *21*
stat_circular_test, *36*, 41, *50*
stat_mean_direction, *27*, 42
stat_rose, *28*, 43
stat_rose(), *27*
stat_uniform_circular *(*stat_vonmises*)*, 44
stat_vonmises, *21*, 44, *47, 48*

stat_vonmises_fit, *21*, 46, *46*, *48*
stat_vonmises_mixture, *21*, *46*, 47, *47*
stat_wrapped_normal *(*stat_vonmises*)*, 44
summarise_circular_draws, *5*, *9*, 48

theme_circular, 49
theme_compass *(*theme_circular*)*, 49
theme_rose *(*theme_circular*)*, 49
tidy_circular *(*augment_circular*)*, 7

watson_williams_test, *36*, *41*, 49
wind_directions, 50