

Package: ggccircular (via r-universe)

October 19, 2026

Title A 'ggplot2' Extension for Circular, Cylindrical and Toroidal Data

Version 0.2.0

Description Provides a 'ggplot2' grammar for circular, axial and directional data, including rose diagrams, circular densities, mean directions, confidence arcs, theoretical circular distributions, movement data visualizations, and displays for circular-linear and circular-circular dependence on cylinders and tori.

License MIT + file LICENSE

URL <https://github.com/AurelienNicosiaULaval/ggccircular>,
<https://aureliennicosiaulaval.github.io/ggccircular/>

BugReports <https://github.com/AurelienNicosiaULaval/ggccircular/issues>

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

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

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

Config/Needs/coverage covr

Config/Needs/website pkgdown

Config/testthat/edition 3

VignetteBuilder knitr

LazyData true

Repository <https://aureliennicosiaulaval.r-universe.dev>

Date/Publication 2026-08-20 17:12:19 UTC

RemoteUrl <https://github.com/aureliennicosiaulaval/ggcircular>

RemoteRef HEAD

RemoteSha b95492cbf23cadafebb25d364ed74697faf3896

Contents

alignment_index	4
angular_difference	5
angular_distance	5
animal_steps	6
as_circular_draws	7
as_step_data	7
augment_circular	8
augment_momentuHMM_angles	9
autoplot.bootstrap_stat_orbit	10
autoplot.bootstrap_toroidal_ridge	11
autoplot.diagnostic_atlas_data	12
autoplot_circular	13
autoplot_circular_draws	13
axial_orientations	14
bootstrap_stat_orbit	15
bootstrap_toroidal_ridge	16
cartesian_to_spherical	17
check_angle	17
circular_mean_ci	18
circular_model_diagnostics	19
circular_residuals	20
circular_sd	20
circular_summary	21
circular_topography_data	22
circular_variance	23
compass_to_rad	23
compute_bearing	24
compute_step_length	25
compute_turn_angle	25
conditional_variation_index	26
coord_circular	27
deg_to_rad	28
diagnostic_atlas_data	28
diagnostic_scenarios	29
estimate_cyl_density	30
estimate_kappa	31
estimate_toroidal_density	31
fit_vonmises_mixture	32
geom_circular_density	34
geom_circular_point	35
geom_confidence_arc	36

geom_direction_arrow	37
geom_mean_direction	38
geom_phase_loom	39
geom_rose	40
geom_toroidal_flow	42
hour_to_rad	43
hourly_activity	43
is_angle	44
local_circ_moment	44
marginal_support_data	45
mean_direction	46
mean_resultant_length	47
mutate_directional_features	47
normalize_angle	48
phase_loom_data	49
plot_circular_topography	50
plot_circular_topography_3d	51
plot_classical_comparison	52
plot_diagnostic_atlas	52
plot_marginal_support	53
plot_phase_loom	54
plot_stat_orbit	55
plot_state_angles	56
plot_toroidal_flow	57
plot_toroidal_ridge	58
plot_toroidal_topography	59
plot_toroidal_topography_3d	60
rad_to_compass	61
rad_to_deg	62
rad_to_hour	62
rayleigh_test	63
resultant_length	63
scale_x_circular_radians	64
select_cyl_smoothing	65
select_toroidal_smoothing	66
sensitivity_bandwidth_grid	67
simulate_cyl_diagnostic	68
simulate_cylindrical	68
simulate_independence_lineup	69
simulate_tor_diagnostic	70
simulate_toroidal	71
spherical_summary	72
spherical_to_cartesian	73
stat_circular_density	73
stat_circular_test	75
stat_circular_topography	76
stat_mean_direction	77
stat_orbit_data	78

stat_rose	79
stat_statistical_orbit	80
stat_statistical_orbit_ribbon	81
stat_toroidal_ridge	82
stat_toroidal_topography	83
stat_vonmises	84
stat_vonmises_fit	86
stat_vonmises_mixture	87
summarise_circular_draws	88
theme_circular	89
toroidal_flow_data	89
toroidal_ridge_data	90
toroidal_topography_data	91
watson_williams_test	92
wind_directions	93

Index	94
--------------	-----------

alignment_index	<i>Alignment index for two angular variables</i>
-----------------	--

Description

Computes $|\text{mean}(\exp(i * (\text{phi} - \text{theta})))|$, a scalar summary of circular alignment between two angular variables.

Usage

```
alignment_index(theta, phi)
```

Arguments

- theta First angle in radians.
- phi Second angle in radians.

Value

A numeric scalar between 0 and 1.

See Also

Other directional dependence indicators: [conditional_variation_index\(\)](#), [local_circ_moment\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, scenario = "diagonal", seed = 1)
alignment_index(dat$theta, dat$phi)
```

angular_difference	<i>Signed angular difference</i>
--------------------	----------------------------------

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	<i>Circular angular distance</i>
------------------	----------------------------------

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	<i>Convert posterior draws to circular draws</i>
-------------------	--

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	<i>Coerce to step data</i>
--------------	----------------------------

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	<i>Circular model helper generics</i>
------------------	---------------------------------------

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

<code>object</code>	A fitted momentuHMM object.
<code>data</code>	Optional data frame. If NULL, <code>object\$data</code> is used.
<code>angle</code>	Optional name of the angle column.
<code>state_method</code>	State extraction method.
<code>...</code>	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.bootstrap_stat_orbit
```

Plot bootstrap uncertainty for a statistical orbit

Description

Plot bootstrap uncertainty for a statistical orbit

Usage

```
## S3 method for class 'bootstrap_stat_orbit'
autoplot(
  object,
  ...,
  quantity = c("mu", "sigma"),
  base_size = 11,
  ribbon_fill = "#8fb4ff",
  line_colour = "#123f9b"
)
```

Arguments

<code>object</code>	Object returned by <code>bootstrap_stat_orbit()</code> .
<code>...</code>	Reserved for future extensions.
<code>quantity</code>	Either "mu" or "sigma".
<code>base_size</code>	Base font size.
<code>ribbon_fill</code>	Ribbon fill colour.
<code>line_colour</code>	Line colour.

Value

A ggplot object.

See Also

Other directional uncertainty helpers: `autoplot.bootstrap_toroidal_ridge()`, `bootstrap_stat_orbit()`, `bootstrap_toroidal_ridge()`, `marginal_support_data()`, `plot_marginal_support()`, `sensitivity_bandwidth_gradients()`, `simulate_independence_lineup()`

Examples

```
dat <- simulate_cylindrical(n = 60, seed = 1)
boot <- bootstrap_stat_orbit(dat$theta, dat$x, n_boot = 5, n_theta = 12, seed = 1)
ggplot2::autoplot(boot)
```

autoplot.bootstrap_toroidal_ridge

Plot bootstrap uncertainty for a toroidal ridge

Description

Plot bootstrap uncertainty for a toroidal ridge

Usage

```
## S3 method for class 'bootstrap_toroidal_ridge'
autoplot(
  object,
  ...,
  quantity = c("rho", "ridge"),
  base_size = 11,
  ribbon_fill = "#99d9d0",
  line_colour = "#0f766e"
)
```

Arguments

object	Object returned by bootstrap_toroidal_ridge() .
...	Reserved for future extensions.
quantity	Either "rho" or "ridge".
base_size	Base font size.
ribbon_fill	Ribbon fill colour.
line_colour	Line colour.

Value

A ggplot object.

See Also

Other directional uncertainty helpers: [autoplot.bootstrap_stat_orbit\(\)](#), [bootstrap_stat_orbit\(\)](#), [bootstrap_toroidal_ridge\(\)](#), [marginal_support_data\(\)](#), [plot_marginal_support\(\)](#), [sensitivity_bandwidth_gradients\(\)](#), [simulate_independence_lineup\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 60, scenario = "diagonal", seed = 1)
boot <- bootstrap_toroidal_ridge(dat$theta, dat$phi, n_boot = 5, n_theta = 12, n_phi = 12, seed = 1)
ggplot2::autoplot(boot)
```

```
autoplot.diagnostic_atlas_data
```

Plot diagnostic atlas data

Description

Plot diagnostic atlas data

Usage

```
## S3 method for class 'diagnostic_atlas_data'
autoplot(
  object,
  ...,
  point_alpha = 0.2,
  point_size = 0.7,
  smooth = TRUE,
  base_size = 11
)
```

Arguments

object	Object returned by diagnostic_atlas_data() .
...	Reserved for future extensions.
point_alpha	Point transparency.
point_size	Point size.
smooth	Should a loess smooth be added for cylindrical data?
base_size	Base font size.

Value

A ggplot object.

See Also

Other directional diagnostics: [diagnostic_atlas_data\(\)](#), [diagnostic_scenarios\(\)](#), [plot_classical_comparison\(\)](#), [plot_diagnostic_atlas\(\)](#), [simulate_cyl_diagnostic\(\)](#), [simulate_tor_diagnostic\(\)](#)

Examples

```
atlas <- diagnostic_atlas_data("cylindrical", n = 20, seed = 1)
ggplot2::autoplot(atlas)
```

autoplot_circular	<i>Autoplot circular data</i>
-------------------	-------------------------------

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	<i>Autoplot circular posterior draws</i>
-------------------------	--

Description

Autoplot circular posterior draws

Usage

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

Arguments

<code>draws</code>	Circular draws or posterior draws.
<code>variables</code>	Optional variables to plot.
<code>type</code>	Plot type.
<code>axial</code>	Should draws be treated as axial, modulo pi?
<code>...</code>	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	<i>Simulated axial orientations</i>
--------------------	-------------------------------------

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.

bootstrap_stat_orbit *Bootstrap a statistical orbit*

Description

Bootstrap a statistical orbit

Usage

```
bootstrap_stat_orbit(  
  theta,  
  x,  
  n_boot = 100,  
  n_theta = 181,  
  kappa = 20,  
  seed = NULL  
)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
n_boot	Number of bootstrap resamples.
n_theta	Number of angular grid points.
kappa	Circular concentration for local weighting.
seed	Optional random seed.

Value

A data frame with point estimates and bootstrap intervals. The returned object has class `bootstrap_stat_orbit`.

See Also

Other directional uncertainty helpers: [autoplot.bootstrap_stat_orbit\(\)](#), [autoplot.bootstrap_toroidal_ridge\(\)](#), [bootstrap_toroidal_ridge\(\)](#), [marginal_support_data\(\)](#), [plot_marginal_support\(\)](#), [sensitivity_bandwidth_gradients\(\)](#), [simulate_independence_lineup\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 60, seed = 1)  
bootstrap_stat_orbit(dat$theta, dat$x, n_boot = 5, n_theta = 12, seed = 1)
```

bootstrap_toroidal_ridge

Bootstrap a toroidal conditional ridge

Description

Bootstrap a toroidal conditional ridge

Usage

```
bootstrap_toroidal_ridge(
  theta,
  phi,
  n_boot = 100,
  n_theta = 181,
  n_phi = 181,
  kappa_theta = 20,
  kappa_phi = 20,
  level = 0.95,
  seed = NULL
)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
n_boot	Number of bootstrap resamples.
n_theta	Number of theta grid points.
n_phi	Number of phi grid points.
kappa_theta	Kernel concentration for theta.
kappa_phi	Kernel concentration for phi.
level	Coverage level for the pointwise circular bootstrap interval.
seed	Optional random seed.

Value

A data frame with ridge and local concentration intervals. The returned object has class `bootstrap_toroidal_ridge`.

See Also

Other directional uncertainty helpers: [autoplot.bootstrap_stat_orbit\(\)](#), [autoplot.bootstrap_toroidal_ridge\(\)](#), [bootstrap_stat_orbit\(\)](#), [marginal_support_data\(\)](#), [plot_marginal_support\(\)](#), [sensitivity_bandwidth_grid\(\)](#), [simulate_independence_lineup\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 60, scenario = "diagonal", seed = 1)
bootstrap_toroidal_ridge(dat$theta, dat$phi, n_boot = 5, n_theta = 12, n_phi = 12, seed = 1)
```

`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

<code>x, y, z</code>	Cartesian coordinates.
<code>convention</code>	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	<i>Confidence interval for a circular mean</i>
------------------	--

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	<i>Circular residuals for angular models</i>
--------------------	--

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	<i>Circular standard deviation</i>
-------------	------------------------------------

Description

Uses the common descriptive statistic $\sqrt{-2 * \log(Rbar)}$.

Usage

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

Arguments

<code>x</code>	Numeric vector of angles in radians.
<code>axial</code>	Should the data be treated as axial, modulo pi?
<code>na.rm</code>	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\(\)](#)

<code>circular_summary</code>	<i>Summarize circular data</i>
-------------------------------	--------------------------------

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

<code>data</code>	A data frame or tibble.
<code>angle</code>	Angle column, in radians.
<code>...</code>	Optional grouping variables.
<code>axial</code>	Should the data be treated as axial, modulo pi?
<code>na.rm</code>	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_topography_data

Compute circular topography data

Description

Compute circular topography data

Usage

```
circular_topography_data(theta, x, conditional = TRUE, ...)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
conditional	If TRUE, rows are normalized to estimate $f(x \mid \theta)$.
...	Additional arguments passed to estimate_cyl_density() .

Value

A data frame with columns theta, x and density.

See Also

Other cylindrical dependence helpers: [estimate_cyl_density\(\)](#), [phase_loom_data\(\)](#), [stat_orbit_data\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
circular_topography_data(dat$theta, dat$x, n_theta = 24, n_x = 30)
```

circular_variance	<i>Circular variance</i>
-------------------	--------------------------

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 $1 - R_{\text{bar}}$.

See Also

Other circular summaries: [circular_mean_ci\(\)](#), [circular_sd\(\)](#), [circular_summary\(\)](#), [estimate_kappa\(\)](#), [mean_direction\(\)](#), [mean_resultant_length\(\)](#), [resultant_length\(\)](#)

compass_to_rad	<i>Convert compass labels to radians</i>
----------------	--

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	<i>Compute bearings</i>
-----------------	-------------------------

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	<i>Compute step lengths</i>
---------------------	-----------------------------

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	<i>Compute turn angles</i>
--------------------	----------------------------

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\(\)](#)

conditional_variation_index
Conditional variation index

Description

Computes a simple squared variation summary for a matrix of conditional densities across conditioning-grid rows.

Usage

```
conditional_variation_index(density_matrix)
```

Arguments

`density_matrix` Matrix with rows representing conditioning-grid values and columns representing conditional density values.

Value

A non-negative numeric scalar.

See Also

Other directional dependence indicators: [alignment_index\(\)](#), [local_circ_moment\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, seed = 1)
est <- estimate_toroidal_density(dat$theta, dat$phi, n_theta = 12, n_phi = 12, conditional = TRUE)
conditional_variation_index(est$density)
```

coord_circular	<i>Circular coordinate system</i>
----------------	-----------------------------------

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	<i>Convert degrees to radians</i>
------------	-----------------------------------

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\(\)](#)

diagnostic_atlas_data	<i>Build diagnostic atlas data</i>
-----------------------	------------------------------------

Description

Build diagnostic atlas data

Usage

```
diagnostic_atlas_data(
  space = c("cylindrical", "toroidal"),
  n = 600,
  seed = 20260609
)
```

Arguments

space	Either "cylindrical" or "toroidal".
n	Number of observations per scenario.
seed	Optional random seed.

Value

A data frame with diagnostic scenarios and plotting columns. The returned object has class `diagnostic_atlas_data`.

See Also

Other directional diagnostics: [autoplot.diagnostic_atlas_data\(\)](#), [diagnostic_scenarios\(\)](#), [plot_classical_comparison\(\)](#), [plot_diagnostic_atlas\(\)](#), [simulate_cyl_diagnostic\(\)](#), [simulate_tor_diagnostic\(\)](#).

Examples

```
diagnostic_atlas_data("cylindrical", n = 20, seed = 1)
```

<code>diagnostic_scenarios</code>	<i>List built-in directional diagnostic scenarios</i>
-----------------------------------	---

Description

List built-in directional diagnostic scenarios

Usage

```
diagnostic_scenarios()
```

Value

A data frame describing the built-in diagnostic scenarios.

See Also

Other directional diagnostics: [autoplot.diagnostic_atlas_data\(\)](#), [diagnostic_atlas_data\(\)](#), [plot_classical_comparison\(\)](#), [plot_diagnostic_atlas\(\)](#), [simulate_cyl_diagnostic\(\)](#), [simulate_tor_diagnostic\(\)](#).

Examples

```
diagnostic_scenarios()
```

estimate_cyl_density *Estimate a density on the cylinder*

Description

Estimates a joint or conditional kernel density for circular-linear data on $S^1 \times \mathbb{R}$. The circular component uses von Mises weights and the linear component uses a Gaussian kernel.

Usage

```
estimate_cyl_density(
  theta,
  x,
  n_theta = 181,
  n_x = 200,
  kappa = 20,
  h = NULL,
  x_grid = NULL,
  conditional = TRUE
)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
n_theta	Number of angular grid points.
n_x	Number of linear grid points.
kappa	Circular concentration for the conditioning kernel.
h	Linear bandwidth. If NULL, a rule-of-thumb bandwidth is used.
x_grid	Optional grid for the linear response.
conditional	If TRUE, rows are normalized to estimate $f(x \mid \theta)$.

Value

A list with grids, a density matrix, a long data frame and smoothing metadata.

See Also

Other cylindrical dependence helpers: [circular_topography_data\(\)](#), [phase_loom_data\(\)](#), [stat_orbit_data\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
est <- estimate_cyl_density(dat$theta, dat$x, n_theta = 24, n_x = 30)
str(est$data)
```

estimate_kappa	<i>Estimate von Mises concentration</i>
----------------	---

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\(\)](#)

estimate_toroidal_density	<i>Estimate a density on the torus</i>
---------------------------	--

Description

Estimates a joint or conditional kernel density for circular-circular data on $S^1 \times S^1$. Both components use von Mises weights.

Usage

```
estimate_toroidal_density(
  theta,
  phi,
  n_theta = 181,
  n_phi = 181,
  kappa_theta = 20,
  kappa_phi = 20,
  conditional = FALSE
)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
n_theta	Number of theta grid points.
n_phi	Number of phi grid points.
kappa_theta	Kernel concentration for theta.
kappa_phi	Kernel concentration for phi.
conditional	If TRUE, rows are normalized to estimate $f(\text{phi} \mid \text{theta})$.

Value

A list with grids, a density matrix, a long data frame and smoothing metadata.

See Also

Other toroidal dependence helpers: [toroidal_flow_data\(\)](#), [toroidal_ridge_data\(\)](#), [toroidal_topography_data\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, seed = 1)
est <- estimate_toroidal_density(dat$theta, dat$phi, n_theta = 24, n_phi = 24)
str(est$data)
```

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- π 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 π ?
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	<i>Circular confidence arc</i>
---------------------	--------------------------------

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	<i>Mean direction layer</i>
---------------------	-----------------------------

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.

<code>conf.int</code>	Should approximate confidence limits be computed?
<code>level</code>	Confidence level used when <code>conf.int = TRUE</code> .
<code>axial</code>	Should the data be treated as axial, modulo π ?
<code>arrow</code>	Should a small arrow head be drawn?
<code>na.rm</code>	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()
```

<code>geom_phase_loom</code>	<i>Add a phase loom layer</i>
------------------------------	-------------------------------

Description

Adds mass-flow curves for circular-linear data. Use this layer with `ggplot2::ggplot(data, ggplot2::aes(x = theta, y = response))`.

Usage

```
geom_phase_loom(
  mapping = NULL,
  data = NULL,
  position = "identity",
  n_sectors = 48,
  n_x_bins = 24,
  min_mass = 0.002,
  max_flows = 180,
  mass_type = c("joint", "conditional"),
  curvature = 0.24,
  ...,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
n_sectors	Number of angular sectors.
n_x_bins	Number of bins for the linear response.
min_mass	Minimum mass to draw.
max_flows	Maximum number of flows to draw. Use NULL to draw all.
mass_type	Either "joint" or "conditional".
curvature	Curvature passed to <code>ggplot2::geom_curve()</code> .
...	Additional arguments passed to the geom.

Value

A ggplot2 layer.

See Also

Other cylindrical dependence layers: [stat_circular_topography\(\)](#), [stat_statistical_orbit\(\)](#), [stat_statistical_orbit_ribbon\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
ggplot2::ggplot(dat, ggplot2::aes(x = theta, y = x)) +
  geom_phase_loom(n_sectors = 12, n_x_bins = 8, min_mass = 0)
```

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()

```

geom_toroidal_flow	<i>Add a toroidal flow layer</i>
--------------------	----------------------------------

Description

Adds rectangular flow curves from theta sectors to phi sectors. Use with `aes(x = theta, y = phi)`.

Usage

```
geom_toroidal_flow(
  mapping = NULL,
  data = NULL,
  position = "identity",
  n_sectors = 32,
  min_mass = 0.002,
  mass_type = c("joint", "conditional"),
  curvature = 0.28,
  ...,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
n_sectors	Number of angular sectors for both variables.
min_mass	Minimum mass to draw.
mass_type	Either "joint" or "conditional".
curvature	Curvature passed to <code>ggplot2::geom_curve()</code> .
...	Additional arguments passed to the geom.

Value

A ggplot2 layer.

See Also

Other toroidal dependence layers: [stat_toroidal_ridge\(\)](#), [stat_toroidal_topography\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, seed = 1)
ggplot2::ggplot(dat, ggplot2::aes(x = theta, y = phi)) +
  geom_toroidal_flow(n_sectors = 12, min_mass = 0)
```

hour_to_rad	<i>Convert hours to radians</i>
-------------	---------------------------------

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	<i>Simulated hourly activity</i>
-----------------	----------------------------------

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	<i>Test whether an object can represent angles</i>
----------	--

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\(\)](#)

local_circ_moment	<i>Local circular moment</i>
-------------------	------------------------------

Description

Computes a kernel-weighted local circular moment of phi conditional on theta.

Usage

```
local_circ_moment(theta, phi, grid = NULL, kappa = 20)
```

Arguments

theta Conditioning angles in radians.
 phi Target angles in radians.
 grid Optional angular grid. If NULL, a default grid is used.
 kappa Circular concentration for local weighting.

Value

A data frame with local real and imaginary components, mean direction mu, and local concentration rho.

See Also

Other directional dependence indicators: [alignment_index\(\)](#), [conditional_variation_index\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, scenario = "diagonal", seed = 1)
local_circ_moment(dat$theta, dat$phi, grid = seq(0, 2 * pi, length.out = 12))
```

marginal_support_data *Estimate angular marginal support*

Description

Computes a kernel-weighted marginal support score along the conditioning angle. Low relative support marks angular regions where conditional displays should be interpreted cautiously.

Usage

```
marginal_support_data(
  theta,
  grid = NULL,
  n_theta = 181,
  kappa = 20,
  relative_threshold = 0.15
)
```

Arguments

theta	Conditioning angles in radians.
grid	Optional angular grid in radians.
n_theta	Number of grid points used when grid is NULL.
kappa	Circular concentration for local kernel weights.
relative_threshold	Relative support threshold used to flag low-support regions. With other ingredients fixed, the local standard error is approximately inflated by $1 / \sqrt{\text{relative_support}}$. The default 0.15 corresponds to an inflation of about 2.58 relative to the best-supported direction; 0.25 is a more conservative threshold corresponding to about twofold inflation.

Value

A data frame with grid values, support, relative support and flags.

See Also

Other directional uncertainty helpers: [autoplot.bootstrap_stat_orbit\(\)](#), [autoplot.bootstrap_toroidal_ridge\(\)](#), [bootstrap_stat_orbit\(\)](#), [bootstrap_toroidal_ridge\(\)](#), [plot_marginal_support\(\)](#), [sensitivity_bandwidth_grid\(\)](#), [simulate_independence_lineup\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 60, seed = 1)
marginal_support_data(dat$theta, n_theta = 12)
```

mean_direction	<i>Circular mean direction</i>
----------------	--------------------------------

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 $\bar{R}$, 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 <code>compute_bearing()</code> .

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	<i>Normalize angles to a periodic interval</i>
-----------------	--

Description

`normalize_angle()` maps numeric angles to $[\text{origin}, \text{origin} + \text{period})$. 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 π 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))
```

phase_loom_data	<i>Compute phase loom data</i>
-----------------	--------------------------------

Description

Discretizes circular-linear observations into angular sectors and linear bins and returns binned mass flows for phase loom displays.

Usage

```
phase_loom_data(
  theta,
  x,
  n_sectors = 48,
  n_x_bins = 24,
  min_mass = 0.002,
  max_flows = 180,
  mass_type = c("joint", "conditional")
)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
n_sectors	Number of angular sectors.
n_x_bins	Number of linear bins.
min_mass	Minimum mass to retain.
max_flows	Maximum number of flows to retain. Use NULL to keep all.
mass_type	Either "joint" for joint masses or "conditional" for masses normalized within each angular sector.

Value

A data frame with sector identifiers, endpoints and mass values.

See Also

Other cylindrical dependence helpers: [circular_topography_data\(\)](#), [estimate_cyl_density\(\)](#), [stat_orbit_data\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
phase_loom_data(dat$theta, dat$x, n_sectors = 12, n_x_bins = 8)
```

```
plot_circular_topography
```

Plot circular topography

Description

Plot circular topography

Usage

```
plot_circular_topography(
  theta,
  x,
  conditional = TRUE,
  ...,
  palette = topography_palette(),
  base_size = 12,
  legend_position = "right",
  show_legend = TRUE
)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
conditional	If TRUE, rows are normalized to estimate $f(x \mid \text{theta})$.
...	Additional arguments passed to estimate_cyl_density() .
palette	Colour palette for density.
base_size	Base font size.
legend_position	Legend position.
show_legend	If FALSE, hide legends.

Value

A ggplot object.

See Also

Other cylindrical dependence plots: [plot_circular_topography_3d\(\)](#), [plot_phase_loom\(\)](#), [plot_stat_orbit\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
plot_circular_topography(dat$theta, dat$x, n_theta = 24, n_x = 30)
```

plot_circular_topography_3d

Experimental 3D circular topography

Description

Displays a circular-linear conditional density on a cylinder. The function is experimental and is intended as a geometric companion to [plot_circular_topography\(\)](#), not as a replacement for the 2D display.

Usage

```
plot_circular_topography_3d(theta, x, radius = 1, conditional = TRUE, ...)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
radius	Radius of the cylinder.
conditional	If TRUE, rows are normalized to estimate $f(x \mid \text{theta})$.
...	Additional arguments passed to circular_topography_data() .

Value

A plotly htmlwidget.

See Also

Other cylindrical dependence plots: [plot_circular_topography\(\)](#), [plot_phase_loom\(\)](#), [plot_stat_orbit\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 120, seed = 1)
plot_circular_topography_3d(dat$theta, dat$x, n_theta = 32, n_x = 40)
```

```
plot_classical_comparison
```

Plot a classical unfolded comparison

Description

Plot a classical unfolded comparison

Usage

```
plot_classical_comparison(data, space = c("cylindrical", "toroidal"))
```

Arguments

<code>data</code>	Data frame containing theta and either x or phi.
<code>space</code>	Either "cylindrical" or "toroidal".

Value

A ggplot object.

See Also

Other directional diagnostics: [autoplot.diagnostic_atlas_data\(\)](#), [diagnostic_atlas_data\(\)](#), [diagnostic_scenarios\(\)](#), [plot_diagnostic_atlas\(\)](#), [simulate_cyl_diagnostic\(\)](#), [simulate_tor_diagnostic\(\)](#)

Examples

```
dat <- simulate_cyl_diagnostic(n = 40, scenario = "smooth", seed = 1)
plot_classical_comparison(dat, "cylindrical")
```

```
plot_diagnostic_atlas
```

Plot a diagnostic atlas

Description

Plot a diagnostic atlas

Usage

```
plot_diagnostic_atlas(
  space = c("cylindrical", "toroidal"),
  n = 600,
  seed = 20260609
)
```

Arguments

space	Either "cylindrical" or "toroidal".
n	Number of observations per scenario.
seed	Optional random seed.

Value

A ggplot object.

See Also

Other directional diagnostics: [autoplot.diagnostic_atlas_data\(\)](#), [diagnostic_atlas_data\(\)](#), [diagnostic_scenarios\(\)](#), [plot_classical_comparison\(\)](#), [simulate_cyl_diagnostic\(\)](#), [simulate_tor_diagnostic\(\)](#)

Examples

```
plot_diagnostic_atlas("toroidal", n = 20, seed = 1)
```

plot_marginal_support *Plot angular marginal support*

Description

Plot angular marginal support

Usage

```
plot_marginal_support(
  theta,
  grid = NULL,
  n_theta = 181,
  kappa = 20,
  relative_threshold = 0.15,
  base_size = 11
)
```

Arguments

theta	Conditioning angles in radians.
grid	Optional angular grid in radians.
n_theta	Number of grid points used when grid is NULL.
kappa	Circular concentration for local kernel weights.
relative_threshold	Relative support threshold used to flag low-support regions.
base_size	Base font size.

Value

A ggplot object.

See Also

Other directional uncertainty helpers: [autoplot.bootstrap_stat_orbit\(\)](#), [autoplot.bootstrap_toroidal_ridge\(\)](#), [bootstrap_stat_orbit\(\)](#), [bootstrap_toroidal_ridge\(\)](#), [marginal_support_data\(\)](#), [sensitivity_bandwidth_grid\(\)](#), [simulate_independence_lineup\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 60, seed = 1)
plot_marginal_support(dat$theta, n_theta = 12)
```

plot_phase_loom	<i>Plot a phase loom</i>
-----------------	--------------------------

Description

Plot a phase loom

Usage

```
plot_phase_loom(
  theta,
  x,
  n_sectors = 48,
  n_x_bins = 24,
  min_mass = 0.002,
  max_flows = 180,
  mass_type = c("joint", "conditional"),
  palette = flow_palette(),
  base_size = 12,
  legend_position = "right",
  show_legend = TRUE
)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
n_sectors	Number of angular sectors.
n_x_bins	Number of linear bins.
min_mass	Minimum mass to retain.
max_flows	Maximum number of flows to retain. Use NULL to keep all.

mass_type	Either "joint" for joint masses or "conditional" for masses normalized within each angular sector.
palette	Colour palette for response levels.
base_size	Base font size.
legend_position	Legend position.
show_legend	If FALSE, hide legends.

Value

A ggplot object.

See Also

Other cylindrical dependence plots: [plot_circular_topography\(\)](#), [plot_circular_topography_3d\(\)](#), [plot_stat_orbit\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
plot_phase_loom(dat$theta, dat$x, n_sectors = 12, n_x_bins = 8)
```

plot_stat_orbit	<i>Plot a statistical orbit</i>
-----------------	---------------------------------

Description

Plot a statistical orbit

Usage

```
plot_stat_orbit(theta, x, n_theta = 181, kappa = 20, base_size = 12)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
n_theta	Number of angular grid points.
kappa	Circular concentration for local weighting.
base_size	Base font size.

Value

A ggplot object.

See Also

Other cylindrical dependence plots: [plot_circular_topography\(\)](#), [plot_circular_topography_3d\(\)](#), [plot_phase_loom\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
plot_stat_orbit(dat$theta, dat$x, n_theta = 24)
```

plot_state_angles	<i>Plot angular distributions by state</i>
-------------------	--

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\(\)](#)

plot_toroidal_flow	<i>Plot toroidal flow</i>
--------------------	---------------------------

Description

Plot toroidal flow

Usage

```
plot_toroidal_flow(
  theta,
  phi,
  n_sectors = 32,
  min_mass = 0.002,
  mass_type = c("joint", "conditional"),
  palette = flow_palette(),
  base_size = 12,
  legend_position = "right",
  show_legend = TRUE
)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
n_sectors	Number of sectors for both angles.
min_mass	Minimum mass to retain.
mass_type	Either "joint" for joint masses or "conditional" for masses normalized within each theta sector.
palette	Colour palette for theta sectors.
base_size	Base font size.
legend_position	Legend position.
show_legend	If FALSE, hide legends.

Value

A ggplot object.

See Also

Other toroidal dependence plots: [plot_toroidal_ridge\(\)](#), [plot_toroidal_topography\(\)](#), [plot_toroidal_topography](#)

Examples

```
dat <- simulate_toroidal(n = 80, seed = 1)
plot_toroidal_flow(dat$theta, dat$phi, n_sectors = 12)
```

plot_toroidal_ridge *Plot a conditional toroidal ridge*

Description

Plot a conditional toroidal ridge

Usage

```
plot_toroidal_ridge(
  theta,
  phi,
  n_theta = 181,
  n_phi = 181,
  kappa_theta = 20,
  kappa_phi = 20,
  tie_tolerance = 0.01,
  density_palette = c("#f6f7f9", "#d7dce1", "#aab4bf", "#6d7f91", "#33485c"),
  rho_palette = ridge_palette(),
  base_size = 12,
  legend_position = "right",
  show_legend = TRUE
)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
n_theta	Number of theta grid points.
n_phi	Number of phi grid points.
kappa_theta	Kernel concentration for theta.
kappa_phi	Kernel concentration for phi.
tie_tolerance	Relative tolerance used to flag near-tied distinct local modes without changing the selected ridge.
density_palette	Colour palette for the density background.
rho_palette	Colour palette for local concentration.
base_size	Base font size.
legend_position	Legend position.
show_legend	If FALSE, hide legends.

Value

A ggplot object.

See Also

Other toroidal dependence plots: [plot_toroidal_flow\(\)](#), [plot_toroidal_topography\(\)](#), [plot_toroidal_topography_](#)

Examples

```
dat <- simulate_toroidal(n = 80, scenario = "diagonal", seed = 1)
plot_toroidal_ridge(dat$theta, dat$phi, n_theta = 24, n_phi = 24)
```

```
plot_toroidal_topography
```

Plot toroidal topography

Description

Plot toroidal topography

Usage

```
plot_toroidal_topography(
  theta,
  phi,
  conditional = FALSE,
  ...,
  palette = topography_palette(),
  base_size = 12,
  legend_position = "right",
  show_legend = TRUE
)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
conditional	If TRUE, rows are normalized to estimate $f(\text{phi} \mid \text{theta})$.
...	Additional arguments passed to estimate_toroidal_density() .
palette	Colour palette for density.
base_size	Base font size.
legend_position	Legend position.
show_legend	If FALSE, hide legends.

Value

A ggplot object.

See Also

Other toroidal dependence plots: [plot_toroidal_flow\(\)](#), [plot_toroidal_ridge\(\)](#), [plot_toroidal_topography_3d\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, seed = 1)
plot_toroidal_topography(dat$theta, dat$phi, n_theta = 24, n_phi = 24)
```

plot_toroidal_topography_3d

Experimental 3D toroidal topography

Description

Displays a circular-circular density on a torus. Colour represents the estimated density and, by default, a small radial displacement also highlights high-density regions. The function is experimental and is intended as a geometric companion to [plot_toroidal_topography\(\)](#).

Usage

```
plot_toroidal_topography_3d(
  theta,
  phi,
  major_radius = 1,
  minor_radius = 0.35,
  density_scale = 0.15,
  conditional = TRUE,
  ...
)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
major_radius	Major torus radius.
minor_radius	Minor torus radius.
density_scale	Non-negative radial displacement applied after scaling density to $[0, 1]$. Use 0 for colour-only density encoding.
conditional	If TRUE, rows are normalized to estimate $f(\text{phi} \mid \text{theta})$.
...	Additional arguments passed to toroidal_topography_data() .

Value

A plotly htmlwidget.

See Also

Other toroidal dependence plots: [plot_toroidal_flow\(\)](#), [plot_toroidal_ridge\(\)](#), [plot_toroidal_topography\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 120, seed = 1)
plot_toroidal_topography_3d(dat$theta, dat$phi, n_theta = 32, n_phi = 32)
```

rad_to_compass	<i>Convert radians to compass labels</i>
----------------	--

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	<i>Convert radians to degrees</i>
------------	-----------------------------------

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	<i>Convert radians to hours</i>
-------------	---------------------------------

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	<i>Rayleigh test for circular uniformity</i>
---------------	--

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

<code>x</code>	Numeric vector of angles in radians.
<code>axial</code>	Should data be treated as axial, modulo π ?
<code>na.rm</code>	Should missing values be removed?

Value

An object of class `htest`.

See Also

Other circular tests: [stat_circular_test\(\)](#), [watson_williams_test\(\)](#)

resultant_length	<i>Resultant length</i>
------------------	-------------------------

Description

Resultant length

Usage

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

Arguments

<code>x</code>	Numeric vector of angles in radians.
<code>axial</code>	Should the data be treated as axial, modulo π ?
<code>na.rm</code>	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 <code>ggplot2::scale_x_continuous()</code> .

Value

A ggplot2 scale.

See Also

Other circular scales: [coord_circular\(\)](#)

Examples

```
scale_x_circular_radians()
```

`select_cyl_smoothing` *Select smoothing parameters for circular-linear conditional density*

Description

Ranks candidate smoothing parameters by K-fold conditional log-likelihood.

Usage

```
select_cyl_smoothing(
  theta,
  x,
  kappa_values = c(6, 8, 12, 16, 24, 36),
  h_values = NULL,
  n_folds = 5,
  seed = NULL
)
```

Arguments

<code>theta</code>	Conditioning angles in radians.
<code>x</code>	Linear response.
<code>kappa_values</code>	Candidate circular concentration values.
<code>h_values</code>	Candidate linear bandwidth values. If <code>NULL</code> , a small grid is built around the rule-of-thumb bandwidth used by estimate_cyl_density() .
<code>n_folds</code>	Number of cross-validation folds.
<code>seed</code>	Optional random seed for fold assignment.

Value

A data frame ranked by decreasing mean held-out log score. The returned object has class `directional_smoothing_select`.

See Also

Other directional smoothing helpers: [select_toroidal_smoothing\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
select_cyl_smoothing(
  dat$theta, dat$x,
  kappa_values = c(8, 12),
  h_values = c(0.3, 0.5),
  n_folds = 2,
  seed = 1
)
```

```
select_toroidal_smoothing
```

Select smoothing parameters for toroidal conditional density

Description

Ranks candidate smoothing parameters by K-fold conditional log-likelihood for $f(\phi \mid \theta)$.

Usage

```
select_toroidal_smoothing(
  theta,
  phi,
  kappa_theta_values = c(6, 8, 12, 16, 24, 36),
  kappa_phi_values = kappa_theta_values,
  n_folds = 5,
  seed = NULL
)
```

Arguments

theta	Conditioning angles in radians.
phi	Response angles in radians.
kappa_theta_values	Candidate concentrations for the conditioning angle.
kappa_phi_values	Candidate concentrations for the response angle.
n_folds	Number of cross-validation folds.
seed	Optional random seed for fold assignment.

Value

A data frame ranked by decreasing mean held-out log score. The returned object has class `directional_smoothing_select`.

See Also

Other directional smoothing helpers: [select_cyl_smoothing\(\)](#)

Examples

```

dat <- simulate_toroidal(n = 80, scenario = "diagonal", seed = 1)
select_toroidal_smoothing(
  dat$theta, dat$phi,
  kappa_theta_values = c(8, 12),
  kappa_phi_values = c(8, 12),
  n_folds = 2,
  seed = 1
)

```

sensitivity_bandwidth_grid

Evaluate sensitivity to smoothing concentration

Description

Evaluate sensitivity to smoothing concentration

Usage

```

sensitivity_bandwidth_grid(
  theta,
  target,
  space = c("cylindrical", "toroidal"),
  kappa_values = c(8, 12, 20, 32)
)

```

Arguments

theta	Conditioning angle in radians.
target	Real-valued response or second angle.
space	Either "cylindrical" or "toroidal".
kappa_values	Numeric vector of candidate concentrations.

Value

A data frame of compact sensitivity summaries.

See Also

Other directional uncertainty helpers: [autoplot.bootstrap_stat_orbit\(\)](#), [autoplot.bootstrap_toroidal_ridge\(\)](#), [bootstrap_stat_orbit\(\)](#), [bootstrap_toroidal_ridge\(\)](#), [marginal_support_data\(\)](#), [plot_marginal_support\(\)](#), [simulate_independence_lineup\(\)](#)

Examples

```

dat <- simulate_cylindrical(n = 60, seed = 1)
sensitivity_bandwidth_grid(dat$theta, dat$x, space = "cylindrical", kappa_values = c(8, 12))

```

```
simulate_cyl_diagnostic
```

Simulate diagnostic circular-linear data

Description

Simulate diagnostic circular-linear data

Usage

```
simulate_cyl_diagnostic(
  n = 800,
  scenario = c("independent", "smooth", "heteroscedastic", "multimodal", "seam"),
  seed = NULL
)
```

Arguments

n	Number of observations.
scenario	Diagnostic scenario.
seed	Optional random seed.

Value

A data frame with theta, x and scenario.

See Also

Other directional diagnostics: [autoplot.diagnostic_atlas_data\(\)](#), [diagnostic_atlas_data\(\)](#), [diagnostic_scenarios\(\)](#), [plot_classical_comparison\(\)](#), [plot_diagnostic_atlas\(\)](#), [simulate_tor_diagnostic\(\)](#)

Examples

```
simulate_cyl_diagnostic(n = 40, scenario = "smooth", seed = 1)
```

```
simulate_cylindrical
```

Simulate circular-linear data

Description

Simulates examples on the cylinder, where theta is an angle in radians and x is a real-valued response.

Usage

```
simulate_cylindrical(
  n = 500,
  scenario = c("nonlinear", "independent", "unimodal", "heteroscedastic", "multimodal"),
  seed = NULL
)
```

Arguments

n	Number of observations.
scenario	Dependence pattern to simulate.
seed	Optional random seed.

Value

A data frame with columns theta, x and scenario.

See Also

Other directional simulation helpers: [simulate_toroidal\(\)](#)

Examples

```
simulate_cylindrical(n = 30, scenario = "nonlinear", seed = 1)
```

```
simulate_independence_lineup
```

Simulate a visual-inference lineup under independence

Description

Simulate a visual-inference lineup under independence

Usage

```
simulate_independence_lineup(
  data,
  plot_fun = NULL,
  m = 20,
  seed = NULL,
  space = c("cylindrical", "toroidal"),
  ...
)
```

Arguments

<code>data</code>	Data frame containing theta and either x or phi.
<code>plot_fun</code>	Optional function taking one data frame and returning a ggplot object.
<code>m</code>	Number of lineup panels.
<code>seed</code>	Optional random seed.
<code>space</code>	Either "cylindrical" or "toroidal".
<code>...</code>	Passed to the default plot function when <code>plot_fun</code> is NULL.

Value

A list with plots, true position and metadata. The returned object has class `directional_lineup`.

See Also

Other directional uncertainty helpers: `autoplot.bootstrap_stat_orbit()`, `autoplot.bootstrap_toroidal_ridge()`, `bootstrap_stat_orbit()`, `bootstrap_toroidal_ridge()`, `marginal_support_data()`, `plot_marginal_support()`, `sensitivity_bandwidth_grid()`

Examples

```
dat <- simulate_cyl_diagnostic(n = 40, scenario = "smooth", seed = 1)
lineup <- simulate_independence_lineup(
  dat,
  plot_fun = function(d) ggplot2::ggplot(d, ggplot2::aes(theta, x)) + ggplot2::geom_point(),
  m = 4,
  seed = 1,
  space = "cylindrical"
)
length(lineup$plots)
```

`simulate_tor_diagnostic`

Simulate diagnostic circular-circular data

Description

Simulate diagnostic circular-circular data

Usage

```
simulate_tor_diagnostic(
  n = 900,
  scenario = c("independent_concentrated", "diagonal", "doubling", "bifurcation",
    "uniform_marginal_dependence"),
  seed = NULL
)
```

Arguments

n	Number of observations.
scenario	Diagnostic scenario.
seed	Optional random seed.

Value

A data frame with theta, phi and scenario.

See Also

Other directional diagnostics: [autoplot.diagnostic_atlas_data\(\)](#), [diagnostic_atlas_data\(\)](#), [diagnostic_scenarios\(\)](#), [plot_classical_comparison\(\)](#), [plot_diagnostic_atlas\(\)](#), [simulate_cyl_diagnostic\(\)](#)

Examples

```
simulate_tor_diagnostic(n = 40, scenario = "diagonal", seed = 1)
```

simulate_toroidal	<i>Simulate circular-circular data</i>
-------------------	--

Description

Simulates examples on the torus, where both theta and phi are angles in radians.

Usage

```
simulate_toroidal(
  n = 500,
  scenario = c("nonlinear", "diagonal", "anti_diagonal", "multimodal", "independent"),
  seed = NULL
)
```

Arguments

n	Number of observations.
scenario	Dependence pattern to simulate.
seed	Optional random seed.

Value

A data frame with columns theta, phi and scenario.

See Also

Other directional simulation helpers: [simulate_cylindrical\(\)](#)

Examples

```
simulate_toroidal(n = 30, scenario = "diagonal", seed = 1)
```

spherical_summary	<i>Summarize spherical directions</i>
-------------------	---------------------------------------

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

<code>theta</code>	Azimuth angle in radians.
<code>phi</code>	Colatitude or elevation angle in radians.
<code>radius</code>	Radius.
<code>convention</code>	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 π ?
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	<i>Annotate circular tests</i>
--------------------	--------------------------------

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_circular_topography

Add a circular topography stat layer

Description

Estimates a circular-linear density and returns grid values for topographic displays. Use with `aes(x = theta, y = response)`.

Usage

```
stat_circular_topography(
  mapping = NULL,
  data = NULL,
  geom = "tile",
  position = "identity",
  n_theta = 181,
  n_x = 200,
  kappa = 20,
  h = NULL,
  conditional = TRUE,
  ...,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

<code>mapping</code> , <code>data</code> , <code>geom</code> , <code>position</code> , <code>show.legend</code> , <code>inherit.aes</code>	Standard ggplot2 layer arguments.
<code>n_theta</code>	Number of angular grid points.
<code>n_x</code>	Number of linear grid points.
<code>kappa</code>	Circular concentration for the conditioning kernel.
<code>h</code>	Linear bandwidth. If <code>NULL</code> , a rule-of-thumb bandwidth is used.
<code>conditional</code>	If <code>TRUE</code> , estimate $f(y \mid x_angle)$.
<code>...</code>	Additional arguments passed to the geom.

Value

A ggplot2 layer.

See Also

Other cylindrical dependence layers: [geom_phase_loom\(\)](#), [stat_statistical_orbit\(\)](#), [stat_statistical_orbit_ribbon\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
ggplot2::ggplot(dat, ggplot2::aes(x = theta, y = x)) +
  stat_circular_topography(n_theta = 24, n_x = 30)
```

stat_mean_direction	<i>Mean direction statistic</i>
---------------------	---------------------------------

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_orbit_data	<i>Compute statistical orbit data</i>
-----------------	---------------------------------------

Description

Computes local conditional means, local standard deviations and local skewness over an angular grid.

Usage

```
stat_orbit_data(theta, x, n_theta = 181, kappa = 20)
```

Arguments

theta	Angles in radians.
x	Real-valued response.
n_theta	Number of angular grid points.
kappa	Circular concentration for local weighting.

Value

A data frame with theta, mu, lower, upper and skew.

See Also

Other cylindrical dependence helpers: [circular_topography_data\(\)](#), [estimate_cyl_density\(\)](#), [phase_loom_data\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
stat_orbit_data(dat$theta, dat$x, n_theta = 24)
```

stat_rose	<i>Rose diagram statistic</i>
-----------	-------------------------------

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_statistical_orbit

Add a statistical orbit line

Description

Add a statistical orbit line

Usage

```
stat_statistical_orbit(
  mapping = NULL,
  data = NULL,
  geom = "line",
  position = "identity",
  n_theta = 181,
  kappa = 20,
  ...,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, geom, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
n_theta	Number of angular grid points.
kappa	Circular concentration for local weighting.
...	Additional arguments passed to the geom.

Value

A ggplot2 layer.

See Also

Other cylindrical dependence layers: [geom_phase_loom\(\)](#), [stat_circular_topography\(\)](#), [stat_statistical_orbit_ribbon\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
ggplot2::ggplot(dat, ggplot2::aes(x = theta, y = x)) +
  stat_statistical_orbit(n_theta = 24)
```

```
stat_statistical_orbit_ribbon
```

Add a statistical orbit ribbon

Description

Add a statistical orbit ribbon

Usage

```
stat_statistical_orbit_ribbon(
  mapping = NULL,
  data = NULL,
  geom = "ribbon",
  position = "identity",
  n_theta = 181,
  kappa = 20,
  ...,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, geom, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
n_theta	Number of angular grid points.
kappa	Circular concentration for local weighting.
...	Additional arguments passed to the geom.

Value

A ggplot2 layer.

See Also

Other cylindrical dependence layers: [geom_phase_loom\(\)](#), [stat_circular_topography\(\)](#), [stat_statistical_orbit_ribbon\(\)](#)

Examples

```
dat <- simulate_cylindrical(n = 80, seed = 1)
ggplot2::ggplot(dat, ggplot2::aes(x = theta, y = x)) +
  stat_statistical_orbit_ribbon(n_theta = 24, alpha = 0.2) +
  stat_statistical_orbit(n_theta = 24)
```

stat_toroidal_ridge	<i>Add a conditional ridge stat layer on the torus</i>
---------------------	--

Description

Adds a conditional modal ridge for circular-circular data. Use with `aes(x = theta, y = phi)`.

Usage

```
stat_toroidal_ridge(
  mapping = NULL,
  data = NULL,
  geom = "path",
  position = "identity",
  n_theta = 181,
  n_phi = 181,
  kappa_theta = 20,
  kappa_phi = 20,
  tie_tolerance = 0.01,
  signed = TRUE,
  ...,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, geom, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
n_theta	Number of theta grid points.
n_phi	Number of phi grid points.
kappa_theta	Kernel concentration for theta.
kappa_phi	Kernel concentration for phi.
tie_tolerance	Relative tolerance used to flag near-tied distinct local modes without changing the selected ridge.
signed	If TRUE, plot angles in the signed interval.
...	Additional arguments passed to the geom.

Value

A ggplot2 layer.

See Also

Other toroidal dependence layers: [geom_toroidal_flow\(\)](#), [stat_toroidal_topography\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, scenario = "diagonal", seed = 1)
ggplot2::ggplot(dat, ggplot2::aes(x = theta, y = phi)) +
  stat_toroidal_ridge(n_theta = 24, n_phi = 24)
```

stat_toroidal_topography

Add a toroidal topography stat layer

Description

Estimates a circular-circular density and returns grid values for topographic displays. Use with `aes(x = theta, y = phi)`.

Usage

```
stat_toroidal_topography(
  mapping = NULL,
  data = NULL,
  geom = "tile",
  position = "identity",
  n_theta = 181,
  n_phi = 181,
  kappa_theta = 20,
  kappa_phi = 20,
  conditional = FALSE,
  signed = TRUE,
  ...,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, geom, position, show.legend, inherit.aes	Standard ggplot2 layer arguments.
n_theta	Number of theta grid points.
n_phi	Number of phi grid points.

kappa_theta	Kernel concentration for theta.
kappa_phi	Kernel concentration for phi.
conditional	If TRUE, estimate $f(\phi \mid \theta)$.
signed	If TRUE, plot angles in the signed interval.
...	Additional arguments passed to the geom.

Value

A ggplot2 layer.

See Also

Other toroidal dependence layers: [geom_toroidal_flow\(\)](#), [stat_toroidal_ridge\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, seed = 1)
ggplot2::ggplot(dat, ggplot2::aes(x = theta, y = phi)) +
  stat_toroidal_topography(n_theta = 24, n_phi = 24)
```

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 π ?
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	<i>Fitted von Mises density</i>
-------------------	---------------------------------

Description

Estimates μ with [mean_direction\(\)](#) and κ 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 π ?
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 <code>ggcircular_vonmises_mixture</code> object. If <code>NULL</code> , 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 <code>posterior::as_draws_df()</code> .
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	<i>Circular plot themes</i>
----------------	-----------------------------

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()
```

toroidal_flow_data	<i>Compute toroidal flow data</i>
--------------------	-----------------------------------

Description

Discretizes two angular variables into sectors and estimates joint or conditional mass flows between sectors.

Usage

```
toroidal_flow_data(
  theta,
  phi,
  n_sectors = 32,
  min_mass = 0.002,
  mass_type = c("joint", "conditional")
)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
n_sectors	Number of sectors for both angles.
min_mass	Minimum mass to retain.
mass_type	Either "joint" for joint masses or "conditional" for masses normalized within each theta sector.

Value

A data frame with sector identifiers, endpoints and mass values.

See Also

Other toroidal dependence helpers: [estimate_toroidal_density\(\)](#), [toroidal_ridge_data\(\)](#), [toroidal_topography_data\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, seed = 1)
toroidal_flow_data(dat$theta, dat$phi, n_sectors = 12)
```

toroidal_ridge_data	<i>Compute a conditional toroidal ridge</i>
---------------------	---

Description

Extracts the modal phi value along a grid of conditioning angles and attaches the local circular concentration.

Usage

```
toroidal_ridge_data(
  theta,
  phi,
  n_theta = 181,
  n_phi = 181,
  kappa_theta = 20,
  kappa_phi = 20,
  tie_tolerance = 0.01
)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
n_theta	Number of theta grid points.
n_phi	Number of phi grid points.
kappa_theta	Kernel concentration for theta.
kappa_phi	Kernel concentration for phi.
tie_tolerance	Relative tolerance used to flag two distinct local maxima as near-tied. The selected ridge remains the first exact grid maximum for backward compatibility.

Value

A data frame with columns theta, phi, rho, ridge_group, n_local_modes, secondary_mode_ratio and ridge_ambiguous.

See Also

Other toroidal dependence helpers: [estimate_toroidal_density\(\)](#), [toroidal_flow_data\(\)](#), [toroidal_topography_data\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, scenario = "diagonal", seed = 1)
toroidal_ridge_data(dat$theta, dat$phi, n_theta = 24, n_phi = 24)
```

```
toroidal_topography_data
```

Compute toroidal topography data

Description

Compute toroidal topography data

Usage

```
toroidal_topography_data(theta, phi, conditional = FALSE, ...)
```

Arguments

theta	First angle in radians.
phi	Second angle in radians.
conditional	If TRUE, rows are normalized to estimate $f(\text{phi} \mid \text{theta})$.
...	Additional arguments passed to estimate_toroidal_density() .

Value

A data frame with columns `theta`, `phi` and `density`.

See Also

Other toroidal dependence helpers: [estimate_toroidal_density\(\)](#), [toroidal_flow_data\(\)](#), [toroidal_ridge_data\(\)](#)

Examples

```
dat <- simulate_toroidal(n = 80, seed = 1)
toroidal_topography_data(dat$theta, dat$phi, n_theta = 24, n_phi = 24)
```

watson_williams_test	<i>Watson-Williams test for equal circular means</i>
----------------------	--

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

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

Value

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

See Also

Other circular tests: [rayleigh_test\(\)](#), [stat_circular_test\(\)](#)

wind_directions	<i>Simulated wind directions</i>
-----------------	----------------------------------

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, 5
 - angular_distance, 5
 - check_angle, 17
 - compass_to_rad, 23
 - deg_to_rad, 28
 - hour_to_rad, 43
 - is_angle, 44
 - normalize_angle, 48
 - rad_to_compass, 61
 - rad_to_deg, 62
 - rad_to_hour, 62
- * **autoplot helpers**
 - autoplot_circular, 13
- * **circular density layers**
 - geom_circular_density, 34
 - stat_circular_density, 73
- * **circular distributions**
 - fit_vonmises_mixture, 32
 - stat_vonmises, 84
 - stat_vonmises_fit, 86
 - stat_vonmises_mixture, 87
- * **circular intervals**
 - geom_confidence_arc, 36
- * **circular model helpers**
 - augment_circular, 8
 - circular_model_diagnostics, 19
 - circular_residuals, 20
- * **circular scales**
 - coord_circular, 27
 - scale_x_circular_radians, 64
- * **circular summaries**
 - circular_mean_ci, 18
 - circular_sd, 20
 - circular_summary, 21
 - circular_variance, 23
 - estimate_kappa, 31
 - mean_direction, 46
 - mean_resultant_length, 47
 - resultant_length, 63
- * **circular tests**
 - rayleigh_test, 63
 - stat_circular_test, 75
 - watson_williams_test, 92
- * **circular themes**
 - theme_circular, 89
- * **cylindrical dependence helpers**
 - circular_topography_data, 22
 - estimate_cyl_density, 30
 - phase_loom_data, 49
 - stat_orbit_data, 78
- * **cylindrical dependence layers**
 - geom_phase_loom, 39
 - stat_circular_topography, 76
 - stat_statistical_orbit, 80
 - stat_statistical_orbit_ribbon, 81
- * **cylindrical dependence plots**
 - plot_circular_topography, 50
 - plot_circular_topography_3d, 51
 - plot_phase_loom, 54
 - plot_stat_orbit, 55
- * **datasets**
 - animal_steps, 6
 - axial_orientations, 14
 - hourly_activity, 43
 - wind_directions, 93
- * **directional dependence indicators**
 - alignment_index, 4
 - conditional_variation_index, 26
 - local_circ_moment, 44
- * **directional diagnostics**
 - autoplot.diagnostic_atlas_data, 12
 - diagnostic_atlas_data, 28
 - diagnostic_scenarios, 29
 - plot_classical_comparison, 52
 - plot_diagnostic_atlas, 52
 - simulate_cyl_diagnostic, 68
 - simulate_tor_diagnostic, 70

- * **directional simulation helpers**
 - simulate_cylindrical, 68
 - simulate_toroidal, 71
- * **directional smoothing helpers**
 - select_cyl_smoothing, 65
 - select_toroidal_smoothing, 66
- * **directional uncertainty helpers**
 - autoplot.bootstrap_stat_orbit, 10
 - autoplot.bootstrap_toroidal_ridge, 11
 - bootstrap_stat_orbit, 15
 - bootstrap_toroidal_ridge, 16
 - marginal_support_data, 45
 - plot_marginal_support, 53
 - sensitivity_bandwidth_grid, 67
 - simulate_independence_lineup, 69
- * **mean direction layers**
 - geom_mean_direction, 38
 - stat_mean_direction, 77
- * **movement helpers**
 - as_step_data, 7
 - augment_momentuHMM_angles, 9
 - compute_bearing, 24
 - compute_step_length, 25
 - compute_turn_angle, 25
 - geom_circular_point, 35
 - geom_direction_arrow, 37
 - mutate_directional_features, 47
 - plot_state_angles, 56
- * **posterior helpers**
 - as_circular_draws, 7
 - autoplot_circular_draws, 13
 - summarise_circular_draws, 88
- * **rose diagram layers**
 - geom_rose, 40
 - stat_rose, 79
- * **spherical helpers**
 - cartesian_to_spherical, 17
 - spherical_summary, 72
 - spherical_to_cartesian, 73
- * **toroidal dependence helpers**
 - estimate_toroidal_density, 31
 - toroidal_flow_data, 89
 - toroidal_ridge_data, 90
 - toroidal_topography_data, 91
- * **toroidal dependence layers**
 - geom_toroidal_flow, 42
 - stat_toroidal_ridge, 82
 - stat_toroidal_topography, 83
- * **toroidal dependence plots**
 - plot_toroidal_flow, 57
 - plot_toroidal_ridge, 58
 - plot_toroidal_topography, 59
 - plot_toroidal_topography_3d, 60
- alignment_index, 4, 26, 45
- angular_difference, 5, 6, 18, 24, 28, 43, 44, 49, 61, 62
- angular_distance, 5, 5, 18, 24, 28, 43, 44, 49, 61, 62
- animal_steps, 6
- as_circular_draws, 7, 14, 88
- as_circular_draws(), 14, 88
- as_step_data, 7, 9, 24–26, 36, 38, 48, 56
- augment_circular, 8, 19, 20
- augment_momentuHMM_angles, 8, 9, 24–26, 36, 38, 48, 56
- autoplot.bootstrap_stat_orbit, 10, 11, 15, 16, 45, 54, 67, 70
- autoplot.bootstrap_toroidal_ridge, 10, 11, 15, 16, 45, 54, 67, 70
- autoplot.diagnostic_atlas_data, 12, 29, 52, 53, 68, 71
- autoplot_circular, 13
- autoplot_circular_draws, 7, 13, 88
- axial_orientations, 14
- bootstrap_stat_orbit, 10, 11, 15, 16, 45, 54, 67, 70
- bootstrap_stat_orbit(), 10
- bootstrap_toroidal_ridge, 10, 11, 15, 16, 45, 54, 67, 70
- bootstrap_toroidal_ridge(), 11
- cartesian_to_spherical, 17, 72, 73
- check_angle, 5, 6, 17, 24, 28, 43, 44, 49, 61, 62
- circular_mean_ci, 18, 21, 23, 31, 46, 47, 64
- circular_model_diagnostics, 8, 19, 20
- circular_residuals, 8, 19, 20
- circular_sd, 19, 20, 21, 23, 31, 46, 47, 64
- circular_summary, 19, 21, 21, 23, 31, 46, 47, 64
- circular_topography_data, 22, 30, 49, 78
- circular_topography_data(), 51
- circular_variance, 19, 21, 23, 31, 46, 47, 64

- compass_to_rad, [5](#), [6](#), [18](#), [23](#), [28](#), [43](#), [44](#), [49](#),
[61](#), [62](#)
- compute_bearing, [8](#), [9](#), [24](#), [25](#), [26](#), [36](#), [38](#), [48](#),
[56](#)
- compute_bearing(), [8](#), [48](#)
- compute_step_length, [8](#), [9](#), [24](#), [25](#), [26](#), [36](#),
[38](#), [48](#), [56](#)
- compute_turn_angle, [8](#), [9](#), [24](#), [25](#), [25](#), [36](#), [38](#),
[48](#), [56](#)
- conditional_variation_index, [4](#), [26](#), [45](#)
- coord_circular, [27](#), [65](#)
- coord_circular(), [40](#)
- deg_to_rad, [5](#), [6](#), [18](#), [24](#), [28](#), [43](#), [44](#), [49](#), [61](#), [62](#)
- diagnostic_atlas_data, [12](#), [28](#), [29](#), [52](#), [53](#),
[68](#), [71](#)
- diagnostic_atlas_data(), [12](#)
- diagnostic_scenarios, [12](#), [29](#), [29](#), [52](#), [53](#),
[68](#), [71](#)
- estimate_cyl_density, [22](#), [30](#), [49](#), [78](#)
- estimate_cyl_density(), [22](#), [50](#), [65](#)
- estimate_kappa, [19](#), [21](#), [23](#), [31](#), [46](#), [47](#), [64](#)
- estimate_kappa(), [86](#)
- estimate_toroidal_density, [31](#), [90–92](#)
- estimate_toroidal_density(), [59](#), [91](#)
- fit_vonmises_mixture, [32](#), [85](#), [86](#), [88](#)
- geom_circular_density, [34](#), [74](#)
- geom_circular_interval
(geom_confidence_arc), [36](#)
- geom_circular_point, [8](#), [9](#), [24–26](#), [35](#), [38](#),
[48](#), [56](#)
- geom_circular_rug
(geom_circular_point), [35](#)
- geom_confidence_arc, [36](#)
- geom_direction_arrow, [8](#), [9](#), [24–26](#), [36](#), [37](#),
[48](#), [56](#)
- geom_mean_direction, [38](#), [78](#)
- geom_phase_loom, [39](#), [76](#), [81](#)
- geom_rose, [40](#), [80](#)
- geom_toroidal_flow, [42](#), [83](#), [84](#)
- glance_circular (augment_circular), [8](#)
- hour_to_rad, [5](#), [6](#), [18](#), [24](#), [28](#), [43](#), [44](#), [49](#), [61](#),
[62](#)
- hourly_activity, [43](#)
- is_angle, [5](#), [6](#), [18](#), [24](#), [28](#), [43](#), [44](#), [49](#), [61](#), [62](#)
- local_circ_moment, [4](#), [26](#), [44](#)
- marginal_support_data, [10](#), [11](#), [15](#), [16](#), [45](#),
[54](#), [67](#), [70](#)
- mean_direction, [19](#), [21](#), [23](#), [31](#), [46](#), [47](#), [64](#)
- mean_direction(), [86](#)
- mean_resultant_length, [19](#), [21](#), [23](#), [31](#), [46](#),
[47](#), [64](#)
- mutate_directional_features, [8](#), [9](#), [24–26](#),
[36](#), [38](#), [47](#), [56](#)
- mutate_directional_features(), [7](#)
- normalize_angle, [5](#), [6](#), [18](#), [24](#), [28](#), [43](#), [44](#), [48](#),
[61](#), [62](#)
- phase_loom_data, [22](#), [30](#), [49](#), [78](#)
- plot_circular_topography, [50](#), [51](#), [55](#), [56](#)
- plot_circular_topography(), [51](#)
- plot_circular_topography_3d, [50](#), [51](#), [55](#),
[56](#)
- plot_classical_comparison, [12](#), [29](#), [52](#), [53](#),
[68](#), [71](#)
- plot_diagnostic_atlas, [12](#), [29](#), [52](#), [52](#), [68](#),
[71](#)
- plot_marginal_support, [10](#), [11](#), [15](#), [16](#), [45](#),
[53](#), [67](#), [70](#)
- plot_phase_loom, [50](#), [51](#), [54](#), [56](#)
- plot_stat_orbit, [50](#), [51](#), [55](#), [55](#)
- plot_state_angles, [8](#), [9](#), [24–26](#), [36](#), [38](#), [48](#),
[56](#)
- plot_toroidal_flow, [57](#), [59–61](#)
- plot_toroidal_ridge, [57](#), [58](#), [60](#), [61](#)
- plot_toroidal_topography, [57](#), [59](#), [59](#), [61](#)
- plot_toroidal_topography(), [60](#)
- plot_toroidal_topography_3d, [57](#), [59](#), [60](#),
[60](#)
- rad_to_compass, [5](#), [6](#), [18](#), [24](#), [28](#), [43](#), [44](#), [49](#),
[61](#), [62](#)
- rad_to_deg, [5](#), [6](#), [18](#), [24](#), [28](#), [43](#), [44](#), [49](#), [61](#),
[62](#), [62](#)
- rad_to_hour, [5](#), [6](#), [18](#), [24](#), [28](#), [43](#), [44](#), [49](#), [61](#),
[62](#), [62](#)
- rayleigh_test, [63](#), [75](#), [92](#)
- resultant_length, [19](#), [21](#), [23](#), [31](#), [46](#), [47](#), [63](#)
- scale_x_circular_compass
(scale_x_circular_radians), [64](#)
- scale_x_circular_degrees
(scale_x_circular_radians), [64](#)

scale_x_circular_hours
 (scale_x_circular_radians), 64
scale_x_circular_radians, 27, 64
select_cyl_smoothing, 65, 66
select_toroidal_smoothing, 65, 66
sensitivity_bandwidth_grid, 10, 11, 15,
 16, 45, 54, 67, 70
simulate_cyl_diagnostic, 12, 29, 52, 53,
 68, 71
simulate_cylindrical, 68, 71
simulate_independence_lineup, 10, 11, 15,
 16, 45, 54, 67, 69
simulate_tor_diagnostic, 12, 29, 52, 53,
 68, 70
simulate_toroidal, 69, 71
spherical_summary, 17, 72, 73
spherical_to_cartesian, 17, 72, 73
stat_circular_density, 34, 73
stat_circular_density(), 34
stat_circular_test, 63, 75, 92
stat_circular_topography, 40, 76, 81
stat_mean_direction, 39, 77
stat_orbit_data, 22, 30, 49, 78
stat_rose, 41, 79
stat_rose(), 40
stat_statistical_orbit, 40, 76, 80, 81
stat_statistical_orbit_ribbon, 40, 76,
 81, 81
stat_toroidal_ridge, 42, 82, 84
stat_toroidal_topography, 42, 83, 83
stat_uniform_circular (stat_vonmises),
 84
stat_vonmises, 33, 84, 86, 88
stat_vonmises_fit, 33, 85, 86, 88
stat_vonmises_mixture, 33, 85, 86, 87
stat_wrapped_normal (stat_vonmises), 84
summarise_circular_draws, 7, 14, 88

theme_circular, 89
theme_compass (theme_circular), 89
theme_rose (theme_circular), 89
tidy_circular (augment_circular), 8
toroidal_flow_data, 32, 89, 91, 92
toroidal_ridge_data, 32, 90, 90, 92
toroidal_topography_data, 32, 90, 91, 91
toroidal_topography_data(), 60

watson_williams_test, 63, 75, 92
wind_directions, 93