Package 'circumplex'

October 31, 2026

Type Package

Title Analysis and Visualization of Circular Data

Version 2.1.0

Description Circumplex models, which organize constructs in a circle around two underlying dimensions, are popular for studying interpersonal functioning, mood/affect, and vocational preferences/environments. This package provides tools for analyzing and visualizing circular data, including scoring functions for relevant instruments and a generalization of the bootstrapped structural summary method from Zimmermann & Wright (2017) <doi:10.1177/1073191115621795> and functions for creating publication-ready tables and figures from the results.

License GPL-3

URL https://github.com/jmgirard/circumplex, http://circumplex.jmgirard.com/

BugReports https://github.com/jmgirard/circumplex/issues

Depends R (>= 4.1)

Imports boot (>= 1.3-18), ggplot2 (>= 4.0.0), grid, htmlTable (>= 1.13.3), parallel, Rcpp, rlang, stats

Suggests brms, covr (>= 3.5.0), ggrepel, glmmTMB, kableExtra (>= 1.1.0), knitr (>= 1.28), lavaan, nlme, OpenMx, psych, RColorBrewer, rmarkdown (>= 2.1), roxygen2 (>= 7.1.0), testthat (>= 3.1.7), vdiffr

LinkingTo Rcpp, RcppArmadillo (>= 0.11)

VignetteBuilder knitr

Encoding UTF-8

LazyData true

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

NeedsCompilation yes

Author Jeffrey Girard [aut, cre] (ORCID:
<https://orcid.org/0000-0002-7359-3746>),
Johannes Zimmermann [aut] (ORCID:
<https://orcid.org/0000-0001-6975-2356>),
Aidan Wright [aut] (ORCID: <https://orcid.org/0000-0002-2369-0601>)

Maintainer Jeffrey Girard <me@jmgirard.com>

Repository CRAN

Date/Publication 2026-10-30 23:20:02 UTC

Contents

anchors . . . 3
angle_unwrap . . . 4
aw2009 . . . 5
axes_reliability . . . 6
coord_circumplex . . . 15
cpm_fit . . . 16
cpm_simulate . . . 21
fit_structure . . . 22
geom_ssm_arc . . . 24
geom_ssm_ellipse . . . 26
geom_ssm_path . . . 27
geom_ssm_point . . . 29
ggcircumplex . . . 30
html_render . . . 32
instruments . . . 33
ipsatize . . . 33
items . . . 34
jz2017 . . . 35
norms . . . 36
norm_standardize . . . 37
octants . . . 39
PANO . . . 39
plot.circumplex_ci_accuracy . . . 40
plot.circumplex_cpm . . . 41
plot.circumplex_structure . . . 42
poles . . . 43
print.circumplex_axes_reliability . . . 44
print.circumplex_cpm . . . 44
print.circumplex_structure . . . 45
quadrants . . . 45
raw_iipsc . . . 46
scales . . . 46
scale_x_circumplex . . . 47
score . . . 48
self_standardize . . . 49
simulated_growth . . . 50

simulated_items 51
simulated_occasions 52
ssm_analyze 52
ssm_analyze_long 58
ssm_ci_accuracy 60
ssm_draws 64
ssm_ellipse_data 66
ssm_growth_data 67
ssm_growth_formula 68
ssm_parameters 70
ssm_parameters_id 72
ssm_plot_circle 73
ssm_plot_contrast 75
ssm_plot_curve 76
ssm_plot_trajectory 78
ssm_score 80
ssm_sem 82
ssm_sem_parameters 87
ssm_sem_syntax 89
ssm_table 91
ssm_trajectory 92
summary.circumplex_axes_reliability 95
summary.circumplex_ci_accuracy 96
summary.circumplex_cpm 97
summary.circumplex_ssm_id 97
summary.circumplex_structure 99
theme_circumplex 99

Index **101**

`anchors` *Display the anchors of a circumplex instrument*

Description

Display the anchors of a circumplex instrument including the total number of anchors and each anchor's numerical value and text label. Anchors are the response options that respondants select from (e.g., 0 = No, 1 = Yes).

Usage

```
anchors(x)
```

Arguments

`x`	Required. An object of the instrument class.

Value

The same input object. Prints text to console.

See Also

Other instrument functions: `instruments()`, `items()`, `norms()`, `scales()`

Examples

```
anchors(csip)
```

`angle_unwrap` *Unwrap a sequence of angles onto a continuous branch*

Description

Unwrap a temporally ordered sequence of angular displacements (e.g., one displacement per measurement wave) onto a continuous numeric branch, so that a trajectory drifting across the 0/360 boundary becomes a smooth sequence suitable for linear growth modeling. Each input is first wrapped to [0, 360) (any real numbers are accepted); the output then starts at the first wave's wrapped value and accumulates the shortest signed rotation between successive waves, so successive values never differ by more than 180 degrees. For example, `c(350, 10, 30)` unwraps to `c(350, 370, 390)`.

Usage

```
angle_unwrap(x)
```

Arguments

`x`	A numeric vector of angles in degrees, in temporal order. Any real values are accepted and are wrapped to [0, 360) first.

Details

Two conventions are pinned. An exact 180-degree step is directionally ambiguous; it is resolved as +180 (ascending), matching the package's contrast convention of reporting an exact half-turn as +180. A missing wave makes every subsequent step branch-ambiguous, so `NA` propagates from the missing wave onward rather than silently bridging the gap.

Unwrapping assumes the sequence really does move by less than a half-turn between successive waves; when the truth moves faster than the sampling (or persons occupy heterogeneous locations with no common branch), the unwrapped branch is wrong without warning. See the package's growth modeling vignette for these failure modes and the bivariate (x, y) alternative that avoids them.

Value

A plain numeric vector of the same length: the unwrapped angles in degrees on a continuous branch anchored at the first wave's wrapped value. Values may legitimately fall outside [0, 360); the LM = 360 reporting convention applies to displacements, not to the unwrapped branch (an input of 360 anchors at 0).

Examples

```
angle_unwrap(c(350, 10, 30))
angle_unwrap(c(10, 350, 330))
angle_unwrap(c(350, NA, 30))
```

---

aw2009 *Standardized octant scores on hypothetical circumplex scales*

Description

A small example dataset containing standardized scores on eight hypothetical circumplex scales. Taken from Wright, Pincus, Conroy, & Hilsenroth (2009).

Usage

```
aw2009
```

Format

A data frame with 5 observations and 8 variables:

PA circumplex scale at 90 degrees

BC circumplex scale at 135 degrees

DE circumplex scale at 180 degrees

FG circumplex scale at 225 degrees

HI circumplex scale at 270 degrees

JK circumplex scale at 315 degrees

LM circumplex scale at 360 degrees

NO circumplex scale at 45 degrees

Source

doi:10.1080/00223890902935696

`axes_reliability` *Reliability of the circumplex axes (Strack, Jacobs & Grosse Holtforth, 2013)*

Description

Estimate the reliability (and standard error of measurement) of the two circumplex axes of an instrument with the item-level restricted tau-equivalent CFA of Strack, Jacobs, and Grosse Holtforth (2013). The model decomposes each item's variance into orthogonal components – a general factor, the two circumplex axes, scale specificity, block specificity for a blockwise instrument, and item specificity – and reads the axes' reliability off the isolated axes-variance component with the Spearman-Brown formula. It is a confirmatory, item-level complement to `fit_structure()`'s exploratory scale-level criteria.

Usage

```
axes_reliability(
  data = NULL,
  items,
  angles = NULL,
  instrument = NULL,
  cormat = NULL,
  n = NULL,
  blocks = NULL,
  sd = "std",
  missing = c("listwise", "fiml")
)
```

Arguments

`data`	A data frame (or matrix) containing the circumplex items. Supply exactly one of `data` or `cormat`.
`items`	Item selection. With `instrument`, a character vector of column names (or numeric indices) giving **all** items in item-number order, as in `score()`. Without `instrument`, a list with one element per scale, each a character vector (or numeric indices) of that scale's item columns. In either form, `items` must not select one column twice (in the list form, within a scale or across two scales), hold a number below 1, or name a column whose name occurs more than once in `data` (or in `cormat`). Items are matched by name, so a number must not reach such a column either.
`angles`	A numeric vector of the scales' angles in degrees (one per scale), required for the explicit map and forbidden with `instrument` (which supplies its own). Must be equally spaced around the circle, at any rotation, with at least four scales; `octants()` gives the canonical eight. Angles outside `[0, 360)` are reduced onto their circumplex positions, so 0 and 360 name the same position.

`instrument` Optional. A `circumplex_instrument` object supplying the scale angles and item membership (`Scales$Angle`, `Scales$Items`).

`cormat` An item correlation matrix (the matrix-input path), symmetric with a unit diagonal and positive definite, with dimnames naming the items. Supply exactly one of `data` or `cormat`.

`n` For the `cormat` path, the sample size (number of observations) the correlation matrix was computed from. Required with `cormat`, and not accepted with `data` (which carries its own).

`blocks` Optional. For a **blockwise** instrument, a list with one element per administration block, each a character vector (or numeric indices) of that block's item columns – the same shape `items` takes for scales. The blocks must partition the items: every item in exactly one block. Supplying them adds the block-specificity component to the model; see "Blockwise instruments" below.

`sd` The scale for the standard error of measurement: `"std"` (the default) reports the z-standardized SEm `sqrt(1 - reliability)`; `"raw"` uses each axis composite's observed raw SD; or a numeric vector (length 1, recycled, or length 2 for the X and Y axes) of axis SDs. A supplied numeric SD must be finite and positive; anything else is refused rather than carried into the reported SEm.

`missing` How item-level missing data are handled on the `data` path: `"listwise"` (the default; complete cases only) or `"fiml"` (full-information maximum likelihood, via lavaan's `missing = "ml"`), which uses every respondent who answered at least one item. Not available with `cormat`, which carries no missing cells.

Details

The model is fit to the item **correlation** matrix (the items are z-standardized) as a flat fixed-links CFA: every item loads on the two axes with fixed cosine weights, on a general factor with weight one, on its scale's specificity factor with weight one, and – when `blocks` are supplied and identified – on its block's specificity factor with weight one; the two axis variances are held equal (the circumplex "no preferred rotation" axiom), every scale-specificity variance shares one value and every block-specificity variance shares one value, while item errors stay free (tau-equivalent). Only the axes-variance component feeds reliability.

The Nunnally-Bernstein axis reliability (`nb_reliability`) is reported alongside for comparison: it **overestimates** axis reliability when scale specificity is large, because it charges scale-specificity variance to the axis rather than isolating it (Strack et al. 2013, Figure 3). It needs each scale's coefficient alpha, which is undefined for a scale carrying a single item, so it is reported as `NA` with a stated reason whenever any scale has fewer than two items – as Strack et al. themselves do, leaving it blank for such instruments.

Because the model is fit to the item **correlation** matrix as if it were a covariance matrix (the paper's own practice), the component point estimates and the reliabilities are correct, and both the component standard errors and the global test statistic are **corrected** for that metric (Cudeck, 1989; Satorra & Bentler, 1994). Results are reported **per axis** (X and Y): for a balanced instrument the two axes carry the same axes-variance estimate and differ only through `item_n`.

What the correction does. Normal-theory maximum likelihood prices its standard errors for a sample **covariance** input, while this estimator consumes a sample **correlation** matrix, whose diagonal cannot vary at all and whose off-diagonal cells are less variable than the corresponding covariances.

Left uncorrected, that mismatch **overstates** sampling variability by about 40% for an instrument whose axes carry a lot of variance (an axes variance of .35), and **understates** it slightly for weak-axes, strong-general instruments – so it could not be stated honestly by any fixed caveat, because it changes sign across the range of instruments this function accepts. The reported SEs are therefore adjusted to the correlation metric and are calibrated uncertainty, not order-of-magnitude guidance. They are typically **smaller** than the standard errors printed in Strack et al. (2013), whose LISREL values carry the uncorrected approximation. Point estimates, reliabilities, and SEm are unchanged by the correction. What lavaan reported before it is kept in `details$se_uncorrected`.

The global test statistic carries the same mismatch in the other direction, and is corrected too. Left alone it is too **small** – fit is flattered, because the sample correlations the estimator consumes vary less than the covariances the reference chi-square is derived for. `chisq`, `pvalue`, `rmsea` and `cfi` are therefore reported as Satorra-Bentler-type **scaled** values, `chisq` divided by a factor computed at the fitted matrix (Satorra & Bentler, 1994), with `cfi` additionally scaling its own baseline model. The factor is not a constant: it is recomputed for every fit, which is the point – the size of the distortion depends on the instrument. `df` and `srmr` are **unchanged**, being a count of restrictions and a residual summary rather than test statistics with a reference distribution. What lavaan reported before the scaling is kept in `details$fit_uncorrected`, and the two factors in `details$scaling_factor`.

One thing the scaling is **not**: it is not a robustness correction for non-normal data. The factor is computed under normal theory throughout, and it corrects one thing only – that the estimator consumes a sample correlation matrix where normal-theory maximum likelihood prices a sample covariance matrix. It does not license the model against skewed or heavy-tailed items, and it is unrelated to the Satorra-Bentler scaled statistics reported by `ssm_sem()`'s robust estimators, which correct for non-normality on the covariance metric.

The scaled statistic matches its reference chi-square in **mean**; it is not exact, and it does not make a badly misspecified model fit. If the factor cannot be computed, all four are `NA` with the reason in `details$fit_scaling_failed` – never the uncorrected value in their place. One refusal is shared with the component-SE correction, under one stated criterion evaluated in the metric the reported numbers are computed in: a fitted covariance matrix whose correlation form `cov2cor()` is degenerate is refused by both surfaces. Which degeneracies those are is settled in two steps. A matrix whose smallest eigenvalue, relative to its largest, falls at or below `sqrt(p * .Machine$double.eps / 1e-5)` is not refused for that alone; it is *checked*. The check replays this fit's own arithmetic in roughly 31-digit precision and estimates the relative error the numbers it produced actually carry – three of them: the corrected standard errors, the factor the scaled statistics are divided by, and the ratio that multiplies the reported standard error on the `missing = "fiml"` path. The fit computes whenever the WORST of those three is within the accuracy target `1e-4`. Of the reachable geometries measured below the floor, all but one committed counterexample estimate around `1e-11` and compute. A fit whose worst estimate exceeds the target, or that the check cannot price at all, is refused as `"uncertified"`, and its warning names that same worst estimate. At a severely ill-conditioned fit that estimate is a property of the fit as computed on the machine that ran it, not of the data alone: the numbers being checked are dominated by rounding there. The one counterexample above shows this. It is the three-variable matrix saved in the package sources as `tests/testthat/fixtures/rb18-counterexample-b.rds`. Three variables are too few scales for `axes_reliability()` itself, so the check was run on it through the package's internal functions. Given that same matrix, the worst estimate differed between macOS (arm64, R's reference BLAS and LAPACK) and Linux (arm64, OpenBLAS), and both machines refused it as `"uncertified"`. The standard-error, scaling-factor and FIML-ratio estimates are read as one because both surfaces refuse as a unit, so a fit on any path can be refused on the FIML ratio's estimate even where that ratio is not part of what it reports. That `1e-5` in the floor is not itself the tolerance: it is the accuracy

target 1e-4, the largest relative error a reported standard error may carry, divided by the factor of 10 by which the floor's a-priori error bound may undershoot the error it stands for. The accuracy target is set from two channels that do not depend on the sample size – the resolution the standard errors are printed at, and the coverage of a nominal 95% Wald interval – and is corroborated by a third, the standard error's own sampling variability, under which a numerical error at the target is about a tenth of the statistical noise already in the number for a typical design (relative sampling coefficient a = 1/sqrt(2)) at n up to about 5e5. That endpoint scales as 1e6 * a^2, so at the least favorable geometry measured, a = 0.045, it falls to n of about 2e3 – below the n of about 1e4 typical of published circumplex correlation matrices. Above whichever endpoint the design's own coefficient sets, the guarantee is the fixed target alone, not noise dominance. The derivation and the premises it rests on are stated beside the constant in the source. The refusal says which degeneracy happened: "indefinite" when the smallest eigenvalue is decisively negative (below -lambda_max * sqrt(p * .Machine$double.eps) – beyond the fit's own numerical noise band, so it is a statement about the model, which no arithmetic check can license), "singular" when the matrix carries non-finite entries or a nonpositive fitted variance, and "uncertified" when the per-fit check could not place this fit's numbers inside the accuracy target – roundoff-level negativity, exact singularity and severe ill-conditioning all arrive here (a numerical caution). The check is also consulted at a fit the floor admits whose information matrix is nearly singular (reciprocal condition estimate below sqrt(.Machine$double.eps)); every such fit of a kind this function can produce that the package's sweep measured passed it, and a fit the floor admits but that this function refuses at its door (fewer than four scales) can be refused "uncertified" at the internal helpers where it used to compute. Either way the corrected standard errors and the four scaled statistics go NA together (each with its own warning naming that reason) rather than one surface refusing while the other silently scales. The standard-error surface additionally applies the same criterion to the raw fitted matrix, which one internal arm of its computation – the uncorrected normal-theory pricing, kept only as a diagnostic tie to lavaan's own standard errors – inverts. A matrix degenerate only in the raw metric (wildly unequal fitted variances over a well-conditioned correlation structure) refuses that internal arm alone: the reported standard errors and scaled fit statistics all compute – details$se_correction_failed and details$fit_scaling_failed are both NULL, and no warning or note fires, because every reported number is present and priced in the metric the criterion cleared – and the internal refusal is recorded, silently, in details$naive_reason under the same reason vocabulary. Under the shared criterion the two surfaces' user-facing refusals therefore agree exactly: whatever the criterion refuses for the scaled statistics it refuses for the standard errors with the same reason, and nothing the raw metric alone refuses touches either. (Each surface retains its own refusals outside the criterion – the saturated-model door below touches only the fit statistics, and either surface's internal computation can still refuse on its own.) On a unit-diagonal fitted matrix the two metrics coincide. Separately, a saturated model (df = 0) is refused as "saturated" before any scaling arithmetic runs – a refusal that touches only the four scaled statistics; the corrected standard errors still compute. (A df = 0 fit is reachable today only at the internal helpers' documented contract boundary: axes_reliability() itself refuses the three-item maps that could saturate.) df and srmr still report.

If you cross-check against lavaan, match the variant. The scaled chisq, pvalue, rmsea and cfi here are built with the definitions lavaan calls chisq.scaled, pvalue.scaled, rmsea.scaled and cfi.scaled – the mean-adjusted Satorra-Bentler forms, cfi scaling its baseline term as well as its model term. They are **not** the *.robust forms (cfi.robust, rmsea.robust), which apply the Brosseau-Liard/Savalei adjustment and give different numbers from the same inputs. And because the model is estimated with plain maximum likelihood, the scaling being applied here rather than by lavaan, fitMeasures() on an equivalent fit reports the **uncorrected** values – those in

details$fit_uncorrected – under the bare names chisq, pvalue, rmsea and cfi. It reports no *.scaled or *.robust measure at all on such a fit: lavaan supplies those only for a genuinely scaled estimator such as "MLM" or "MLR", and silently returns a shorter vector when asked for one it does not have. So a cross-check against lavaan's bare cfi will disagree with fitcfi, a request for cfi.robust will come back empty rather than disagreeing, and neither outcome is a defect. details$baseline and details$scaling_factor carry what you need to rebuild the reported values yourself.

Value

An object of class circumplex_axes_reliability with print() and summary() methods: results (one row per axis: the axes variance, item_n, reliability, SEm, Nunnally-Bernstein reliability, and boundary flag), components (the estimated variance components with SEs – four rows by default, three when scale specificity was dropped, five when block specificity was fitted), fit (global fit indices), and details (including zeta1_fitted and zeta2_fitted, whether scale and block specificity were in the model, blocks, the block labels when a block map was supplied, nb_reason, why the Nunnally-Bernstein comparison is NA, missing, which missing-data treatment lavaan actually used, n_complete, the complete-case count, min_coverage, the fewest respondents behind any item pair, se_uncorrected, the component standard errors as normal-theory maximum likelihood reports them before the correlation-structure correction, se_correction_failed, NULL when that correction succeeded or a string naming why the reported SEs are NA – notably the shared degeneracy criterion's two literals (evaluated on cov2cor() of the fitted covariance matrix; the eigenvalue floor sqrt(p * .Machine$double.eps / 1e-5) decides which fits are *checked* by the per-fit accuracy check rather than which are refused (a fit the floor admits is also checked when its information matrix's reciprocal condition estimate falls below sqrt(.Machine$double.eps), a near-singularity in the derivative structure the floor cannot see), where the 1e-5 is the 1e-4 accuracy target divided by the floor's factor-of-10 calibration ceiling, and the target's noise-dominance reading is calibrated for n up to about 5e5 at a typical design (a = 1/sqrt(2)) and only to about 2e3 at the least favorable geometry measured (a = 0.045)): "indefinite" for a decisively negative smallest eigenvalue (below -lambda_max * sqrt(p * .Machine$double.eps)), a statement about the model, and "uncertified" where the check could not place this fit's numbers inside the accuracy target, a numerical caution whose warning names the estimated relative error – which also sets fit_scaling_failed when the shared criterion is what tripped it – naive_reason, NULL unless the internal uncorrected arm (the diagnostic tie to lavaan's own standard errors) was refused while every reported number computed, whether by the same criterion evaluated on the raw fitted matrix or by that arm's own pricing ("singular", "unidentified", "indefinite", and "ill_conditioned" – the raw arm is refused by the floor itself and never reaches the per-fit check, so this is the one field on which that literal still appears, and it inverts at R's default solve() tolerance, so at a fitted matrix whose information matrix sits near machine precision this field can differ across platforms while every reported field does not); it is deliberately silent – no warning or printed note accompanies it – fit_uncorrected, the six fit statistics as lavaan reports them before the correlation-metric scaling, scaling_factor, the two Satorra-Bentler factors (model and baseline), and fit_scaling_failed, NULL when the scaling succeeded or a string naming why chisq, pvalue, rmsea and cfi are NA). details also carries n_moments, the number of distinct analyzed moments $p^* = p(p+1)/2$, and baseline, the independence model's **unscaled** chisq and df. Those two, with fit$chisq, fit$df and the baseline element of scaling_factor – five inputs, since the baseline chi-square must be scaled by its own factor before it is used – reproduce the reported cfi. Note that details$baseline and the baseline element of details$scaling_factor are different quantities that share a name: the first is a chi-square and df pair, the second a scaling

factor. `fit` carries `chisq`, `df`, `pvalue`, `rmsea`, `cfi` and `srmr`; the four chi-square-derived values are scaled and `df` and `srmr` are not. Three sample sizes sit beside each other in `details` and are not interchangeable. `n` is the one the fit was priced at – the number of rows the estimator was actually handed, after listwise deletion or after dropping rows with no observed item, and the `n` you supplied on the correlation-matrix path. It is the N to divide `n_moments` by when locating a fit on the calibration table in `vignette("axes-reliability-caveats")`. `n_total` is the number of rows supplied before any of that, and `n_complete` the number answering every item. `n_complete` and `min_coverage` are present on every path so that a caller can read them unconditionally, and are `NA` where they carry no information: `min_coverage` outside `missing = "fiml"`, and both of them when a correlation matrix was supplied in place of raw data.

How well calibrated is the test, and at what sample size

The scaling fixes the metric error, and the χ^2 test built on it is asymptotically exact: its rejection rate approaches the nominal α as the number of distinct moments $p^* = p(p+1)/2$ falls relative to N. Measured by simulation at one population (8 octant scales, 3 items each, axes variance .35), the rejection rate at α = .05 runs .092, .079, .062, .054 at p^*/N = 0.50, 0.25, 0.12, 0.06 – reaching the nominal band by p^*/N of about 0.06. That is a sweep at a single population, not a general threshold.

At **N = 600** the χ^2 test **over-rejects**: measured .06 to .11 at three populations chosen to bracket the range of instruments this function accepts. The uncorrected statistic under-rejects over the same range, at .02 to .03, and – unlike the scaled one – moves *further* from nominal as N grows, because its error is asymptotic while the scaled statistic's is a finite-sample one that shrinks away.

The over-rejection at a fixed N grows with instrument size (larger `df`) and shrinks with N. So a p-value near whatever threshold you are using deserves caution at moderate N and a large item count – but note the direction: the scaled test **over-flags** misfit rather than flattering it, which is the safer error and the opposite of what the uncorrected statistic did.

All of that evidence is **complete-data**. Under `missing = "fiml"` the scaled statistic is calibrated in mean, but its rejection rate has not been measured, so none of the rates above should be read as applying to that path.

A related detail, in case you check: the fitted model does **not** reproduce the correlation matrix's unit diagonal exactly, and that is expected rather than a defect. With the loadings fixed, the stationarity condition available for a free item error is the *weighted* diagonal, not the raw one, so off-diagonal sampling misfit leaks into the implied diagonal at roughly the sampling standard error of a correlation.

Which instruments this accepts

Any set of **equally spaced** scale angles, at any rotation: the canonical octants, an interstitial set rotated 22.5 degrees off the axes, or a non-octant count such as six or twelve scales. What matters is equal spacing, not the count or the starting angle – for any equally spaced set of `k` scales, each axis draws the same effective test length (`k / 2` per item), which is what keeps the equal-axis-variance restriction as innocuous as it is for octants.

Scales may carry **one item each**, as Strack et al.'s types e and f do. With a single item at every position no two items share a scale, so the scale-specificity component is not identified and is dropped from the model rather than estimated: the components table then has three rows instead of four, and `details$zeta1_fitted` is `FALSE`. A *mixed* instrument still estimates it – one multi-item scale supplies the information, and the shared-value restriction carries it to the rest.

Two limits. At least **four** scales are required: with three, every pair of scales sits the same angular distance apart, and the general, axes, and scale-specificity variances are then not separately identified. And spacing must be equal, not merely close – a quasi-circumplex is refused rather than approximated, since Strack et al. (2013) excluded such instruments from the model's validation. Every scale needs at least one item.

The model is two-dimensional. Instruments whose items span three dimensions – spherical designs such as SYMLOG (Strack et al.'s type f) – are out of scope, even though Strack et al. (2013) analyze one; their Table 3 SYMLOG rows arise from a three-axis sphere model, not from any configuration this function accepts.

Missing data

`missing = "listwise"` is the default: only complete cases are used, and a message reports how many there were. Pairwise-deletion correlations are never used on either setting.

`missing = "fiml"` instead estimates from every respondent who answered at least one item, by full-information maximum likelihood. Two assumptions come with it, and both are stronger than listwise deletion's: the data must be **missing at random** (missingness may depend on values you observed, but not on the unobserved values themselves) **and multivariate normal**. Under MCAR — the special case where missingness is unrelated to anything — listwise deletion is *consistent*, merely inefficient, so FIML buys precision there and not correctness. Under MAR listwise deletion is genuinely biased and FIML is not, which is when the switch is worth its assumptions.

The items are standardized by the saturated model's own FIML means and SDs, never by the means and SDs of whichever cells happen to be observed, and those standardized columns feed a single FIML fit. The reported standard errors are observed-information standard errors on that standardized metric, conditional on the standardization constants (they do not propagate the uncertainty in the constants themselves). They carry the same correlation-metric correction as every other path, applied multiplicatively so that the observed information's own pricing of the missing data survives it. What the correction does not reach is the uncertainty in the standardization constants above. At mild rates that residual is too small to pin down: at 2%, 5%, and 10% cellwise missingness it measures 0.1%, 0.8%, and 1.8%, all well inside the Monte-Carlo error of the comparison itself (about 3.6% over 200 replicates), so its size is bounded but its direction at those rates is not established.

It becomes measurable, and **anti-conservative**, as missingness grows. Over 201 replicates at 15% cellwise MCAR the reported standard errors average about 7% **below** the estimator's actual sampling variability, so at that rate a confidence interval built from them is slightly too narrow. Note the direction reverses: the mild-rate figures above, such as they are, sit on the conservative side. Treat heavy missingness as the regime where these standard errors are least trustworthy, and prefer a resampling interval there if the uncertainty matters to your conclusion.

The global fit statistics are scaled on this path too, by the same factor the complete-data paths use rather than one rebuilt from the FIML fit's own saturated stage. That is deliberate and follows the standard errors above: lavaan's FIML chi-square is already referenced against the FIML saturated loglikelihood, so it already prices the missing information, while the scaling factor's normal-theory reference is exactly 1 – which makes the factor a metric-only ratio. A factor that priced missingness a second time would double-count it.

Two results are unavailable under `missing = "fiml"`, both because they need items observed by every respondent: the Nunnally-Bernstein comparison is `NA` with a stated reason, and `sd = "raw"` is refused — supply the axis SDs numerically instead.

A note on provenance: Strack et al. (2013) report no missing-data analyses, so nothing about the FIML path rests on their results. It is certified against this package's own synthetic oracle, where the true variance components are known by construction.

Boundary solutions

This contract governs every input path – raw items on either `missing` setting, and a supplied correlation matrix alike. A boundary fit returns `NA` reliability and SEm with a warning and a boundary flag rather than a clipped, negative, or missing value. A fit counts as a boundary when the estimated axes variance falls outside `(0, 1)` – at or below zero the axes carry no variance to be reliable, and at or above one they carry all of it, which drives the Spearman-Brown reliability to one or beyond, leaving the standard error of measurement at zero or undefined – or when any estimated variance is negative.

Supplying a correlation matrix instead of raw data

Give `cormat` and `n` in place of `data` to estimate from an item correlation matrix that someone else published, with no raw data in hand. The matrix must be symmetric, positive definite, and carry a unit diagonal (the model assumes unit-variance items); `items` selects and orders its rows by name, so the matrix's own column order does not matter. Estimates are identical to those the raw-data path would give for the same matrix.

Two results are unavailable on this path, because both need the respondents' own item scores rather than their correlations: the Nunnally-Bernstein comparison is reported as `NA` (it needs each scale's alpha and the axis composite's variance), and `sd = "raw"` is refused (there are no scale scores to take an observed SD from). Supply the axis SDs numerically if you want SEm on a raw scale.

Blockwise instruments

Some circumplex instruments are administered in **blocks** – items grouped by something other than their scale – which carries a block-specificity variance of its own (Strack et al. 2013 report it as high as 6.7%). Supply `blocks` to estimate it as a fifth component: the `components` table then carries a `zeta2` row and `details$zeta2_fitted` is `TRUE`. The package's instrument objects record no block structure, so the map comes from you.

Block specificity is estimable only when the blocks are not a relabelling of something the model already has. If every block coincides with a scale, or all items share one block, or every item sits in its own block, the component explains nothing the others do not; it is dropped from the model, `details$zeta2_fitted` is `FALSE`, and the component table keeps its four rows. That decision is read off the data's own moment structure rather than from a rule of thumb about how the blocks look, so it also catches maps whose redundancy is not obvious by eye.

What omitting `blocks` costs depends on the block geometry, and it is not a uniform penalty. The general factor never gives block variance back, so `xi2` is inflated under most layouts and unchanged under a few; it is never deflated. The axes variance – the one reliability is read from – moves only when block membership carries information about the angular distance between items, over and above what sharing a scale already says.

The clean case is worth stating exactly, because it is both common and checkable: **when each block draws exactly one item from every scale**, every within-block pair is a different-scale pair and the blocks span every pair of scale positions equally often. Block membership then says nothing about

angular distance, and `xi1`, the reliability, and the SEm are unaffected – the component is worth estimating for its own sake, but ignoring it costs the reliability nothing.

Away from that case the bias runs in either direction and **"the blocks are spread evenly around the circle" is not the test.** Blocks that pair diametrically opposite scales are as dispersed as a block can be – their angles average to the centre of the circle – and at eight scales they still pull `xi1` about 9% below truth, because every within-block pair sits half a turn apart and that is emphatically information about angular distance. Blocks covering contiguous arcs pull it the other way, about 12% above. When the blocks are neither one item per scale nor obviously arbitrary, estimate the component rather than reasoning about the geometry.

References

Strack, S., Jacobs, K. A., & Grosse Holtforth, M. (2013). The reliability of circumplex axes. *SAGE Open*, 3(2). doi:10.1177/2158244013486115

Cudeck, R. (1989). Analysis of correlation matrices using covariance structure models. *Psychological Bulletin*, 105(2), 317-327.

Satorra, A., & Bentler, P. M. (1994). Corrections to test statistics and standard errors in covariance structure analysis. In *Latent variables analysis: Applications for developmental research* (pp. 399-419).

See Also

`fit_structure()` for exploratory circumplex-structure criteria.

Examples

```
# A simulated 32-item octant dataset (four items per octant scale).
data("simulated_items")

# Map the item columns to their eight scales (four items each), in the
# octants() angle order, then estimate the axes reliability.
items <- split(names(simulated_items), rep(1:8, each = 4))
res <- axes_reliability(simulated_items, items = items, angles = octants())
res
summary(res)

# The same estimates from the item correlation matrix alone, as when
# reanalyzing a matrix published without its raw data.
axes_reliability(
  cormat = cor(simulated_items), items = items, angles = octants(),
  n = nrow(simulated_items)
)
```

`coord_circumplex` *Circumplex coordinate system*

Description

A **ggplot2** coordinate system that maps Structural Summary Method parameters onto the circular circumplex canvas: the `displacement` aesthetic (degrees, counterclockwise from the right, with the 0/360 pole labelled 360) becomes the angle and the `amplitude` aesthetic becomes the radius. It owns the amplitude-to-radius scaling, so `geom_ssm_point()` and `geom_ssm_arc()` no longer take an `amax`, and the canvas and data layers can never disagree.

Usage

```
coord_circumplex(
  amax = NULL,
  center = 0,
  r_axis_angle = NULL,
  grid = c("polar", "cartesian"),
  ...
)
```

Arguments

`amax`	Optional. A single positive number giving the amplitude represented by the outer ring. `NULL` (the default) trains it from the data (as `ssm_plot_circle()` does).
`center`	Optional. A single number giving the amplitude at the center of the circle (default = 0). Ring labels and the amplitude-to-radius mapping are guaranteed to agree.
`r_axis_angle`	Optional. A single number giving the displacement (in degrees) along which the amplitude (radial) axis and its labels are drawn. `NULL` (the default) places it automatically in the widest gap between the displacement spokes, so the amplitude labels never collide with a spoke label. Ignored when `grid = "cartesian"`, which draws no amplitude axis.
`grid`	Optional. A single string naming the canvas furniture. `"polar"` (the default) draws amplitude rings, displacement spokes and the amplitude axis. `"cartesian"` draws one ring at the outer amplitude, a crosshair along displacements 0/180 and 90/270, and a tick mark with a signed label at each amplitude break between the center and the rim on every half-axis (the center and the rim value itself are not labelled), and no other ring, spoke or amplitude axis (the canvas of Nagy et al., 2019). The labels on the 180 and 270 halves are negative Cartesian coordinates, not negative amplitudes. The mapping of the data onto the canvas is the same in both modes.
`...`	Reserved for future extensions; currently unused.

Details

coord_circumplex() subclasses ggplot2::coord_radial() and hard-pins the angular convention (displacement 0 at the right, increasing counterclockwise, the 0/360 range with no expansion) so the circumplex angle invariants survive the transform. The amplitude at the circle's center and at its outer ring are the radial limits, set once here.

Value

A **ggplot2** coordinate system that can be added to a plot with +.

See Also

Other circumplex layers: geom_ssm_arc(), geom_ssm_ellipse(), geom_ssm_path(), geom_ssm_point(), ggcircumplex(), scale_x_circumplex(), theme_circumplex()

Examples

```
data("jz2017")
res <- ssm_analyze(jz2017, scales = 2:9, measures = "NARPD")
ggplot2::ggplot(res$results) +
  coord_circumplex(amax = 0.5) +
  geom_ssm_point(ggplot2::aes(amplitude = a_est, displacement = d_est))

# A Cartesian grid: one rim ring and a labelled crosshair, no rings or spokes
ggplot2::ggplot(res$results) +
  coord_circumplex(amax = 0.5, grid = "cartesian") +
  geom_ssm_point(ggplot2::aes(amplitude = a_est, displacement = d_est))
```

---

cpm_fit *Fit Browne's circular stochastic process model (circumplex fit statistics)*

Description

Estimate Browne's (1992) circular stochastic process model (CPM) for the correlational structure of a set of circumplex scales or items, the native replacement for the archived CircE package. Each variable is modeled as a point on a circle at an estimated angle, with a communality index and a shared correlation function; the fit of that structure is summarized with the usual covariance-structure indices (chi-square, RMSEA, SRMR, CFI, TLI).

Usage

```
cpm_fit(
  data = NULL,
  scales = NULL,
  angles = octants(),
  cormat = NULL,
  n = NULL,
```

```
  m = 3,
 model = c("quasi-circumplex", "constrained-angles", "equal-communality", "circulant"),
  scaling = c("unit", "free"),
  reference = 1,
  interval = 0.95,
  ci_method = c("bootstrap", "analytic"),
  boots = 2000,
  listwise = TRUE,
  check_lower = TRUE
)
```

Arguments

data	A data frame or matrix containing the circumplex scales (raw-data path). Supply exactly one of data or cormat.
scales	For the raw-data path, a character vector of column names (or a numeric vector of column indexes) selecting the circumplex scales; it must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in data. For the cormat path, optional labels for the variables (defaults to the matrix dimnames, or V1, V2, ...); each label, given or taken from the dimnames, must appear only once, and NA at most once.
angles	A numeric vector of the theoretical angular displacement of each scale, in degrees, used both as the reference/identifying angle and as optimization start values (default = octants()). Its length must match the number of scales.
cormat	A correlation matrix (the matrix-input path, CircE-style). Supply exactly one of data or cormat. Must be symmetric with a unit diagonal and positive definite.
n	For the cormat path, the sample size (number of observations) the correlation matrix was computed from. The test statistic uses N - 1 (the Wishart degrees of freedom); pass the raw sample size here.
m	The number of harmonics in the correlation function (default = 3, the octant-scale convention). Capped at floor((p - 1) / 2) for the free-angle variants and floor(p / 2) for the fixed-angle variants.
model	The model variant (design of Browne 1992): "quasi-circumplex" (default; free angles and communalities), "constrained-angles" (angles fixed at their theoretical values), "equal-communality" (a single shared communality), or "circulant" (both constraints).
scaling	The covariance-scaling family, orthogonal to model: "unit" (default) fits the correlation structure with a unit model-implied diagonal (the historical behaviour); "free" fits Browne's covariance structure $\Sigma = D_\sigma P D_\sigma$ with p free variance scales, the parameterization CIRCUM and CircE use, so cpm_fit() can reproduce their published output exactly. Free scaling adds p parameters but leaves the degrees of freedom unchanged (it fits the p extra diagonal covariance moments). For the model test the two families are calibration-indistinguishable at correlation input — in paired simulation at the measured truths (n from 250 to 50,000) their test statistics differed by well under 1 percent of the degrees of freedom — so neither family's chi-square is more trustworthy than the other's.

(The two statistics are close but not interchangeable digit-for-digit: the free family nests the unit family, and its optimizer is additionally started from the unit solution, so on the same input the free statistic never exceeds the unit statistic beyond the engine's numerical tolerance — fits whose boundary harmonics are polished away are compared on their own reduced model.) `"unit"` remains the default because free scaling buys no inferential benefit for correlation input while adding `p` parameters whose analytic standard errors are frequently undefined below `n = 2000`; choose `"free"` when the goal is reproducing published CIRCUM/CircE results. The fitted $\hat{\sigma}^2$ are reported as a `VarRatio` column (the ratio of reproduced to input variance); they carry no confidence interval. The analytic intervals for the remaining parameters follow the same sample-size caution as the default family: their coverage regime was measured (the free-family coverage oracle, on fits with a defined standard error) to match it, so `summary()` cautions below `n = 2000`, and from `n = 2000` up to but not including `n = 50000` in fits that show a boundary or weak-identification marker. The input is still a correlation matrix (unit diagonal).

`reference` The index into `scales` of the variable whose angle is fixed at its theoretical value to identify the rotation (default = 1).

`interval` The confidence level for the parameter intervals (default = 0.95). The RMSEA interval is always the conventional 90 percent.

`ci_method` How to construct the parameter confidence intervals: `"bootstrap"` (the default on the raw-data path) resamples rows, recomputes the correlation matrix, and refits the model warm-started from the reported solution; `"analytic"` uses Wald intervals from the information matrix. On the `cormat` path only `"analytic"` is available (there is no raw data to resample), and it is the default there.

`boots` The number of bootstrap resamples for `ci_method = "bootstrap"` (default = 2000).

`listwise` Whether to handle missing values by listwise deletion. Only listwise deletion is supported in this release (default = TRUE).

`check_lower` Whether to run the extra starts described in "Which solution is reported" and note a lower discrepancy they reach (default = TRUE). The check runs only for `model = "quasi-circumplex"` with `scaling = "unit"`, and only on the fit itself, never on bootstrap resamples. Its cost grows with about the square of the number of scales, as the number of extra starts does, so set it to FALSE to skip it when fitting many models.

Value

A `circumplex_cpm` object: a list with `results` (a data frame of estimated angles and communality indices with confidence intervals), `betas` (the correlation-function weights), `fit` (the fit indices), `corfun` (the estimated correlation function), `matrices` (the sample and model-implied matrices and residuals), and `details` (model, diagnostics, and settings). See `print.circumplex_cpm()` and `summary.circumplex_cpm()`.

Which solution is reported

The engine minimizes the discrepancy function from a fixed set of starts. For the free-angle models (`"quasi-circumplex"` and `"equal-communality"`), the starts are the theoretical `angles`, their

mirror image about the reference scale, and four fixed patterns of 15- and 30-degree offsets from them. For the fixed-angle models (`"constrained-angles"` and `"circulant"`), the angles stay at their theoretical values, and the starts are one start and three fixed rescalings of its communality indices. In both cases the communality-index and weight starts are computed from the sample correlation matrix. With `scaling = "free"`, one more run starts from the `"unit"` solution. The estimate is the solution with the lowest discrepancy that these runs reach: the maximum-likelihood solution in the basin they reach. (When a correlation-function weight of that solution is near zero, the harmonic can be removed and the reduced model, which has more degrees of freedom, is refit from it and reported instead.)

That solution is not always the global minimum. A configuration with a lower discrepancy can exist that these runs do not reach, for example with more scales at or near a communality index of 1 or with scales in other positions on the circle. Two checks compare these runs only, so neither of them signals such a configuration: the reproduction part of the convergence acceptance criterion, which requires at least two independent starts to reach the lowest discrepancy, and the "competing near-tied optima" note. The lower-discrepancy check below is the one that can. Where one exists, the reported chi-square statistic is larger than at the lower discrepancy. At the same degrees of freedom, its p-value is then smaller and its RMSEA and CFI are no better.

With `check_lower = TRUE` (the default), a fixed set of extra runs then follows, most of them from the reported solution: with each communality index in turn, and then all of them, set near the upper bound (0.99); with each free angle shifted by 90, -90 and 180 degrees; with each pair of free angles swapped, alone and with either swapped scale's communality index near the bound; and, from the theoretical start, with each communality index in turn near the bound. If one of these runs reaches a discrepancy lower than the reported one by more than 0.01 on the chi-square scale, `print()` and `summary()` print a note. It gives that difference, the degrees of freedom of the lower solution when they differ, whether the lower solution keeps the scales' order around the circle, and how many more or fewer communality indices it has above 0.995. The reported solution, its fit indices and its confidence intervals do not change: in the package's simulations of normal data at two eight-scale truths with three harmonics, the lower solutions these runs reached estimated the true angles and weights worse on average than the reported one, though not on every fit. The extra runs do not find every lower discrepancy, so a fit with no note can still have one. If the check cannot finish (every extra run fails, or it stops with an error), a note says so, and the fit itself is unaffected. The check uses no random numbers, and it runs only for `model = "quasi-circumplex"` with `scaling = "unit"` on a fit that meets the convergence acceptance criterion: the model, scaling and fits those simulations covered.

`details$lower_check` holds the check's record: `checked` (whether it ran), `found` (whether it found a lower discrepancy; `NA` when it stopped with an error), `gap` (the difference on the chi-square scale, after the lower solution's weight polish), `df` (the lower solution's degrees of freedom), `order_changed`, `more_wall` (the change in the number of communality indices above 0.995), `failed_starts`, `n_starts` and `error`.

Confidence intervals

The bootstrap (the raw-data default) refits the model to each resampled correlation matrix, warm-started from the reported solution, and forms percentile intervals; angle replicates are pooled with the package's circular quantile machinery, so an angle interval that straddles the 0/360 boundary is reported wrapped (its lower limit numerically exceeds its upper limit, as with displacement intervals in `ssm_analyze()`). Resamples with a degenerate (NA or non-positive-definite) correlation

matrix, or whose refit fails the convergence acceptance criterion or stops with an error, are excluded with a warning reporting how many; the intervals are then conditional on estimability. The count of refits failing the criterion, also stored as `details$boots_nonconvergent`, includes refits that stopped with an error. Analytic (Wald) intervals are asymptotically valid but can materially mis-cover at field-typical sample sizes. For `ci_method = "analytic"`, `summary()` prints a caution whenever the sample size N is below 2000, and for 2000 <= N < 50000 when the fit shows a boundary or weak-identification marker (Heywood communality, boundary harmonic removed, small correlation-function weight, ill-conditioned Hessian, or competing near-tied optima); at N >= 50000 it prints none. A correlation-function weight counts as small below 0.10. The coverage behind these thresholds was measured on fits with a defined analytic standard error. Analytic intervals come from the normal-theory information matrix, so they assume multivariate-normal data; on the `cormat` path, where they are the only choice, the data cannot be resampled to avoid that. Analytic angle intervals are reported on the unwrapped branch of the estimate (endpoints may fall outside [0, 360) near the boundary).

Reproducibility

Only the bootstrap consumes R's random number stream; the engine's point estimates, fit indices, and the analytic intervals are deterministic, so the estimates are identical across seeds and the default `cormat`-path fit never touches the stream. Call `set.seed()` immediately before `cpm_fit()` for reproducible bootstrap intervals. All resample indices are drawn in one block before any refitting, so a given seed yields the same intervals regardless of how many replicates are later excluded.

References

Browne, M. W. (1992). Circumplex models for correlation matrices. *Psychometrika, 57*(4), 469-497.

See Also

Other analysis functions: `cpm_simulate()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Examples

```
data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")


# Raw-data path on the eight IIP-SC octant scales (bootstrap CIs; a small
# `boots` keeps the example short -- the default is 2000)
set.seed(12345)
fit <- cpm_fit(jz2017, scales = scales, boots = 100)
fit


# Matrix-input path (supply the sample size; CIs are analytic). This fit
# reaches a boundary (a communality at its upper limit), and its Hessian is
# also ill-conditioned, so the call warns and reports no usable intervals;
# vignette("cpm-boundary-fits", package = "circumplex") explains both.
```

```
R <- cor(jz2017[scales])
cpm_fit(cormat = R, scales = scales, n = nrow(jz2017))
```

`cpm_simulate` *Simulate data from a fitted circular process model*

Description

Draw `n` standardized observations from the model-implied correlation matrix $\hat{P}$ of a fitted `cpm_fit()` object, using the low-rank factor representation $P = \Lambda\Lambda^\top + (I - D_\zeta^2)$ (Browne, 1992; design sec. 1.3/sec. 5.4). Each observation is generated as $x = \Lambda z + (I - D_\zeta^2)^{1/2}\varepsilon$ with independent standard-normal common-factor scores z and unique deviates ε, so the draws are exactly positive semidefinite by construction (no eigenvalue clamping) and their population correlation matrix is $\hat{P}$ exactly.

Usage

```
cpm_simulate(object, n)
```

Arguments

`object`	A `circumplex_cpm` object from `cpm_fit()`.
`n`	The number of observations (rows) to simulate; a positive whole number.

Details

This is the mean-based simulation path of the SSM CI-trustworthiness diagnostic (`ssm_ci_accuracy()`; the M4 Brief-B contract): a caller draws standardized data here and rescales it to a group's means and SDs. The correlation-based (augmented scales-plus-measures) path is **not** produced here – it reduces to the returned population block `object$matrices$Phat`, from which the caller assembles and repairs its own joint matrix.

Value

A numeric matrix with `n` rows and one column per fitted scale, columns in the fitted scale order with `colnames` set to the scale names (`rownames` are `NULL`). The population covariance is `object$matrices$Phat`, so `cov()` of the returned matrix converges to it as `n` grows. Under the default unit scaling `Phat` is a correlation matrix (zero-mean, unit-variance margins), so `cor()` converges to it too; under `scaling = "free"` the margins carry the fitted variance ratios $\hat{\sigma}^2$.

Reproducibility

`cpm_simulate()` consumes R's global random number stream (the common-factor scores then the unique deviates, drawn in that fixed order), so it follows the package's `set.seed()`-immediately-before convention: a given seed reproduces the draw exactly. It is one of the package's stochastic entry points (alongside `ssm_analyze()` and the bootstrap path of `cpm_fit()`); the fit itself is deterministic.

References

Browne, M. W. (1992). Circumplex models for correlation matrices. *Psychometrika, 57*(4), 469-497.

See Also

cpm_fit()

Other analysis functions: cpm_fit(), ssm_analyze(), ssm_analyze_long(), ssm_ci_accuracy(), ssm_draws(), ssm_parameters(), ssm_parameters_id(), ssm_score(), ssm_sem(), ssm_sem_parameters(), summary.circumplex_ssm_id()

Examples

```
data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
fit <- cpm_fit(cormat = cor(jz2017[scales]), scales = scales,
               n = nrow(jz2017))
set.seed(1)
x <- cpm_simulate(fit, n = 500)
round(cor(x) - fit$matrices$Phat, 2)
```

---

fit_structure *Evaluate circumplex structure (Acton & Revelle, 2004)*

Description

Run the exploratory circumplex-structure criteria of Acton and Revelle (2004) on a set of scales and return one object bundling all of the tests. Four criteria are computed from the first two unrotated principal-axis factors of the scales' correlation matrix – the **Fisher Test** of equal axes, the **Gap Test** of equal spacing, the **Variance Test** (VT2) and **Rotation Test** of interstitiality – and each statistic is classified against simulation-derived, scoring- and scale-count-specific cutoffs. A fifth test, **RANDALL** (Hubert & Arabie, 1987; Tracey, 1997), evaluates the hypothesised circular *order* of the scales with a randomization test that yields a genuine p-value.

Usage

```
fit_structure(
  data,
  scales,
  scoring = c("deviation", "raw"),
  ridge = 0,
  n_perm = NULL,
  listwise = TRUE
)
```

Arguments

`data`	A data frame (or matrix) containing the circumplex scales.
`scales`	A character vector of column names (or a numeric vector of column indexes) selecting the circumplex scales, **in hypothesised circular order** (the order is RANDALL's order hypothesis). At least four scales are required. They must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in `data`.
`scoring`	Either `"deviation"` (the default; row-mean-center the scales before analysis) or `"raw"` (analyze the scores as given). Selects the matching interpretive cutoffs.
`ridge`	A non-negative ridge added to the diagonal of the correlation matrix (then rescaled to a unit diagonal) to repair a non-positive-definite matrix before factoring; default `0`, which matches the cutoff calibration. Raise it only if factoring fails, noting that a nonzero ridge moves the statistics off the calibrated scale.
`n_perm`	`NULL` (the default) to compute RANDALL's p-value by exact enumeration, available for up to nine scales; otherwise a single positive whole number of Monte Carlo relabelings (required for ten or more scales). The Monte Carlo path draws from the global RNG stream, so set a seed with `set.seed()` beforehand for reproducibility. Ten or more scales require `n_perm` (exact enumeration is infeasible); supplying it is validated up front, before any criteria are computed.
`listwise`	A logical indicating whether missing values are handled by listwise deletion (`TRUE`, the default) or pairwise deletion (`FALSE`), matching `ssm_analyze()`. Listwise deletion gives all five tests one complete-case correlation matrix, which is the metric the interpretive cutoffs were calibrated on; pairwise deletion can yield a non-positive- definite matrix and moves the statistics off that calibrated scale.

Details

The four factor-analytic criteria have the most power when there is no large general factor, which *deviation scoring* (centering each respondent on their own mean across the scales, exactly what `ipsatize()` does) approximates by removing it (Acton & Revelle, 2004, p. 9). Deviation scoring is therefore the default and is applied to all five tests; pass `scoring = "raw"` to leave the scores untouched. The two scorings carry different cutoffs, matched automatically.

The interpretive cutoffs are **heuristic likelihood classifications read off simulated distributions, not significance tests**, and they are specific to the number of scales. Only eight scales (the canonical octant instrument) are calibrated; with any other count the statistics are still reported but no interpretation is attached (see `print()`/`summary()`). The cutoffs were re-derived under Acton and Revelle's own generating model at eight scales; see `vignette("structure-tests")`. RANDALL needs no cutoffs: with up to nine scales its null distribution is enumerated exactly, so its p-value is available at any scale count of four or more.

Value

An object of class `circumplex_structure` with `print()`, `summary()`, and `plot()` methods. Its components are `results` (a data frame with one row per factor-analytic criterion: statistic, cutoffs, and interpretive category), `randall` (the RANDALL index, p-value, and method), `loadings` (the two unrotated principal-axis factors), and `details`.

References

Acton, G. S., & Revelle, W. (2004). Evaluation of ten psychometric criteria for circumplex structure. *Methods of Psychological Research Online*, 9(1), 1-27.

Hubert, L., & Arabie, P. (1987). Evaluating order hypotheses within proximity matrices. *Psychological Bulletin*, 102(1), 172-178.

Tracey, T. J. G. (1997). RANDALL: A Microsoft FORTRAN program for a randomization test of hypothesized order relations. *Educational and Psychological Measurement*, 57(1), 164-168.

See Also

`cpm_fit()` for a confirmatory circumplex model; `ipsatize()` for deviation scoring.

Examples

```
data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
res <- fit_structure(jz2017, scales = scales)
res
summary(res)
```

---

geom_ssm_arc *Draw SSM confidence-region arcs in circumplex space*

Description

A **ggplot2** layer that draws, for each profile, the wedge spanning its amplitude confidence interval (radially) and its displacement confidence interval (angularly) on a circumplex canvas built with `coord_circumplex()` (for example the canvas from `ggcircumplex()`). The bounds are supplied directly in SSM units; the coordinate system bends the (displacement, amplitude) rectangle into an annular wedge.

Usage

```
geom_ssm_arc(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  amax = NULL,
  n = NULL,
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `stat`, `position`, `show.legend`, `inherit.aes`, `...`
: Standard **ggplot2** layer arguments. `mapping` must supply the `amplitude_min`, `amplitude_max`, `displacement_min`, and `displacement_max` aesthetics.

`amax`
: (Deprecated) The amplitude represented by the outer ring is now owned by `coord_circumplex()`; a value supplied here is ignored with a one-time note.

`n`
: (Deprecated) Arc smoothness is now owned by the coordinate system, which curves the wedge automatically; a value supplied here is ignored with a one-time note.

`na.rm`
: If `FALSE`, warn (with the dropped-row count) before removing profiles with an incomplete confidence region (a missing amplitude or displacement bound); if `TRUE` (the default) remove them silently.

Details

Each arc spans **counterclockwise** from `displacement_min` to `displacement_max` (both in degrees). Supply them in `[0, 360]` (a bound of exactly 360 is the 0/360 pole under the package's LM = 360 labeling). A `displacement_min` greater than `displacement_max` is read as an interval that crosses the 0/360 seam and is drawn the short way across it (e.g. `350 -> 10` is a 20 degree arc, matching how the package stores a displacement CI that straddles the boundary). The interval must describe less than a full circle; bounds that imply a span of 360 degrees or more (for example, values outside `[0, 360]`) are rejected, since they do not name a unique arc.

Value

A **ggplot2** layer.

See Also

Other circumplex layers: `coord_circumplex()`, `geom_ssm_ellipse()`, `geom_ssm_path()`, `geom_ssm_point()`, `ggcircumplex()`, `scale_x_circumplex()`, `theme_circumplex()`

Examples

```
data("jz2017")
res <- ssm_analyze(jz2017, scales = 2:9, measures = "NARPD")
ggcircumplex(octants(), amax = 0.5) +
  geom_ssm_arc(
    data = res$results,
    mapping = ggplot2::aes(
      amplitude_min = a_lci, amplitude_max = a_uci,
      displacement_min = d_lci, displacement_max = d_uci
    ),
    alpha = 0.4
  )
```

`geom_ssm_ellipse` *Draw a joint confidence ellipse for SSM coordinates in circumplex space*

Description

A **ggplot2** layer that draws, for each profile, the ellipse of a bivariate normal region on the Cartesian `(x, y)` coordinates of the Structural Summary Method, on a circumplex canvas built with `coord_circumplex()` (for example the canvas from `ggcircumplex()`). Each row supplies the centre `(x0, y0)` and the three elements of a 2 by 2 covariance matrix; the layer computes the ellipse's outline in Cartesian space, converts each vertex to a displacement and an amplitude, and hands those to the coordinate system, which owns the polar transform.

Usage

```
geom_ssm_ellipse(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  level = 0.95,
  n = 100,
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `stat`, `position`, `show.legend`, `inherit.aes`, `...`
: Standard **ggplot2** layer arguments. `mapping` must supply the `x0`, `y0`, `var_x`, `var_y`, and `cov_xy` aesthetics: the centre and the variances of `x` and `y` and their covariance, all in the score metric of the Cartesian coordinates.

`level`
: A single number strictly between 0 and 1: the confidence level of the region (default 0.95).

`n`
: The number of vertices on each outline (default 100), a single whole number of at least 3; the path returns to its first vertex.

`na.rm`
: If `FALSE`, warn (with the dropped-row count) before removing rows with a non-finite centre or covariance element; if `TRUE` (the default) remove them silently. A covariance that is not positive definite is an error, not a missing value.

Details

The outline is the contour `(v - c)' S^-1 (v - c) = qchisq(level, 2)`, where `c` is the centre and `S` the covariance: under a bivariate normal approximation to the distribution of `(x, y)`, sampling or

posterior, it encloses the joint region at confidence `level`. `ssm_ellipse_data()` computes the five columns from an `ssm_draws()` object. The ellipse is a statement about the Cartesian pair, not about amplitude and displacement separately; the wedge `geom_ssm_arc()` draws is the pair of marginal intervals on those two parameters, and the two regions need not coincide.

Vertices are unwrapped along the outline so that an ellipse straddling the 0/360 seam is drawn across it, and an ellipse whose inscribed polygon contains the origin winds once round the centre of the canvas. Unwrapped displacements may therefore fall outside `[0, 360)`. Each retained input row is one outline: the layer sets the `group` aesthetic to one value per retained row, replacing any `group` the mapping supplies.

Value

A **ggplot2** layer.

See Also

Other circumplex layers: `coord_circumplex()`, `geom_ssm_arc()`, `geom_ssm_path()`, `geom_ssm_point()`, `ggcircumplex()`, `scale_x_circumplex()`, `theme_circumplex()`

Examples

```
set.seed(1)
draws <- cbind(rnorm(500, 0.4, 0.1), rnorm(500, 0.3, 0.05),
               rnorm(500, -0.2, 0.05))
res <- ssm_draws(draws, type = "parameters")
ggcircumplex(octants(), amax = 0.5) +
  geom_ssm_ellipse(
    data = ssm_ellipse_data(res),
    mapping = ggplot2::aes(
      x0 = x0, y0 = y0, var_x = var_x, var_y = var_y, cov_xy = cov_xy
    )
  ) +
  geom_ssm_point(
    data = res$results,
    mapping = ggplot2::aes(amplitude = a_est, displacement = d_est)
  )
```

---

geom_ssm_path *Draw a profile's movement across occasions in circumplex space*

Description

A **ggplot2** layer that connects a profile's successive positions on a circumplex canvas built with `coord_circumplex()` (for example the canvas from `ggcircumplex()`), so change in amplitude and displacement reads as movement through circumplex space. Each segment is curved along the polar geodesic by the coordinate system, which owns the transform; the layer owns the ordering, the 0/360 seam handling, and the optional arrowheads.

Usage

```
geom_ssm_path(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  arrow = NULL,
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`, `data`, `stat`, `position`, `show.legend`, `inherit.aes`, `...`
: Standard **ggplot2** layer arguments. `mapping` must supply the `amplitude` and `displacement` aesthetics.

`arrow`
: An arrow specification produced by `ggplot2::arrow()`, or `NULL` (the default) for a path drawn without arrowheads. Arrowheads mark the direction of time along the path.

`na.rm`
: If `FALSE`, warn when occasions with no location are removed from the ends of a path; if `TRUE` (the default) remove them silently. Occasions with no location in the *middle* of a path break it either way.

Details

Points are connected in the order the rows appear in the data, exactly as `ggplot2::geom_path()` does, and the `group` aesthetic separates one series from another. Sort the data into time order before plotting: an occasion label sorted as text puts `T10` before `T2`, which silently reverses time. `ssm_plot_circle(path = TRUE)` does this sorting for you, taking the order from the object's own occasion list.

Consecutive occasions are joined the **short** way around the circle. The displacements of each group are unwrapped onto a continuous branch before the coordinate system sees them, so a step from `350` to `10` degrees is drawn as the 20 degree arc across the pole rather than a 340 degree sweep the long way round. Unwrapped values may therefore fall outside `[0, 360)`. This assumes the profile rotates less than a half-turn between consecutive occasions at which its displacement is defined; no data can verify that, so widely spaced occasions should be read with it in mind.

An occasion with no defined location – a flat or zero-amplitude profile, whose displacement is undefined – **breaks** the path rather than being interpolated through, and the segment after the gap is still drawn on the correct branch. Non-finite amplitudes and displacements are treated the same way, since an infinite angle names no position on the circle.

Value

A **ggplot2** layer.

See Also

Other circumplex layers: coord_circumplex(), geom_ssm_arc(), geom_ssm_ellipse(), geom_ssm_point(), ggcircumplex(), scale_x_circumplex(), theme_circumplex()

Examples

```
data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
# Three occasions that actually move: shifting the octant scores by one
# position rotates the fitted profile by 45 degrees per occasion.
waves <- lapply(1:3, function(k) {
  idx <- ((seq_along(scales) - 1 + (k - 1)) %% length(scales)) + 1
  d <- jz2017[, scales[idx]]
  names(d) <- scales
  d$id <- seq_len(nrow(d))
  d$occasion <- paste0("T", k)
  d
})
res <- ssm_analyze_long(do.call(rbind, waves),
  scales = scales, id = "id", occasion = "occasion"
)
ggcircumplex(octants(), amax = 0.5) +
  geom_ssm_point(
    data = res$results,
    mapping = ggplot2::aes(amplitude = a_est, displacement = d_est),
    size = 2
  ) +
  geom_ssm_path(
    data = res$results,
    mapping = ggplot2::aes(amplitude = a_est, displacement = d_est),
    arrow = ggplot2::arrow(
      length = ggplot2::unit(0.18, "inches"), type = "closed"
    )
  )
```

---

geom_ssm_point *Draw SSM profile points in circumplex space*

Description

A **ggplot2** layer that places a point for each profile at its amplitude and displacement on a circumplex canvas built with coord_circumplex() (for example the canvas from ggcircumplex()). The amplitude and displacement are supplied directly in SSM units (amplitude in the score metric, displacement in degrees); the coordinate system performs the polar transform.

Usage

```
geom_ssm_point(
  mapping = NULL,
```

```
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  amax = NULL,
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping, data, stat, position, show.legend, inherit.aes, ...	Standard **ggplot2** layer arguments. mapping must supply the amplitude and displacement aesthetics.
amax	(Deprecated) The amplitude represented by the outer ring is now owned by coord_circumplex(); a value supplied here is ignored with a one-time note.
na.rm	If FALSE, warn (with the dropped-row count) before removing profiles with a missing displacement or amplitude, since they have no location on the circle; if TRUE (the default) remove them silently.

Value

A **ggplot2** layer.

See Also

Other circumplex layers: coord_circumplex(), geom_ssm_arc(), geom_ssm_ellipse(), geom_ssm_path(), ggcircumplex(), scale_x_circumplex(), theme_circumplex()

Examples

```
data("jz2017")
res <- ssm_analyze(jz2017, scales = 2:9, measures = "NARPD")
ggcircumplex(octants(), amax = 0.5) +
  geom_ssm_point(
    data = res$results,
    mapping = ggplot2::aes(amplitude = a_est, displacement = d_est)
  )
```

---

ggcircumplex *Create a circumplex plotting canvas*

Description

Build an empty circular canvas – the amplitude rings, displacement spokes, and scale labels that circumplex figures are drawn on – as a **ggplot2** object. Additional layers (points, arcs, annotations) can be added to it with +, so it serves as the reusable foundation for custom circumplex visualizations. The package's own ssm_plot_circle() draws on the same canvas.

Usage

```
ggcircumplex(
  angles = octants(),
  labels = NULL,
  amax = 0.5,
  font_size = 12,
  instrument = NULL,
  grid = c("polar", "cartesian"),
  angle_labels = FALSE
)
```

Arguments

angles	Optional. A numeric vector of the angular position (in degrees) of each circumplex scale, going counterclockwise from the right (default = octants()). Ignored if instrument is supplied.
labels	Optional. Either NULL or a character vector of text labels to draw around the circle, one per angle and in the same order (default = NULL, which draws the numeric angles). If instrument is supplied, NULL uses the instrument's scale abbreviations.
amax	Optional. A single positive number giving the amplitude at the outer ring, which sets the amplitude-axis labels; the center of the circle is fixed at amplitude 0 (default = 0.5).
font_size	Optional. A single positive number giving the size (in pt) of the scale and amplitude labels (default = 12).
instrument	Optional. Either NULL or a circumplex_instrument object (see instrument()). When supplied, the scale angles and (unless labels is given) the scale abbreviations are taken from the instrument (default = NULL).
grid	Optional. A single string naming the canvas furniture, passed to coord_circumplex(). "polar" (the default) draws amplitude rings, displacement spokes and the amplitude axis. "cartesian" draws one rim ring, a crosshair with a tick mark and a signed label at each amplitude break between the center and the rim on every half-axis, and tick marks outward from the rim at the scale angles (the canvas of Nagy et al., 2019). The labels on the 180 and 270 halves are negative Cartesian coordinates, not negative amplitudes.
angle_labels	Optional. A single logical, independent of grid. TRUE appends each scale's angle to its label, in parentheses with the degree sign, the angle rounded to the nearest whole degree and the 0/360 pole written as 360, and turns each label to read along its radius (default = FALSE). It applies to text labels, from labels or an instrument; the default degree labels already show the angle and are left as they are. Each side of the plot margin is widened to fit the labels that point toward it: a label reaches past a side by its length, estimated at half of font_size per character, times the share of its direction that points toward that side. A side that no label points toward keeps the theme's margin, so the circle can sit off-center on the page. Add + theme(plot.margin = ...) to set the margin yourself.

Value

A **ggplot2** object containing the empty circumplex canvas.

See Also

`coord_circumplex()`, which owns the transform this canvas is built on; `ssm_plot_circle()`, which draws SSM results on this canvas.

Other circumplex layers: `coord_circumplex()`, `geom_ssm_arc()`, `geom_ssm_ellipse()`, `geom_ssm_path()`, `geom_ssm_point()`, `scale_x_circumplex()`, `theme_circumplex()`

Examples

```
# A default octant canvas
ggcircumplex()

# Label the scales with their circumplex pole abbreviations
ggcircumplex(octants(), labels = PANO())

# Derive the angles and labels from a circumplex instrument
ggcircumplex(instrument = csip)

# A Cartesian grid: one rim ring and a labelled crosshair
ggcircumplex(octants(), labels = PANO(), grid = "cartesian")

# Each scale labelled with its angle, read along the radius
ggcircumplex(octants(), labels = PANO(), grid = "cartesian", angle_labels = TRUE)
```

---

html_render *Format and render data frame as HTML table*

Description

Format a data frame as an HTML table and render it to the web viewer.

Usage

```
html_render(df, caption = NULL, align = "l", ...)
```

Arguments

`df`	A data frame to be rendered as an HTML table.
`caption`	A string to be displayed above the table.
`align`	A string indicating the alignment of the cells (default = "l").
`...`	Other arguments to pass to `htmlTable`.

Value

HTML syntax for the `df` table.

See Also

Other table functions: ssm_table()

`instruments` *List all available instruments*

Description

The circumplex package includes information about numerous circumplex instruments including instructions for scoring and standardizing items. Individual instruments can be loaded using the `instrument` function.

Usage

```
instruments()
```

See Also

Other instrument functions: anchors(), items(), norms(), scales()

Examples

```
instruments()
```

`ipsatize` *Ipsatize circumplex items using deviation scoring across variables*

Description

Rescore each circumplex item using deviation scoring across variables. In other words, subtract each observation's mean response from each response. This effectively removes the presence of a general factor, which can make certain circumplex fit analyses more powerful.

Usage

```
ipsatize(data, items, na.rm = TRUE, prefix = "", suffix = "_i", append = TRUE)
```

Arguments

data	Required. A data frame or matrix containing at least circumplex scales.
items	Required. A character vector containing the column names, or a numeric vector containing column indexes, of item variables in data to be ipsatized. They must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in data.
na.rm	Optional. A logical that determines whether missing values should be ignored during the calculation of the mean during ipsatization (default = TRUE).
prefix	Optional. A string that will be added to the start of each items name in the output (default = "").
suffix	Optional. A string that will be added to the end of each items name in the output (default = "_i").
append	Optional. A logical that determines whether to append the ipsatized scores to data in the output or just return the ipsatized scores alone (default = TRUE).

Value

A data frame that matches data except that the variables specified in items have been rescored using ipsatization.

See Also

Other tidying functions: norm_standardize(), score(), self_standardize()

Examples

```
data("raw_iipsc")
ipsatize(raw_iipsc, items = 1:32)
ipsatize(raw_iipsc, items = sprintf("IIP%02d", 1:32))
```

---

items *Display the items of a circumplex instrument*

Description

Display the items of a circumplex instrument including the total number of items and each item's number and text. The item ordering/numbering displayed here is the same ordering/numbering assumed by the score() function.

Usage

```
items(x)
```

Arguments

x	Required. An object of the instrument class.

Value

The same input object. Prints text to console.

See Also

Other instrument functions: `anchors()`, `instruments()`, `norms()`, `scales()`

Examples

```
items(csip)
```

---

jz2017 *Raw octant scores on real circumplex scales with covariates*

Description

A large example dataset containing gender, raw mean scores on the Inventory of Interpersonal Problems - Short Circumplex (IIP-SC), and raw sum scores on the Personality Diagnostic Questionnaire - 4th Edition Plus (PDQ-4+).

Usage

```
jz2017
```

Format

A data frame with 1166 observations and 19 variables:

Gender Self-reported Gender

PA Domineering Problems (IIP-SC) 90 degrees

BC Vindictive Problems (IIP-SC) 135 degrees

DE Cold Problems (IIP-SC) 180 degrees

FG Socially Avoidant Problems (IIP-SC) 225 degrees

HI Nonassertive Problems(IIP-SC) 270 degrees

JK Easily Exploited Problems (IIP-SC) 315 degrees

LM Overly Nurturant Problems (IIP-SC) 360 degrees

NO Intrusive Problems (IIP-SC) 45 degrees

PARPD Paranoid PD Symptoms (PDQ-4+)

SCZPD Schizoid PD Symptoms (PDQ-4+)

SZTPD Schizotypal PD Symptoms (PDQ-4+)

ASPD Antisocial PD Symptoms (PDQ-4+)

BORPD Borderline PD Symptoms (PDQ-4+)

HISPD Histrionic PD Symptoms (PDQ-4+)

NARPD Narcissistic PD Symptoms (PDQ-4+)

AVPD Avoidant PD Symptoms (PDQ-4+)

DPNPD Dependent PD Symptoms (PDQ-4+)

OCPD Obsessive-Compulsive PD Symptoms (PDQ-4+)

Source

doi:10.1177/1073191115621795

`norms` *Display the norms for a circumplex instrument*

Description

Display the norms for a circumplex instrument including the total number of normative data sets available and each data set's number, sample size, population, and source reference and hyperlink. If another normative data set exists that is not yet included in the package, please let us know.

Usage

```
norms(x)
```

Arguments

`x`	Required. An object of the instrument class.

Details

The population is a short standardized label chosen by this package so that samples can be compared across instruments; it is deliberately broader than the description the original source gives. Several instruments normed on students at a single named university, in a stated period or region, are all labelled "American college students" here. Consult the reference and hyperlink printed alongside it for the source's own description of the sample before treating a normative sample as representative of a population.

For most samples the label names the group they were drawn from rather than a frame they were drawn to represent – but not for all of them, and which is which is recorded per sample in the `Kind` column and printed as the sample's reference kind:

standardization sample The sample was drawn to represent a defined population, so its mean and standard deviation estimate that population's. Only the IIP-32 and IIP-64 samples are of this kind.

identified published source The sample's octant statistics are printed in an identified source – a study report or an author's norms page – and describe that group of people rather than any wider frame.

no identified source The sample's octant statistics appear in no source that has been identified, whatever is known about the sample itself, and should be treated as unverified.

See `vignette("using-instruments")` for what the shipped reference samples are and how to choose among them.

Value

The same input object. Prints text to console.

See Also

Other instrument functions: `anchors()`, `instruments()`, `items()`, `scales()`

Examples

```
norms(csip)
```

`norm_standardize` *Standardize circumplex scales using normative data*

Description

Take in a data frame containing circumplex scales, angle definitions for each scale, and an instrument whose normative data will be used, and return that same data frame with each specified circumplex scale transformed into standard scores (i.e., z-scores) based on comparison to that instrument's normative sample.

Usage

```
norm_standardize(
  data,
  scales,
  angles = octants(),
  instrument,
  sample = 1,
  prefix = "",
  suffix = "_z",
  append = TRUE,
  quiet = FALSE
)
```

Arguments

`data`	Required. A data frame or matrix containing at least circumplex scales.
`scales`	Required. A character vector containing the column names, or a numeric vector containing the column indexes, for the variables (scale scores) to be standardized. They must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in `data`.
`angles`	Required. A numeric vector containing the angular displacement of each circumplex scale included in `scales` (in degrees). Can use the `octants()`, `poles()`, or `quadrants()` convenience functions. Each angle is matched to the instrument's normative data by angular position, so 0 and 360 degrees are treated as the same angle; an angle with no matching normative row (or with more than one) produces an informative error.

instrument	Required. An instrument object from the package. To see the available circumplex instruments, see instruments().
sample	Required. An integer corresponding to the normative sample to use in standardizing the scale scores (default = 1). See ?norms to see the normative samples available for an instrument. Two conditions are refused with an error rather than used: a sample the instrument does not carry (the error lists the sample numbers it does), and a sample whose mean scores fall outside the instrument's own response range, which cannot be on the same metric as the scores being standardized. norms() lists the alternatives in both cases.
prefix	Optional. A string to include at the beginning of the newly calculated scale variables' names, before the scale name and suffix (default = "").
suffix	Optional. A string to include at the end of the newly calculated scale variables' names, after the scale name and prefix (default = "_z").
append	Optional. A logical that determines whether the calculated standardized scores should be added as columns to data in the output or the standardized scores alone should be output (default = TRUE).
quiet	Optional. A logical that suppresses the message naming the normative sample used (default = FALSE). Set to TRUE in loops, knitted documents, and anywhere else the message is noise; the returned attribute below records the same facts either way.

Details

The sample the scores are compared against is a result-determining choice, not a technicality: different samples of the same instrument can move a respondent's z-scores by more than half a standard deviation. So unless quiet = TRUE, every successful call reports which sample it used, how large that sample was, and how it is described, and every call attaches the same facts to the result (see the Value section below). Use norms() to see the samples an instrument carries before choosing one.

Value

A data frame that contains the norm-standardized versions of scales. It carries a "norm_sample" attribute – a list with elements Instrument, Sample, Size, Population and Kind – recording which normative sample produced the scores and what kind of reference distribution it is (see norms() for the three kinds), so a script that never sees the console can still report what its z-scores are relative to. Retrieve it with attr(x, "norm_sample").

See Also

Other tidying functions: ipsatize(), score(), self_standardize()

Examples

```
data("jz2017")
norm_standardize(jz2017, scales = 2:9, instrument = iipsc, sample = 1)

# The IIP-SC carries more than one normative sample. Omitting `sample` takes
# the first, and the message says which one that was.
```

```
z <- norm_standardize(jz2017, scales = 2:9, instrument = iipsc)
attr(z, "norm_sample")
```

`octants` *Angular displacements for octant circumplex scales*

Description

Return a vector of angular displacements, in degrees, for eight equally spaced circumplex scales corresponding to the circumplex octants. Can be passed to the `angles` parameter of other functions in this package.

Usage

```
octants()
```

Value

A numeric vector with eight elements, each corresponding to the angular displacement (in degrees) of a subscale, in the following order: PA, BC, DE, FG, HI, JK, LM, NO.

Examples

```
octants()
```

`PANO` *Two-letter abbreviations for octant circumplex scales*

Description

Return a vector of abbreviations for octant circumplex scales, from PA to NO.

Usage

```
PANO(case = "upper")
```

Arguments

`case`	An optional string the determines whether the abbreviations should be in uppercase or lowercase. (default = "upper")

Value

A character vector with eight elements, each corresponding to the abbreviation of an octant subscale: PA, BC, DE, FG, HI, JK, LM, NO.

Examples

```
PANO()
PANO(case = "lower")
```

`plot.circumplex_ci_accuracy` *Plot SSM CI accuracy across the amplitude ladder*

Description

Draw the empirical coverage from an `ssm_ci_accuracy()` run against its amplitude-ladder conditions: one panel per SSM parameter (including displacement conditional on guardrail certification), one line per profile row, with 95% Wilson score intervals as error bars, Bradley's (1978) liberal robustness band shaded, and the nominal confidence level as a dashed line. Amplitude rungs whose coverage is structurally zero (a percentile interval of strictly positive amplitude replicates cannot contain a zero truth; see `ssm_ci_accuracy()`) are drawn as open symbols. This is a Cartesian diagnostic plot, not a circumplex figure.

Usage

```
## S3 method for class 'circumplex_ci_accuracy'
plot(x, ...)
```

Arguments

`x`	A `circumplex_ci_accuracy` object from `ssm_ci_accuracy()`.
`...`	Currently ignored.

Value

A **ggplot2** object.

See Also

Other ssm functions: `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `ssm_table()`, `summary.circumplex_ssm_id()`

Other visualization functions: `ssm_plot_circle()`, `ssm_plot_contrast()`, `ssm_plot_curve()`, `ssm_plot_trajectory()`

Examples

```
data("jz2017")
set.seed(12345)
res <- ssm_analyze(
  jz2017[1:200, ],
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  boots = 100
)
set.seed(23456)
acc <- ssm_ci_accuracy(res, reps = 25)
plot(acc)
```

---

plot.circumplex_cpm *Plot a circular process model fit*

Description

Draw the estimated item configuration of a `cpm_fit()` object on the circular canvas from `ggcircumplex()`. Each scale is placed at its *estimated* angle (θ), at a radius given by its communality (ζ^2, the share of its variance explained by the common circumplex factors), so items that the model explains well sit near the outer ring and items it explains poorly sit near the centre. The canvas spokes mark the *theoretical* angles supplied to `cpm_fit()`, so the gap between a point and its spoke shows how far the estimated angle departed from the hypothesised one. Where the confidence intervals are estimable, a wedge spans each item's angle CI (angularly) and communality CI (radially). An interval of zero width has no area, so it is drawn as a line instead: along the radius for a zero-width angle CI, along the arc for a zero-width communality CI, and as a short cap across the interval when both are zero. The cap has the same drawn length at every radius except near the centre, where it spans at most a quarter turn and so is shorter. A scale whose communality CI is zero at both ends has no visible interval at the centre and is drawn as a point, with a warning.

Usage

```
## S3 method for class 'circumplex_cpm'
plot(x, amax = 1, angle_labels = NULL, legend = TRUE, ...)
```

Arguments

`x`	A `circumplex_cpm` object from `cpm_fit()`.
`amax`	A single positive number giving the communality represented by the canvas's outer ring (default = 1, the maximum possible communality).
`angle_labels`	Either `NULL` or a character vector of spoke labels, one per scale in the fitted order. `NULL` (default) labels the spokes with the scale names.
`legend`	A logical: draw a legend keying the colours to the scale names (default = `TRUE`).
`...`	Not used. Supplying an unrecognized argument produces a warning.

Details

The communality axis and its labels are drawn along the midpoint of the widest gap between spokes that holds no estimated angle (ties go to the smallest midpoint; a point on a spoke counts as in both gaps next to it). When every gap holds a point, the axis goes in the widest gap, as `coord_circumplex()` places it by default.

Value

A **ggplot2** object.

See Also

`cpm_fit()`, `ggcircumplex()`

Examples

```
data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
set.seed(12345)
fit <- cpm_fit(jz2017, scales = scales, boots = 100)
plot(fit)
```

`plot.circumplex_structure`

Plot a circumplex-structure configuration

Description

Draw the two-factor loading configuration of a `fit_structure()` object on the circular canvas from `ggcircumplex()`. Each scale is placed at its estimated angle (`atan2` of its two principal-axis loadings) and at a radius given by its communality (the share of its variance on the first two factors), so a clean circumplex shows the scales spread evenly around a ring of similar radius, unequal axes show scales at differing radii (what the Fisher Test measures), and simple structure shows scales bunched near a few angles (what the Gap and interstitiality tests measure). The canvas spokes mark the same estimated angles, labelled by scale.

Usage

```
## S3 method for class 'circumplex_structure'
plot(x, amax = 1, legend = TRUE, ...)
```

Arguments

x	A circumplex_structure object from fit_structure().
amax	A single positive number giving the communality represented by the canvas's outer ring (default = 1). Principal-axis communalities can exceed 1 in a Heywood case; when any scale's communality exceeds amax the ring is expanded to contain it, so no point is ever drawn outside the canvas.
legend	A logical: draw a legend keying the colours to the scale names (default = TRUE).
...	Not used. Supplying an unrecognized argument produces a warning.

Value

A **ggplot2** object.

See Also

fit_structure(), ggcircumplex()

Examples

```
data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
plot(fit_structure(jz2017, scales = scales))
```

---

poles *Angular displacements for pole circumplex scales*

Description

Return a vector of angular displacements, in degrees, for four equally spaced circumplex scales corresponding to the circumplex poles. Can be passed to the angles parameter of other functions in this package.

Usage

```
poles()
```

Value

A numeric vector with four elements, each corresponding to the angular displacement (in degrees) of a subscale, in the following order: PA, DE, HI, LM.

Examples

```
poles()
```

print.circumplex_axes_reliability *Print circumplex axes-reliability results*

Description

Compact display of an `axes_reliability()` object: the per-axis reliability, SEm, and Nunnally-Bernstein comparison, with the correlation-as-covariance standard-error caveat.

Usage

```
## S3 method for class 'circumplex_axes_reliability'
print(x, digits = 3, ...)
```

Arguments

`x`	A `circumplex_axes_reliability` object.
`digits`	The number of decimal places to display (default = 3).
`...`	Not used.

Value

`x`, invisibly.

print.circumplex_cpm *Print a circular process model fit*

Description

Compact display of a `cpm_fit()` object: the estimated angles and communality indices with confidence intervals, a fit summary that wraps after a `;` when it is wider than the console, and any boundary/convergence notes and lower-discrepancy note.

Usage

```
## S3 method for class 'circumplex_cpm'
print(x, digits = 3, ...)
```

Arguments

`x`	A `circumplex_cpm` object.
`digits`	The number of decimal places to display (default = 3).
`...`	Not used.

Value

`x`, invisibly.

print.circumplex_structure *Print circumplex-structure test results*

Description

Compact display of a `fit_structure()` object: the four factor-analytic criteria with their statistics and interpretive classifications, and the RANDALL order test with its randomization p-value.

Usage

```
## S3 method for class 'circumplex_structure'
print(x, digits = 3, ...)
```

Arguments

`x`	A `circumplex_structure` object.
`digits`	The number of decimal places to display (default = 3).
`...`	Not used.

Value

`x`, invisibly.

quadrants *Angular displacements for quadrant circumplex scales*

Description

Return a vector of angular displacements, in degrees, for four equally spaced circumplex scales corresponding to the circumplex quadrants. Can be passed to the `angles` parameter of other functions in this package.

Usage

```
quadrants()
```

Value

A numeric vector with eight elements, each corresponding to the angular displacement (in degrees) of a subscale, in the following order: BC, FG, JK, NO.

Examples

```
quadrants()
```

raw_iipsc *Raw item responses on real circumplex scales*

Description

A small example dataset containing raw item responses on the Inventory of Interpersonal Problems, Short Circumplex (IIP-SC). This data set is useful for testing functions that operate on item-level data.

Usage

```
raw_iipsc
```

Format

A data frame with 10 observations and 32 variables.

scales *Display the scales of a circumplex instrument*

Description

Display the scales of a circumplex instrument including the total number of scales and each scale's abbreviation, hypothetical angle, and text label.

Usage

```
scales(x, items = FALSE)
```

Arguments

x	Required. An object of the instrument class.
items	Optional. A logical determining whether the items for each scale should be displayed below its other information (default = FALSE).

Value

The same input object. Prints text to console.

See Also

Other instrument functions: anchors(), instruments(), items(), norms()

Examples

```
scales(csip)
scales(csip, items = TRUE)
```

scale_x_circumplex *Angle-labeled x-axis scale for circumplex plots*

Description

A **ggplot2** continuous position scale for the angle axis of a linear circumplex plot, such as the score-by-angle curve drawn by `ssm_plot_curve()`. It places axis breaks at the circumplex scale angles and labels them, by default, with their angular position in degrees. Custom text labels or a `circumplex_instrument` can be supplied instead, using the same conventions as `ggcircumplex()`, so the linear axis and the circular canvas label their scales consistently.

Usage

```
scale_x_circumplex(angles = octants(), labels = NULL, instrument = NULL, ...)
```

Arguments

`angles`	Optional. A numeric vector of the angular position (in degrees) of each circumplex scale (default = `octants()`). Ignored if `instrument` is supplied.
`labels`	Optional. Either `NULL` or a character vector of axis labels, one per angle and in the same order (default = `NULL`, which labels each break with its angle in degrees, or with the instrument's scale abbreviations when `instrument` is supplied).
`instrument`	Optional. Either `NULL` or a `circumplex_instrument` object (see `instrument()`) from which to take the scale angles and (unless `labels` is given) abbreviations (default = `NULL`).
`...`	Additional arguments passed to `ggplot2::scale_x_continuous()`, such as `name` or `limits`.

Value

A **ggplot2** scale object that can be added to a plot with `+`.

See Also

Other circumplex layers: `coord_circumplex()`, `geom_ssm_arc()`, `geom_ssm_ellipse()`, `geom_ssm_path()`, `geom_ssm_point()`, `ggcircumplex()`, `theme_circumplex()`

Examples

```
# Degree-labeled angle axis
scale_x_circumplex(octants())

# Label the axis with an instrument's scale abbreviations
scale_x_circumplex(instrument = csip)
```

score *Score circumplex scales from item responses*

Description

Calculate mean scores on circumplex scales from item responses by using the scoring instructions stored in an instrument object from the package.

Usage

```
score(
  data,
  items,
  instrument,
  na.rm = TRUE,
  prefix = "",
  suffix = "",
  append = TRUE
)
```

Arguments

data	Required. A data frame or matrix containing at least circumplex scales.
items	Required. The variable names or column numbers for the variables in .data that contain all the circumplex items from a single circumplex measure, in ascending order from item 1 to item N. They must not select one column twice (the scoring key reads the items by position, so a repeat would put a column in another item's place), hold a number below 1, or name a column whose name occurs more than once in data.
instrument	Required. An instrument object from the package. To see the available circumplex instruments, use instruments().
na.rm	Optional. A logical that determines if missing values should be omitted from the calculation of scores (default = TRUE). When set to TRUE, scales with missing data are essentially calculated with mean imputation.
prefix	Optional. A string to include at the beginning of the newly calculated scale variables' names, before Abbrev from key and suffix (default = "").
suffix	Optional. A string to include at the end of the newly calculated scale variables' names, after Abbrev from key and prefix (default = "").
append	Optional. A logical that determines whether the calculated score variables will be appended to data or returned on their own (default = TRUE).

Value

A data frame that matches .data except that new variables are appended that contain mean scores on each variable included in key.

See Also

Other tidying functions: `ipsatize()`, `norm_standardize()`, `self_standardize()`

Examples

```
data("raw_iipsc")
score(raw_iipsc, items = 1:32, instrument = iipsc, prefix = "IIPSC_")
```

`self_standardize` *Standardize circumplex scales using sample data*

Description

Take in a data frame containing circumplex scales (or items) and return that same data frame with each specified variable transformed into standard scores (i.e., z-scores) based on observed means and SDs.

Usage

```
self_standardize(
  data,
  scales,
  na.rm = TRUE,
  prefix = "",
  suffix = "_z",
  append = TRUE
)
```

Arguments

`data`	Required. A data frame or matrix containing at least circumplex scales.
`scales`	Required. A character vector containing the column names, or a numeric vector containing the column indexes, for the variables (scale scores) to be standardized. They must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in `data`.
`na.rm`	Optional. A logical that determines whether to remove missing values from scales when calculating the means and SDs used for standardization (default = TRUE).
`prefix`	Optional. A string to include at the beginning of the newly calculated scale variables' names, before the scale name and `suffix` (default = "").
`suffix`	Optional. A string to include at the end of the newly calculated scale variables' names, after the scale name and `prefix` (default = "_z").
`append`	Optional. A logical that determines whether the calculated standardized scores should be added as columns to `data` in the output or the standardized scores alone should be output (default = TRUE).

Value

A data frame that contains the self-standardized versions of `scales`.

See Also

Other tidying functions: `ipsatize()`, `norm_standardize()`, `score()`

Examples

```
self_standardize(aw2009, scales = 1:8)
```

---

simulated_growth *Simulated octant scores for growth models*

Description

Two simulated datasets for demonstrating growth models on SSM parameters with `ssm_growth_data()`, `ssm_growth_formula()` and `ssm_trajectory()`. The data are simulated, not collected from people. In both datasets, 150 persons have scores on the eight octant scales at waves 0 to 4.

Usage

```
simulated_growth

simulated_growth_origin
```

Format

Each is a data frame with 750 rows (one per person per wave) and 10 columns:

person Person identifier, 1 to 150.

wave Wave number, an integer from 0 to 4.

PA, BC, DE, FG, HI, JK, LM, NO Scores on the eight octant scales, at the angles given by `octants()`.

An object of class `data.frame` with 750 rows and 10 columns.

Details

A person's score on a scale at a wave is the elevation 0.5, plus a person elevation effect, plus $(x+v_x)\cos\theta+(y+v_y)\sin\theta$, plus noise. Here θ is the scale angle and (x, y) is the group coordinate at that wave. The person elevation effect has standard deviation 0.30, the person effects v_x and v_y have standard deviation 0.15, and the noise has standard deviation 0.40. All of them are drawn from normal distributions. The person effects stay the same at every wave, and the two datasets share them, so each person has the same effects in both.

The two datasets differ in the group coordinates:

- In `simulated_growth`, (x, y) moves in a straight line, in equal steps, from the point with amplitude 0.6 at 350 degrees to the point with amplitude 0.6 at 10 degrees. The displacement crosses the 0/360 degree boundary.
- In `simulated_growth_origin`, x moves in equal steps from 0.5 to -0.5, and y stays at 0.02. The group passes near the origin at wave 2.

Both datasets are written by one seeded script, `data-raw/simulated_growth.R`, which is not in the installed package. Read it on GitHub: https://github.com/jmgirard/circumplex/blob/master/data-raw/simulated_growth.R.

See Also

`vignette("growth-ssm-analysis")`, which fits a growth model to both datasets.

`simulated_items` *Simulated item responses on octant circumplex scales*

Description

A simulated item-level dataset for demonstrating `axes_reliability()`. It contains 1-7 Likert responses from 500 respondents on 32 items – four items on each of the eight octant circumplex scales (`PA` at 90 degrees through `NO` at 45 degrees, following `octants()`). The items were drawn from the five-component population of Strack, Jacobs, and Grosse Holtforth (2013): a general factor, two equal circumplex axes (axes variance .18), one shared scale-specificity component (.10), no block specificity (the instrument is not blockwise, so that fifth component is zero here), and free item error, giving an axes reliability of about .78. The data are synthetic – there is no public raw-data source for the method – and are generated by a seeded script (`data-raw/simulated_items.R`).

Usage

```
simulated_items
```

Format

A data frame with 500 observations and 32 variables named `PA_1`–`PA_4`, `BC_1`–`BC_4`, ..., `NO_1`–`NO_4` (four items per octant scale).

References

Strack, S., Jacobs, K. A., & Grosse Holtforth, M. (2013). The reliability of circumplex axes. *SAGE Open*, 3(2). doi:10.1177/2158244013486115

simulated_occasions *Simulated octant scores at three occasions*

Description

A simulated dataset for demonstrating `ssm_analyze_long()` and `ssm_plot_trajectory()`. The data are simulated, not collected from people. 200 persons have scores on the eight octant scales at three waves. At each wave the group profile is a cosine curve with elevation 0 and amplitude 0.6. Its displacement is 330 degrees at `T1`, 355 degrees at `T2` and 20 degrees at `T3`, so the profile crosses the 0/360 degree boundary between `T2` and `T3`. Each person has one offset, added to all eight scales at every wave, drawn from a normal distribution with standard deviation 0.5. Each score also has its own noise, drawn from a normal distribution with standard deviation 0.5.

Usage

```
simulated_occasions
```

Format

A data frame with 600 rows (one per person per wave) and 10 columns:

id Person identifier, 1 to 200.

wave Occasion, a factor with levels `T1`, `T2` and `T3` in that order.

PA, BC, DE, FG, HI, JK, LM, NO Scores on the eight octant scales, at the angles given by `octants()`.

Details

The data are written by a seeded script, `data-raw/simulated_occasions.R`, which is not in the installed package. Read it on GitHub: https://github.com/jmgirard/circumplex/blob/master/data-raw/simulated_occasions.R.

ssm_analyze *Perform analyses using the Structural Summary Method*

Description

Calculate SSM parameters with confidence intervals (bootstrapped by default, or Monte Carlo via `method`) for a variety of different analysis types. Depending on what arguments are supplied, either mean-based or correlation-based analyses will be performed, one or more groups will be used to stratify the data, and contrasts between groups or measures will be calculated.

Usage

```
ssm_analyze(
  data,
  scales = NULL,
  angles = octants(),
  measures = NULL,
  grouping = NULL,
  contrast = FALSE,
  boots = 2000,
  interval = 0.95,
  listwise = TRUE,
  measures_labels = NULL,
  parallel = "no",
  ncpus = 1,
  method = "bootstrap",
  occasions = NULL
)
```

Arguments

data	Required. A data frame or matrix containing at least circumplex scales.
scales	Required unless occasions is supplied (the two are mutually exclusive). A character vector of column names, or a numeric vector of column indexes, from data that contains the circumplex scale scores to be analyzed. It must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in data.
angles	Optional. A numeric vector containing the angular displacement of each circumplex scale included in scales (in degrees). (default = octants()). The closed-form SSM estimator used here equals the ordinary-least-squares cosine fit for equally spaced angles (e.g., octants at 45-degree intervals) – more generally, for any angle set satisfying first- and second-harmonic balance. For angle sets violating that balance (generic unequally spaced sets), it is the conventional Gurtman estimator, not a least-squares fit, and the reported model fit is then no longer a bounded R-squared in [0, 1] (it can fall below 0).
measures	Optional. Either NULL or a character vector of column names from data that contains one or more variables to be correlated with the circumplex scales and analyzed using correlation-based SSM analyses.
grouping	Optional. Either NULL or a string that contains the column name from data of the variable that indicates the group membership of each observation.
contrast	Optional. A logical indicating whether to output the difference between two measures', two groups', or two occasions' SSM parameters. Can only be set to TRUE when exactly one of these holds: two measures and one group; one measure and two groups; no measures and two groups; or two occasions and one group (default = FALSE). The contrast is always the second level minus the first. For two groups, this is the second level of grouping alphabetically, unless grouping is already a factor with an explicit level order, in which case that order is used. For two measures, this is simply the second entry of measures as given

(no reordering). For two occasions, it is the second listed element of `occasions` minus the first (list order as supplied – temporal order – never alphabetical). The direction is shown in the result's Label (e.g., "Male - Female").

`boots` Optional. A single positive whole number indicating how many bootstrap resamples (or, when `method = "montecarlo"`, Monte Carlo draws) to use when estimating the confidence intervals (default = 2000).

`interval` Optional. A single positive number between 0 and 1 (exclusive) that indicates what confidence level to use when estimating the confidence intervals (default = 0.95).

`listwise` Optional. A logical indicating whether missing values should be handled by listwise deletion (TRUE) or pairwise deletion (FALSE). Note that pairwise deletion may result in different missing data patterns in each bootstrap resample and is slower to compute (default = TRUE). Occasions analyses require `listwise = TRUE`: a person missing any occasion is dropped from all occasions (complete cases across waves), so the paired contrast stays a within-person comparison. Note the selection caution: complete-cases-across-waves estimates completers' change, which can differ from population change when dropout relates to the outcome.

`measures_labels` Optional. Either `NULL` or a character vector providing a label for each measure provided in `measures` (in the same order) to appear in the results as well as tables and plots derived from the results.

`parallel` Optional. A string indicating whether to distribute the bootstrap computation across multiple CPU cores: "no" (default), "multicore" (process forking; available on macOS and Linux, ignored on Windows), or "snow" (a local PSOCK cluster; available on all platforms). Passed to `boot`. Because the bootstrap resample indices are drawn in the main R process before any work is distributed, results for a given `set.seed()` are identical regardless of the `parallel` and `ncpus` settings.

`ncpus` Optional. A single positive whole number indicating how many CPU cores to use when `parallel` is not "no" (default = 1).

`method` Optional. A string indicating how to estimate the confidence intervals: "bootstrap" (default) resamples the data, whereas "montecarlo" draws parameter replicates from the asymptotic sampling distribution of the group mean vector (mean-based analyses) or the measure-scale correlation vector (correlation-based analyses) – a multivariate normal with empirically estimated covariance – and propagates them through the SSM parameter transformation. The Monte Carlo method is much faster for large samples but relies on the asymptotic normality of the means or correlations, so prefer the bootstrap for small samples; it also requires listwise-complete data. Correlations are drawn jointly across measures within each group on the Fisher z scale and back-transformed. The `parallel` and `ncpus` arguments apply only to the bootstrap.

`occasions` Optional. Either `NULL` or a named list of character or numeric vectors, each selecting the same circumplex scales measured at one occasion, in the same scale order, all of length `length(angles)` (e.g., `occasions = list(T1 = c("PA_1", ..., "NO_1"), T2 = c("PA_2", ..., "NO_2"))`). Mutually exclusive with `scales`

(and not combinable with measures). A character block must not name a column whose name occurs more than once in data. Data must be wide – one row per person – so persons remain the resampling unit and within-person dependence across occasions is preserved in both engines. Results gain an Occasion column (labels are names(occasions), defaulting to T1..Tk); this column is present only for occasions analyses. Grouping is time-invariant by construction (one group per person-row). Cross-occasion column alignment is validated by stem matching; when the columns have no common stem structure, positional alignment is assumed and messaged.

Value

A list containing the results and description of the analysis.

results	A data frame with the SSM parameter estimates
details	A list with the number of bootstrap resamples or Monte Carlo draws (boots), the confidence interval percentage level (interval), the angular displacement of scales (angles), and the interval estimation method (method)
call	A language object containing the function call that created this object
scores	A data frame containing the mean scale scores
type	A string indicating what type of SSM analysis was done

The profile displacement parameter is reported in the half-open interval [0, 360) degrees. A profile that peaks exactly at the 0/360 degree boundary is reported as approximately 360 (equivalently 0, the same direction); which of the two appears is a floating-point detail and both denote the same pole. A displacement *confidence-interval endpoint* that lands exactly on that pole is always reported as 360 (never 0), matching the package's LM = 360 labeling. Contrast displacements are instead reported as a signed difference in (-180, 180] degrees (see the "Contrast" block in the printed output).

Degenerate profiles (flat or zero-amplitude) have undefined displacement (and fit, if flat), which is reported as NA with a warning. Bootstrap resamples that produce degenerate profiles (e.g., a resampled measure with zero variance) are excluded from the confidence intervals with a warning reporting how many were dropped; the intervals are then conditional on estimability.

[0, 360)degrees. A profile that peaks exactly at the 0/360 degree boundary is reported as approximately 36
180, 180]: R:0,%20360)%60%20degrees.%20A%20profile%20that%20peaks%20exactly%20at%20the%200/360%20degre
point%20detail%20and%20both%0A%20%20denote%20the%20same%20pole.%20A%20displacement%20*confidence-
interval%20endpoint*%20that%0A%20%20lands%20exactly%20on%20that%20pole%20is%20always%20reported%20as%
180,%20180

Reproducibility

This function consumes R's random number stream (so do cpm_fit(ci_method = "bootstrap"), cpm_simulate(), and ssm_ci_accuracy(); ssm_score()/ssm_parameters() and the tidying functions are deterministic). Call set.seed() immediately before ssm_analyze() for reproducible confidence intervals:

- **Bootstrap** (method = "bootstrap", the default): the same seed gives byte-identical results, *regardless of* the parallel/ncpus settings (see their descriptions below), because boot::boot() draws all resample indices from the seed before any work is parallelized.

- **Monte Carlo** (`method = "montecarlo"`): the same seed gives byte-identical `results`. Adding a group or measure, or reordering `scales`/`measures`, changes the random draw sequence, so results are reproducible for a fixed call but will not match after such structural edits even with the same seed.
- The two methods are **not** expected to agree numerically for the same seed – they consume the random stream in unrelated ways. Their statistical agreement (validated on real data; see `vignette("introduction-to-ssm-analysis")`) is a separate property from RNG reproducibility.
- Increasing `boots` changes the CI by design (more resamples/draws should tighten Monte Carlo error), so results are not expected to be stable across different `boots` values, only within a fixed call.

Occasions (repeated measures)

Supplying `occasions` analyzes the same circumplex scales measured at k >= 2 occasions on the same persons (wide data, one row per person). Each occasion yields its own profile row; with `contrast = TRUE` (exactly 2 occasions, single group) the paired within-person contrast is estimated with both engines preserving the within-person dependence (the bootstrap resamples persons; the Monte Carlo engine draws the stacked occasion mean vectors jointly).

Interpretation notes. A paired displacement-contrast CI is interpretable only when *both* occasions' amplitudes are reliably nonzero (both profiles print without the amplitude note); if only one occasion's profile is interpretable, do not read the contrast as directional change. Paired designs are not unconditionally more efficient than independent groups: the paired elevation contrast has a narrower CI exactly when the within-person elevation correlation is positive, while for the amplitude and displacement contrasts the paired CI is narrower only when the gradient-projected cross-occasion covariance is positive – under isotropic dependence this is proportional to cos(displacement change), so paired CIs are narrower for displacement changes under 90 degrees and can be *wider* than independent-groups CIs for changes beyond 90 degrees, even with strongly positive within-person correlation.

With `method = "montecarlo"` the per-group draw has dimension k x p (occasions times scales); group sizes should comfortably exceed k x p for the asymptotic covariance to be well estimated (the percentile bootstrap is the safer small-sample choice). Grouping is time-invariant by construction (one group per person-row).

See Also

Other ssm functions: `plot.circumplex_ci_accuracy()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `ssm_table()`, `summary.circumplex_ssm_id()`

Other analysis functions: `cpm_fit()`, `cpm_simulate()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Examples

```
# `boots` is lowered from its default of 2000 throughout these examples so
# they run quickly; a reported analysis should use the default.
```

```
# Load example data
data("jz2017")

# Single-group mean-based SSM
ssm_analyze(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  boots = 200
)

# Single-group correlation-based SSM
ssm_analyze(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  measures = c("NARPD", "ASPD"),
  boots = 200
)

# Monte Carlo confidence intervals (faster for large samples)
ssm_analyze(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  method = "montecarlo",
  boots = 200
)

# Multiple-group mean-based SSM
ssm_analyze(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  grouping = "Gender",
  boots = 200
)

# Multiple-group mean-based SSM with contrast
ssm_analyze(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  grouping = "Gender",
  contrast = TRUE,
  boots = 200
)

# Single-group correlation-based SSM with contrast
ssm_analyze(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  measures = c("NARPD", "ASPD"),
  contrast = TRUE,
  boots = 200
)

# Multiple-group correlation-based SSM
```

```
ssm_analyze(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  measures = "NARPD",
  grouping = "Gender",
  boots = 200
)

# Multiple-group correlation-based SSM with contrast
ssm_analyze(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  measures = "NARPD",
  grouping = "Gender",
  contrast = TRUE,
  boots = 200
)
```

`ssm_analyze_long` *Perform SSM analyses on long-format repeated-measures data*

Description

A convenience wrapper around the `occasions` interface of `ssm_analyze()` for data stored in long format (one row per person per occasion). It reshapes the data into the wide, one-row-per-person layout that `ssm_analyze()` consumes and then delegates to it; all estimation is performed by `ssm_analyze()` unchanged. See the `occasions` argument of `ssm_analyze()` for the analysis semantics – per-occasion profiles, paired within-person contrasts, and the listwise-only handling of missing waves (a person missing any occasion is dropped from all occasions).

Usage

```
ssm_analyze_long(
  data,
  scales,
  angles = octants(),
  id,
  occasion,
  grouping = NULL,
  contrast = FALSE,
  boots = 2000,
  interval = 0.95,
  parallel = "no",
  ncpus = 1,
  method = "bootstrap"
)
```

Arguments

`data`	Required. A data frame (or matrix) in long format containing an identifier column, an occasion column, and the circumplex scale scores (one set of score columns, repeated across occasions in different rows).
`scales`	Required. A character vector of column names, or a numeric vector of column indexes, giving the circumplex scale scores in `data` (the same scales measured at every occasion). It must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in `data`. The scores are reshaped by scale name, so numbers must not reach two columns that share a name either.
`angles`	Optional. A numeric vector containing the angular displacement of each circumplex scale included in `scales` (in degrees) (default = `octants()`).
`id`	Required. A single column name or index identifying the person that each row belongs to.
`occasion`	Required. A single column name or index identifying the occasion (wave) that each row belongs to. Occasion order – which governs the second-minus-first direction of a paired contrast – is taken from the factor levels of this column when it is a factor, and otherwise from the order in which the occasions first appear in `data`. It is never sorted alphabetically, so a `T10`/`T2` pair keeps its supplied order.
`grouping`	Optional. A single column name or index giving a time-invariant grouping variable (one value per person; an error is raised if a person's grouping value varies across occasions).
`contrast`	Optional. A logical value; if `TRUE` (and the data contain exactly two occasions in a single group), the paired within-person contrast (second occasion minus first) is calculated (default = FALSE).
`boots`, `interval`, `parallel`, `ncpus`, `method`	Optional. Passed through to `ssm_analyze()`; see its documentation.

Value

A list containing the results and description of the analysis, as returned by `ssm_analyze()` (with an `Occasion` column). See `ssm_analyze()`.

See Also

`ssm_analyze()` for the wide-format interface and the analysis semantics this wrapper delegates to.

Other ssm functions: `plot.circumplex_ci_accuracy()`, `ssm_analyze()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `ssm_table()`, `summary.circumplex_ssm_id()`

Other analysis functions: `cpm_fit()`, `cpm_simulate()`, `ssm_analyze()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Examples

```
# `boots` is lowered from its default of 2000 throughout these examples so
# they run quickly; a reported analysis should use the default.

# Build a small two-occasion dataset in long format (one row per person per
# occasion). In practice `data` already stores the repeated occasions this
# way; here we stack two copies of jz2017 as an illustration.
data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
t1 <- jz2017[, scales]
t1$id <- seq_len(nrow(t1))
t1$occasion <- "T1"
t2 <- t1
t2$occasion <- "T2"
long <- rbind(t1, t2)


# Per-occasion SSM profiles from long-format data
ssm_analyze_long(
  long, scales = scales, id = "id", occasion = "occasion",
  boots = 200
)
```

`ssm_ci_accuracy` *Assess the accuracy of SSM confidence intervals by simulation*

Description

Estimate, by simulation, whether the confidence intervals of a fitted `ssm_analyze()` object would cover the true SSM parameters at their nominal rate if the population looked like the fitted estimates, at the observed sample size(s). Following Zimmermann & Wright (2017), the population's scale intercorrelation structure is characterized by fitting Browne's (1992) circular process model (`cpm_fit()`) to the pooled within-group scale correlations; each of `reps` datasets is then simulated from that plug-in population at the object's exact group sizes and the object's own interval procedure (same engine, same `boots`, same `interval`) is rerun on it. Coverage is reported per profile row, parameter, and amplitude condition, with 95% Wilson score intervals classified against Bradley's (1978) liberal robustness band at the as-estimated condition.

Usage

```
ssm_ci_accuracy(
  ssm_object,
  reps = 1000,
  amplitude_factors = c(1, 0.5, 0.25, 0),
  structure = c("cpm", "observed"),
  m = NULL,
  cpm = NULL,
```

```
  data = NULL,
  parallel = "no",
  ncpus = 1
)
```

Arguments

ssm_object	Required. A circumplex_ssm object from ssm_analyze().
reps	Optional. The number of simulated datasets per amplitude condition (default = 1000, binomial SE about 0.7 percentage points; 500 is a reasonable quick-look floor).
amplitude_factors	Optional. Numeric vector of amplitude scaling factors in [0, 1] defining the ladder conditions (default = c(1, 0.5, 0.25, 0)). Must include 1, the as-estimated condition that the verdict is keyed to. Applied to every profile row jointly.
structure	Optional. "cpm" (default) simulates from the Browne model's model-implied scale correlations; "observed" bypasses the model and uses the pooled within-group correlation matrix directly (a sensitivity switch: if the two verdicts differ, structure uncertainty is itself material). Not accepted for occasions analyses, whose population is always the observed stacked cross-occasion covariance (no model is fit).
m	Optional. The number of harmonics passed to cpm_fit() (default NULL uses min(3, floor((p - 1) / 2))).
cpm	Optional. A pre-fitted circumplex_cpm object to reuse for the population structure instead of refitting (its scales must match the ssm object's). Must be a unit-scaling fit (the default cpm_fit() scaling); a free-scaling fit models a covariance structure and is not a valid population correlation structure for this diagnostic. Ignored when structure = "observed"; not accepted for occasions analyses.
data	Optional. The original data set, required only for ssm objects created before sufficient statistics were stored at analysis time; the statistics are then recomputed and checked against the stored profile vectors.
parallel	Optional. "no" (default), "multicore", or "snow"; distributes the simulation replicates across ncpus cores. Results are identical for a given seed regardless of these settings (see Reproducibility).
ncpus	Optional. Number of cores when parallel is not "no" (default = 1).

Details

Displacement coverage is angular: the truth is inside the reported interval as an arc (membership modulo 360 degrees), so populations peaking at the 0/360 boundary and contrast intervals reported beyond +/-180 degrees are handled without special-casing. Because displacement is only interpreted when the printed amplitude guardrail certifies it, displacement coverage is also reported conditional on certification under the shipped decision rule a_lci / (a_uci - a_lci) >= 0.35 (the rule the printed ssm_analyze() output applies): the amplitude CI's lower bound must sit at least 0.35 CI widths above zero. The rule is scale-free (invariant to the score metric) and print-independent, so no scale-dependent threshold is reported; the 0.35 constant is calibrated for

the default 95% confidence interval. A contrast row is a signed difference, not a prototypicality measure, so `print.circumplex_ssm()` never certification-gates it; its displacement verdict and printed coverage are therefore reported unconditionally (matching that profiles-only stance). Its certification-conditional coverage – where "certified" means both profile rows were certified – is still computed and retained in the returned object as a descriptive that no display consumes.

The `amplitude_factors` ladder manufactures populations whose closed-form amplitude is scaled toward zero (the regime where percentile amplitude intervals are theoretically weakest) while keeping the residual profile content fixed. The ladder is defined through the estimator functional (a 3x3 solve on the images of 1, cos, and sin), so the condition-`c` truths are exact for any angle spacing: elevation is unchanged and the amplitude is exactly `c` times the estimate. Truths are nevertheless recomputed from each condition's population profile. At `c = 0` amplitude coverage is structurally zero (a percentile interval of strictly positive amplitude replicates cannot contain 0; such rows are flagged in the `Structural` column) and displacement truth is undefined (reported `NA`); the guardrail certification rate carries the inferential weight there, and the informative rungs for amplitude coverage are the small `c > 0` ones.

When a profile row's amplitude estimate is itself below half its observed CI width, the relative ladder degenerates: the analysis already sits in the near-zero regime. One absolute rung is then added at the certification margin (`c` chosen so `c` times the amplitude estimate equals the observed amplitude-CI half-width, the largest such `c` across affected rows) and `summary()` notes the regime. On the correlation path this rung is dropped with a warning if it would push a population correlation to +/-1.

Value

A `circumplex_ci_accuracy` object: a list with `coverage` (per Profile x Parameter x Condition: coverage, its Monte Carlo SE, the one-sided miss rates, median CI width, and for displacement the certification-conditional coverage with the number of certified replicates behind it – for a contrast row this conditional column is retained as a joint-certification descriptive that no display consumes; `Structural` flags the amplitude rows whose zero coverage is a theorem rather than a measurement), `guardrail` (per Profile x Condition: certification rate with its 95% Wilson score interval, the user-expectation benchmark `(1 - interval) / 2`, the stored false-certification caution decision at the `c = 0` rung (`Caution`, true when the Wilson lower bound exceeds the benchmark; `NA` off that rung, and `NA` for a contrast row, which `print.circumplex_ssm()` never gates), fit-pass rate, and the branch-pathology rate – the rate at which a displacement point estimate falls geometrically outside its own interval), `verdict` (Wilson-vs-Bradley classification of elevation, amplitude, and displacement coverage at the as-estimated condition – a profile's displacement is classified certification-conditionally (`Parameter "d_conditional"`), a contrast's unconditionally (`Parameter "d"`) – plus an overall worst-of row per profile; note the printed verdict headline additionally elevates to CAUTION whenever the guardrail `Caution` fired, so it can read worse than the overall coverage class), `cpm` (the embedded `cpm_fit()` object, or `NULL` when `structure = "observed"`; when this function fits it, it skips the lower-discrepancy check of `cpm_fit(check_lower = )`), `population` (per profile row: the population profile vectors, truth parameters, and any positive-semidefiniteness repair magnitude, by condition), and `details`. The `plot()` method draws coverage against the amplitude ladder with the Bradley band shaded; `summary()` adds a plain-language verdict (see `summary.circumplex_ci_accuracy()`).

Reproducibility

This function is stochastic: call `set.seed()` immediately before it. It draws one `sample.int()` value from the caller's random number stream to seed an internal L'Ecuyer-CMRG stream, gives every simulated dataset its own deterministic substream, and then restores the caller's `.Random.seed` and generator kind on exit, so results for a given seed are identical regardless of `parallel`/`ncpus` and the caller's stream is advanced by exactly that one draw.

Limitations

Coverage is evaluated at the fitted structure, not the unknown truth ("would the procedure work in a population like your estimates", not "did your interval cover"). Simulated populations are multivariate normal with the fitted correlation structure; heavy tails or skew in the real data can degrade coverage further than reported. The diagnostic assesses the complete-data procedure (missing data are not simulated), and groups are assumed to share one circumplex structure. When the Browne model fits poorly, the simulated population may misrepresent the data; the embedded fit and its diagnostics are returned for inspection.

For an occasions (repeated-measures) analysis the population is instead a multivariate normal with the *observed* stacked cross-occasion covariance (no circular-model idealization): the within-person dependence across occasions is carried directly, and no `structure`/`cpm` alternative is offered. When the stacked covariance is rank-deficient (per-group sample size at or below the number of occasions times scales) the simulated population is a proper degenerate normal, so the reported coverage and width remain valid but the fit-statistic pass rate is descriptive only. For a paired contrast, the joint-certification rate (both occasions certified) is reported as a descriptive caveat on the contrast's interpretability. Assessing the occasions object per occasion via `scales =` one occasion's columns instead uses the default `"cpm"` structure and can give slightly different per-occasion verdicts – a structure-sensitivity fact, not a bug.

References

Zimmermann, J., & Wright, A. G. C. (2017). Beyond description in interpersonal construct validation: Methodological advances in the circumplex Structural Summary Approach. *Assessment, 24*(1), 3-23.

Browne, M. W. (1992). Circumplex models for correlation matrices. *Psychometrika, 57*(4), 469-497.

Bradley, J. V. (1978). Robustness? *British Journal of Mathematical and Statistical Psychology, 31*(2), 144-152.

See Also

Other ssm functions: `plot.circumplex_ci_accuracy()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `ssm_table()`, `summary.circumplex_ssm_id()`

Other analysis functions: `cpm_fit()`, `cpm_simulate()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Examples

```
data("jz2017")
set.seed(12345)
res <- ssm_analyze(
  jz2017[1:200, ],
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  boots = 100
)
# Small reps/boots keep the example fast; use the defaults in practice
set.seed(23456)
acc <- ssm_ci_accuracy(res, reps = 25, amplitude_factors = c(1, 0.25))
acc
summary(acc)
```

`ssm_draws` *Summarize posterior draws as SSM parameters*

Description

Transform posterior draws from a user-fitted Bayesian model (e.g., a brms cosine regression) into Structural Summary Method parameter draws and summarize them with the package's circular-statistics machinery. Two draw shapes are accepted, distinguished explicitly (never guessed):

Usage

```
ssm_draws(draws, angles = NULL, interval = 0.95, type = NULL)
```

Arguments

`draws`	Required. A numeric matrix or data frame of posterior draws: one row per draw, columns per the shape rules above.
`angles`	Optional. A numeric vector of angular displacements (in degrees) for profile draws, one per column of `draws`; `NULL` (default) for parameter draws.
`interval`	Optional. A single number between 0 and 1 giving the credible level for the equal-tailed intervals (default = 0.95).
`type`	Optional. `"parameters"` or `"profiles"`, required only where the shape is ambiguous (`angles = NULL` with exactly 3 columns); when given elsewhere it must not contradict `angles`.

Details

- **Parameter draws** (`angles = NULL`, `type = "parameters"`): a numeric matrix or data frame with exactly three columns interpreted **in column order** as (e, x, y) – elevation (intercept), the cosine coefficient (x), and the sine coefficient (y). Each row is mapped to amplitude `a = sqrt(x^2 + y^2)` and displacement `d = atan2(y, x)` wrapped to [0, 360] (the 0/360 pole is reported as 360, the package's LM = 360 convention). Column names are not used to reorder:

when names are present but do not look (intercept, cos, sin)-like, a message states the assumed mapping. A row with exactly zero amplitude has undefined displacement (NA); model fit is undefined for parameter draws (fit = NA) because no profile is available to measure it against.

- **Profile draws** (angles supplied): a numeric matrix with one column per circumplex scale (ncol(draws) == length(angles)); each row goes through the closed-form SSM transform exactly as a bootstrap replicate would, inheriting the standing degenerate-profile NA semantics.

With angles = NULL and a column count other than 3 the input matches neither shape and an error explains both. With angles = NULL and exactly 3 columns the shape is ambiguous (a p = 3 instrument's profile draws look like parameter draws), so type = "parameters" is required.

Point estimates are posterior medians for e, x, y, a, and fit (amplitude is right-skewed, so a mean would be biased upward), and the circular mean for displacement. Marginal summaries are not jointly coherent: the reported a is not sqrt(x^2 + y^2) of the reported (x, y), and the reported d is not their direction – each is the honest marginal summary of its own posterior. Intervals are equal-tailed credible intervals (percentile quantiles of the draws), with displacement handled by the package's circular quantile machinery (centered on the circular mean, so intervals straddling 0/360 wrap correctly). Draws with undefined displacement are excluded from the displacement summaries only, which are therefore conditional on estimability (measure-zero for continuous parameter-draw posteriors; can bind for profile draws). A diffuse posterior with zero circular resultant has an undefined circular mean, reported as NA rather than invented.

Note that independent priors on (x, y) induce a non-uniform prior on (a, d) – roughly Rayleigh-shaped on amplitude, with mass pushed away from a = 0 – so the prior on the SSM scale should be inspected (e.g., by prior-predictive simulation) rather than assumed flat; see the package's Bayesian SSM vignette.

Value

An object of class "circumplex_ssm_draws" holding draws (the SSM parameter draws, one row per posterior draw, columns e, x, y, a, d, fit, displacement in degrees [0, 360], pole reported as 360), results (the point summaries and credible bounds), and details, whose certified field records the package's displacement-interpretability certification applied to the amplitude credible interval (a_lci / (a_uci - a_lci) >= 0.35): when it fails, the displacement interval is not interpretable and printing adds a note saying so. Printing shows the summary table; summary() adds the analysis details.

See Also

Other ssm functions: plot.circumplex_ci_accuracy(), ssm_analyze(), ssm_analyze_long(), ssm_ci_accuracy(), ssm_parameters(), ssm_parameters_id(), ssm_score(), ssm_sem(), ssm_sem_parameters(), ssm_table(), summary.circumplex_ssm_id()

Other analysis functions: cpm_fit(), cpm_simulate(), ssm_analyze(), ssm_analyze_long(), ssm_ci_accuracy(), ssm_parameters(), ssm_parameters_id(), ssm_score(), ssm_sem(), ssm_sem_parameters(), summary.circumplex_ssm_id()

Examples

```
# Parameter draws (e.g., brms fixed-effect draws b_Intercept, b_cos, b_sin)
set.seed(1)
```

```
draws <- cbind(rnorm(500, 0.4, 0.1), rnorm(500, 0.9, 0.1),
               rnorm(500, -0.3, 0.1))
ssm_draws(draws, type = "parameters")
```

`ssm_ellipse_data` *Centre and covariance of the Cartesian SSM coordinates for an ellipse layer*

Description

Compute the five columns `geom_ssm_ellipse()` draws from: the centre (`x0`, `y0`) of a profile's Cartesian coordinates and the elements `var_x`, `var_y`, `cov_xy` of their 2 by 2 covariance matrix. This is an S3 generic; the method for `ssm_draws()` objects takes the centre from the object's point summaries (`results$x_est`, `results$y_est`, the posterior medians of `x` and `y`) and the covariance from `stats::cov()` of the `x` and `y` columns of the object's draws. The point `geom_ssm_point()` draws from the same object is `a_est` at `d_est`, which need not coincide with those medians. Objects that retain no draws, such as the result of `ssm_analyze()`, have no method and are refused.

Usage

```
ssm_ellipse_data(x, ...)

## S3 method for class 'circumplex_ssm_draws'
ssm_ellipse_data(x, ...)

## Default S3 method:
ssm_ellipse_data(x, ...)
```

Arguments

`x`	An object of class `"circumplex_ssm_draws"`.
`...`	Passed to methods; unused by the shipped method.

Value

A one-row data frame with columns `x0`, `y0`, `var_x`, `var_y`, and `cov_xy`, in the score metric of the coordinates.

See Also

`geom_ssm_ellipse()`, which draws the ellipse, and `ssm_draws()`, which produces the input.

Examples

```
set.seed(1)
draws <- cbind(rnorm(500, 0.4, 0.1), rnorm(500, 0.9, 0.1),
               rnorm(500, -0.3, 0.1))
ssm_ellipse_data(ssm_draws(draws, type = "parameters"))
```

ssm_growth_data *Build the long table for a growth model on SSM coordinates*

Description

Score every row of data as its own circumplex profile and stack the three SSM coordinates, elevation e, x and y, into the long table that the package's growth recipe fits. One input row gives three output rows, one per coordinate, and the output holds each row's id and time value beside the coordinate's name and value. ssm_growth_formula() gives the fit call that takes this table.

Usage

```
ssm_growth_data(data, scales, angles = octants(), id, time)
```

Arguments

data	Required. A data frame or matrix with one row per person per time point, containing the circumplex scales, an id column and a time column. Two rows with the same id and the same time are an error. Time values are compared exactly, with no rounding. The output's id is a factor, which labels each value by as.character() (15 significant digits for most numbers, though a large whole number can keep more). Two distinct ids that share a label, such as 0.1 + 0.2 and 0.3, are an error. A column name that occurs more than once is an error when id, time or a name in scales refers to it.
scales	Required. The variable names or column numbers for the variables in data that contain circumplex scales. They must not select the id or time column, select one column more than once, hold a number below 1, or name a column whose name occurs more than once in data.
angles	Optional. A numeric vector containing the angular displacement of each circumplex scale included in scales, in degrees (default = octants()).
id	Required. The name of the column in data identifying persons. A column name, not a number. Missing values are an error, and so is a list column.
time	Required. The name of the numeric column in data holding each row's time point. A column name, not a number. A column that is not numeric (a Date, factor or character column, among others) is an error, as are missing and infinite values: the growth model fits time as a number, so the caller chooses its origin and unit. id and time must differ from each other and from dv and value, the two column names the output reserves.

Details

Growth in displacement is modeled through x and y, never through the angle itself, so the table carries no amplitude or displacement. A flat profile has x and y at zero to floating-point precision and a defined e, and a profile with a scale entirely missing has NA on all three coordinates. Both keep their rows. Each engine's fit call then drops a row whose value is NA: glmmTMB and brms do so by default, and the nlme call that ssm_growth_formula() prints sets na.action = na.omit for the same effect.

Value

A data frame with `3 * nrow(data)` rows and four columns, named `<id>` (a factor), `<time>` (numeric), `dv` (a factor with levels `e`, `x` and `y`) and `value`. Rows are ordered by input row, then by `dv`. Each `value` is the `Elev`, `Xval` or `Yval` that `ssm_parameters_id()` with `id = NULL` gives that input row.

See Also

Other growth functions: `ssm_growth_formula()`, `ssm_trajectory()`

Examples

```
data("simulated_growth")
long <- ssm_growth_data(
  simulated_growth,
  scales = PANO(),
  id = "person",
  time = "wave"
)
head(long, 6)
```

---

ssm_growth_formula *The joint growth model on SSM coordinates as an engine's formulas*

Description

Return the package's fixed joint growth model in the formula pieces one mixed-model engine takes. The model reads the long table `ssm_growth_data()` builds: one intercept and one linear time slope per coordinate (`e`, `x`, `y`), a person-level random intercept per coordinate with the three intercepts free to correlate, and a separate residual variance per coordinate. Printing the object shows the complete fit call for the engine and, below it, the lines that keep the fixed effects and their covariance from the fit, or the posterior draws for brms. Those are the inputs of the recipe's next step, the trajectory.

Usage

```
ssm_growth_formula(
  engine = c("glmmTMB", "nlme", "brms"),
  time = "wave",
  id = "person"
)

## S3 method for class 'circumplex_growth_formula'
print(x, ...)
```

Arguments

engine	Optional. One of "glmmTMB" (default), "nlme" or "brms". The engine is named only; it is not loaded or called.
time	Optional. The name of the time column in the long table (default "wave"), the same time given to ssm_growth_data().
id	Optional. The name of the person column in the long table (default "person"), the same id given to ssm_growth_data(). time and id must each be one non-empty name, different from each other and from dv and value, the columns the long table reserves. A name that is not syntactic, such as "my wave", is backticked in the formulas, so every name reads as one column and never as formula syntax.
x	An object of class "circumplex_growth_formula".
...	Ignored (S3 consistency).

Details

The model has no options. Adding a covariate, a quadratic time term or another random-effects structure changes the model whose interval coverage the package validated, so the pieces are built for the user to paste and, if they choose, to edit by hand with that in mind. That validation ran the frequentist REML fit. The brms dialect fits the same likelihood under brms's default priors, and its posterior intervals were not part of it. The nlme call sets na.action = na.omit, since nlme::lme() otherwise stops on a row whose value is NA; glmmTMB and brms drop such rows by default.

Value

A list of class "circumplex_growth_formula" with attributes engine, time and id. Its elements are formula objects, one per argument or formula part the engine's fit call takes. For "glmmTMB": formula, value ~ 0 + dv + dv:<time> + us(0 + dv | <id>), and dispformula, ~ 0 + dv. For "nlme": fixed, value ~ 0 + dv + dv:<time>, random, ~ 0 + dv | <id>, and weights, ~ 1 | dv, the form that nlme::varIdent() takes. For "brms": formula, value ~ 0 + dv + dv:<time> + (0 + dv | <id>), and sigma, sigma ~ 0 + dv, the two parts of a brms::bf() call.

See Also

Other growth functions: ssm_growth_data(), ssm_trajectory()

Examples

```
ssm_growth_formula("glmmTMB", time = "wave", id = "person")
ssm_growth_formula("nlme")
ssm_growth_formula("brms")
```

`ssm_parameters` *Calculate Structural Summary Method parameters for a set of scores*

Description

Calculate SSM parameters (without confidence intervals) for a set of scores and generate a data frame with customizable labels for each parameter value. This function requires the input to be a numeric vector (or coercable to one) and returns only the parameters. See `ssm_score()` for a similar function that calculates SSM parameters for each row of a data frame.

Usage

```
ssm_parameters(
  scores,
  angles = octants(),
  prefix = "",
  suffix = "",
  e_label = "Elev",
  x_label = "Xval",
  y_label = "Yval",
  a_label = "Ampl",
  d_label = "Disp",
  f_label = "Fit"
)
```

Arguments

`scores`	Required. A numeric vector (or single row data frame) containing one score for each of a set of circumplex scales.
`angles`	Required. A numeric vector containing the angular displacement of each circumplex scale included in `scores` (in degrees). The closed-form SSM estimator used here equals the ordinary-least-squares cosine fit for equally spaced `angles` (e.g., octants at 45-degree intervals) – more generally, for any angle set satisfying first- and second-harmonic balance. For angle sets violating that balance (generic unequally spaced sets), it is the conventional Gurtman estimator, not a least-squares fit, and the reported fit is then no longer a bounded R-squared in `[0, 1]` (it can fall below 0).
`prefix`	Optional. A string to append to the beginning of all of the SSM parameters' variable names (default = "").
`suffix`	Optional. A string to append to the end of all of the SSM parameters' variable names (default = "").
`e_label`	Optional. A string representing the variable name of the SSM elevation parameter (default = "Elev").
`x_label`	Optional. A string representing the variable name of the SSM x-value parameter (default = "Xval").

y_label — Optional. A string representing the variable name of the SSM y-value parameter (default = "Yval").

a_label — Optional. A string representing the variable name of the SSM amplitude parameter (default = "Ampl").

d_label — Optional. A string representing the variable name of the SSM displacement parameter (default = "Disp").

f_label — Optional. A string representing the variable name of the SSM fit or R-squared value (default = "Fit"). This value is a bounded R-squared in `[0, 1]` when the closed form coincides with the least-squares fit (equally spaced or otherwise harmonic-balanced `angles`; see `angles`).

Value

A data frame containing the SSM parameters calculated from `scores`. For degenerate profiles the undefined parameters are returned as `NA` with a warning: a flat profile (zero variance) has undefined displacement and fit, and a profile with real variance but zero amplitude (i.e., no first-harmonic component) has undefined displacement and a fit of 0. Note that this applies only to amplitudes that are zero up to machine precision; small real amplitudes are always estimated, and their uncertainty is expressed through confidence intervals (see `ssm_analyze()`).

See Also

Other ssm functions: `plot.circumplex_ci_accuracy()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `ssm_table()`, `summary.circumplex_ssm_id()`

Other analysis functions: `cpm_fit()`, `cpm_simulate()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Examples

```
# Manually enter octant scores
scores <- c(0.55, 0.58, 0.62, 0.76, 1.21, 1.21, 1.48, 0.90)
ssm_parameters(scores)

# Customize several of the labels
ssm_parameters(scores, x_label = "LOV", y_label = "DOM")

# Add a prefix to all labels
ssm_parameters(scores, prefix = "IIP_")
```

ssm_parameters_id *Calculate SSM parameters for each person*

Description

Score each person's own circumplex profile through the closed-form SSM transform and return a per-person parameter table. When `id` is `NULL`, every row of `data` is treated as one person's profile (like `ssm_score()`, but returning a fresh table rather than appending columns). When `id` names a column, rows sharing an id (e.g., occasions of intensive longitudinal data) are first averaged within person – each scale's mean uses that person's available (non-missing) rows – and the within-person mean profile is scored.

Usage

```
ssm_parameters_id(data, scales, angles = octants(), id = NULL)
```

Arguments

`data`	Required. A data frame or matrix containing at least circumplex scales, with one row per person or (with `id`) per person-occasion.
`scales`	Required. The variable names or column numbers for the variables in `data` that contain circumplex scales to be analyzed. They must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in `data`.
`angles`	Optional. A numeric vector containing the angular displacement of each circumplex scale included in `scales`, in degrees (default = `octants()`). The closed-form SSM estimator used here equals the ordinary-least-squares cosine fit for equally spaced `angles` – more generally, for any angle set satisfying first- and second-harmonic balance; see `ssm_parameters()`.
`id`	Optional. A single variable name or column number identifying persons. If `NULL` (default), each row is scored as its own person; otherwise rows sharing an id are averaged within person before scoring. Missing id values are an error (a person cannot be silently dropped).

Details

Degenerate profiles keep their row and are reported as `NA`, never silently dropped: a flat (zero-variance) profile has undefined displacement and fit, a profile with real variance but zero first-harmonic amplitude has undefined displacement and a fit of 0, and a person with a completely missing scale has an undefined profile (all parameters `NA`). The `na_rate` column exposes each person's share of missing scale cells so missingness is visible alongside its consequences.

Value

A data frame of class `"circumplex_ssm_id"` with one row per person, in order of first appearance: the id column (named after `id`, or `id` when `NULL`), `n_obs` (rows contributing to that person), `na_rate`

(proportion of missing scale cells among those rows), and the SSM parameters `Elev`, `Xval`, `Yval`, `Ampl`, `Disp` (degrees in [0, 360], with the 0/360 pole reported as 360 per the package's LM = 360 convention), and `Fit`. Use `summary.circumplex_ssm_id()` for group-level summaries with circular statistics for displacement.

See Also

Other ssm functions: `plot.circumplex_ci_accuracy()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `ssm_table()`, `summary.circumplex_ssm_id()`

Other analysis functions: `cpm_fit()`, `cpm_simulate()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Examples

```
data("aw2009")
ssm_parameters_id(
  aw2009,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
)
```

---

ssm_plot_circle *Create a Circular Plot of SSM Results*

Description

Take in the results of a Structural Summary Method analysis and plot the point and interval estimate for each row (e.g., group or measure) in a circular space quantified by displacement and amplitude.

Usage

```
ssm_plot_circle(
  ssm_object,
  amax = NULL,
  legend_font_size = 12,
  scale_font_size = 12,
  drop_lowfit = FALSE,
  repel = FALSE,
  angle_labels = NULL,
  palette = "Set2",
  vary_shapes = FALSE,
  path = FALSE,
  ...
)
```

Arguments

ssm_object	Required. The output of ssm_analyze().
amax	A positive real number corresponding to the radius of the circle. It is used to scale the amplitude values and will determine which amplitude labels are drawn.
legend_font_size	A positive real number corresponding to the size (in pt) of the text labels in the legend (default = 12).
scale_font_size	A positive real number corresponding to the size (in pt) of the text labels for the amplitude and displacement scales (default = 12).
drop_lowfit	A logical determining whether profiles with low model fit (<.70) should be omitted or plotted with dashed borders (default = FALSE).
repel	A logical determining whether each profile is labelled with a repelled text label (placed on the circumplex canvas by coord_circumplex(), so labels avoid overlapping each other and the points) instead of distinguished by colour and a legend (default = FALSE). Requires the **ggrepel** package.
angle_labels	A character vector specifying text labels to plot around the circle for each scale. Can also specify NULL to default to numerical angle labels or a vector of empty strings ("") to hide the labels. If not NULL, must have the same length and ordering as the angles argument to ssm_analyze(). (default = NULL)
palette	A string corresponding to the palette to be used from ColorBrewer for the color and fill aesthetics. If set to NULL, all points will appear blue and no legend will be there (useful for showing the coverage of a high number of variables).
vary_shapes	A logical determining whether profiles should each get their own shape or vary only by fill color. This only works when the number of profiles is five or less. (default = FALSE)
path	A logical determining whether each series' movement across occasions is drawn as an arrowed path on the circle (default = FALSE). Requires an SSM object with occasions, from ssm_analyze() with the occasions argument or from ssm_analyze_long(); supplying TRUE for any other object is an error. Occasions are connected in the order they were supplied, never alphabetically, and the path is drawn the short way across the 0/360 boundary. An occasion whose displacement is undefined (a flat or zero-amplitude profile) breaks the path rather than being interpolated through. See geom_ssm_path() for the underlying layer.
...	Not used. Supplying an unrecognized argument produces a warning.

Details

The amplitude axis and its labels are drawn along the midpoint of the widest gap between displacement spokes that holds no plotted point estimate (ties go to the smallest midpoint; a point on a spoke counts as in both gaps next to it). When every gap holds a point, the axis goes in the widest gap, as coord_circumplex() places it by default.

Value

A ggplot variable containing a completed circular plot.

See Also

Other visualization functions: plot.circumplex_ci_accuracy(), ssm_plot_contrast(), ssm_plot_curve(), ssm_plot_trajectory()

Examples

```
# `boots` is lowered from its default of 2000 throughout these examples so
# they run quickly; a reported analysis should use the default.


data("jz2017")
res <- ssm_analyze(
  jz2017,
  scales = 2:9,
  measures = c("NARPD", "ASPD"),
  boots = 200
)
ssm_plot_circle(res)
```

---

ssm_plot_contrast *Create a Difference Plot of SSM Contrast Results*

Description

Take in the results of a Structural Summary Method analysis with pairwise contrasts and plot the point and interval estimates for each parameter's contrast (e.g., between groups or measures).

Usage

```
ssm_plot_contrast(
  ssm_object,
  drop_xy = FALSE,
  sig_color = "#fc8d62",
  ns_color = "white",
  linesize = 1.25,
  fontsize = 12,
  ...
)
```

Arguments

ssm_object	Required. The results output of ssm_analyze().
drop_xy	A logical determining whether the X-Value and Y-Value parameters should be removed from the plot (default = FALSE).
sig_color	Optional. A string corresponding to the color to use to denote significant contrasts (default = "#fc8d62").

ns_color	Optional. A string corresponding to the color to use to denote non-significant contrasts (default = "white").
linesize	Optional. A positive number corresponding to the size of the point range elements in mm (default = 1.5).
fontsize	Optional. A positive number corresponding to the size of the axis labels, numbers, and facet headings in pt (default = 12).
...	Not used. Supplying an unrecognized argument produces a warning.

Value

A ggplot variable containing difference point-ranges faceted by SSM parameter. An interval that does not contain the value of zero has p<.05.

See Also

Other visualization functions: plot.circumplex_ci_accuracy(), ssm_plot_circle(), ssm_plot_curve(), ssm_plot_trajectory()

Examples

```
# `boots` is lowered from its default of 2000 throughout these examples so
# they run quickly; a reported analysis should use the default.


data("jz2017")
res <- ssm_analyze(
  jz2017,
  scales = 2:9,
  measures = c("NARPD", "ASPD"),
  contrast = TRUE,
  boots = 200
)
ssm_plot_contrast(res)
```

---

ssm_plot_curve *Create a Curve Plot of SSM Results*

Description

Take in the results of a Structural Summary Method analysis and plot the scores by angle and the estimated SSM curve.

Usage

```
ssm_plot_curve(
  ssm_object,
  angle_labels = NULL,
  base_size = 11,
  drop_lowfit = FALSE,
  ...
)
```

Arguments

ssm_object	Required. The results output of ssm_analyze().
angle_labels	Optional. Either NULL or a character vector that determines the x-axis labels. If NULL, the labels will be the angle numbers. If a character vector, must be the same length and in the same order as the angles argument to ssm_analyze() (default = NULL).
base_size	Optional. A positive number corresponding to the base font size in pts (default = 11).
drop_lowfit	Optional. A logical indicating whether to omit profiles with low fit (<.70) or include them with dashed lines (default = FALSE).
...	Not used. Supplying an unrecognized argument produces a warning.

Value

A ggplot object depicting the SSM curve(s) of each profile.

See Also

Other visualization functions: plot.circumplex_ci_accuracy(), ssm_plot_circle(), ssm_plot_contrast(), ssm_plot_trajectory()

Examples

```
# `boots` is lowered from its default of 2000 throughout these examples so
# they run quickly; a reported analysis should use the default.


data("jz2017")
res <- ssm_analyze(
  jz2017,
  scales = 2:9,
  measures = 10:13,
  boots = 200
)
ssm_plot_curve(res)
ssm_plot_curve(res, angle_labels = PANO())
```

`ssm_plot_trajectory` *Create a Trajectory Plot of SSM Results Over Time*

Description

Plot each Structural Summary Method parameter against time, one facet per parameter, with its confidence interval as a band. This is a Cartesian diagnostic plot, not a circumplex figure: the horizontal axis is time, not angle.

Usage

```
ssm_plot_trajectory(x, ...)

## Default S3 method:
ssm_plot_trajectory(x, ...)

## S3 method for class 'circumplex_ssm'
ssm_plot_trajectory(x, drop_xy = FALSE, base_size = 11, na.rm = TRUE, ...)

## S3 method for class 'data.frame'
ssm_plot_trajectory(
  x,
  time,
  drop_xy = FALSE,
  base_size = 11,
  na.rm = TRUE,
  ...
)

## S3 method for class 'circumplex_ssm_trajectory'
ssm_plot_trajectory(
  x,
  time = attr(x, "time"),
  drop_xy = FALSE,
  base_size = 11,
  na.rm = TRUE,
  ...
)
```

Arguments

`x`	An SSM results object produced by `ssm_analyze()` with the `occasions` argument or by `ssm_analyze_long()`, or a trajectory table (a data frame) as described above.
`...`	Not used. Supplying an unrecognized argument produces a warning.
`drop_xy`	A logical determining whether the X-value and Y-value panels should be omitted (default = `FALSE`), leaving elevation, amplitude, and displacement.

base_size	A positive number determining the base font size of the plot (default = 11).
na.rm	A logical determining whether time points that cannot be plotted (no defined displacement) are dropped silently (default = TRUE) or with a warning naming how many were removed (FALSE).
time	A string naming the numeric time column of a trajectory table. Required for the data frame method; unused for SSM objects. An object from ssm_trajectory() carries the name as its time attribute, which its method reads by default.

Details

Two kinds of input are accepted, and they differ only in their time axis:

- An **SSM results object** with occasions, from ssm_analyze() with the occasions argument or from ssm_analyze_long(). Occasions are discrete and ordered, so the axis is discrete.
- A **trajectory table**: a data frame with one row per time point, a numeric time column named by time, and the columns a_est, a_lci, a_uci, d_est, d_lci, and d_uci (optionally the e_*, x_*, and y_* triples, and a logical certified column). This is the shape a model-based workflow assembles by evaluating a fitted growth model at each time point and passing the draws through ssm_draws(); see vignette("growth-ssm-analysis"). The axis is continuous, so unequally spaced time points are drawn at their actual spacing.

Both paths share one implementation of the displacement unwrap and the certification marking described below.

The displacement panel is drawn on an *unwrapped* branch, so a profile whose displacement crosses the 0/360 boundary renders as one continuous path rather than jumping a full turn. Values on that panel may therefore fall outside [0, 360). The lower confidence bound is placed below the estimate by the counterclockwise angle from the lower bound to the estimate, and the upper bound is placed above the lower bound by the counterclockwise angle from the lower bound to the upper bound. An interval is therefore always drawn as the counterclockwise arc from its lower to its upper bound, even when that arc is wider than a half-turn. Unwrapping assumes the profile rotates less than a half-turn between consecutive time points at which its displacement is defined – no data can verify this, so time points that are far apart, or a series with a gap, should be read with that in mind.

Occasions appear in the order they were supplied to ssm_analyze() (or in the occasion factor's level order for ssm_analyze_long()), never in alphabetical order.

On the displacement panel, a time point whose amplitude confidence interval is too close to zero for its displacement to be interpretable is drawn as a hollow point; see ssm_analyze() for the certification rule. The line segment on either side of such a point is drawn dashed: it touches a time point whose displacement is not interpretable, so the direction of change along it is not to be read. The segment is still drawn, so the unwrapped branch stays visible across the point. The legend shows the two marks together, a filled point on a solid line for an interpretable time point and a hollow point on a dashed line otherwise. For an SSM object the verdict is computed from the amplitude interval; for a trajectory table it is read from the optional certified column. An NA in that column is drawn as not interpretable. When the column is absent, or holds no verdict at all, no interpretability claim is made or shown. A profile with no defined displacement at all (a flat profile) leaves a gap in that panel, and no segment spans the gap.

A contrast row is never plotted as a time point – it is a difference, not a time point. Use ssm_plot_contrast() for it.

Value

A ggplot object depicting each SSM parameter's trajectory over time, with confidence bands.

See Also

`geom_ssm_path()` and `ssm_plot_circle(path = TRUE)`, which draw the same change across occasions as movement on the circumplex canvas rather than as parameter-by-time panels.

Other visualization functions: `plot.circumplex_ci_accuracy()`, `ssm_plot_circle()`, `ssm_plot_contrast()`, `ssm_plot_curve()`

Examples

```
# `boots` is lowered from its default of 2000 throughout these examples so
# they run quickly; a reported analysis should use the default.


data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
t1 <- jz2017[, scales]
t1$id <- seq_len(nrow(t1))
t1$occasion <- "T1"
t2 <- t1
t2$occasion <- "T2"
res <- ssm_analyze_long(rbind(t1, t2),
  scales = scales, id = "id", occasion = "occasion",
  boots = 200
)
ssm_plot_trajectory(res)
ssm_plot_trajectory(res, drop_xy = TRUE)


# A model-based trajectory table, plotted on a continuous time axis
trajectory <- data.frame(
  wave = 0:4,
  a_est = c(0.60, 0.55, 0.52, 0.58, 0.63),
  a_lci = c(0.48, 0.43, 0.40, 0.46, 0.51),
  a_uci = c(0.72, 0.67, 0.64, 0.70, 0.75),
  d_est = c(350, 355, 2, 8, 12),
  d_lci = c(340, 345, 352, 358, 2),
  d_uci = c(0, 5, 12, 18, 22),
  certified = c(TRUE, TRUE, FALSE, TRUE, TRUE)
)
ssm_plot_trajectory(trajectory, time = "wave")
```

`ssm_score` *Calculate SSM parameters by row and add results as new columns*

Description

Calculate the SSM parameters for each row of a data frame and add the results as additional columns. This can be useful when the SSM is being used for the description or visualization of individual data points rather than for statistical inference on groups of data points.

Usage

```
ssm_score(data, scales, angles = octants(), append = TRUE, ...)
```

Arguments

`data`	Required. A data frame or matrix containing at least circumplex scales.
`scales`	Required. The variable names or column numbers for the variables in `data` that contain circumplex scales to be analyzed. They must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in `data`.
`angles`	Required. A numeric vector containing the angular displacement of each circumplex scale included in `scales` (in degrees). The closed-form SSM estimator used here equals the ordinary-least-squares cosine fit for equally spaced `angles` (e.g., octants at 45-degree intervals) – more generally, for any angle set satisfying first- and second-harmonic balance. For angle sets violating that balance (generic unequally spaced sets), it is the conventional Gurtman estimator, not a least-squares fit, and the reported fit is then no longer a bounded R-squared in `[0, 1]` (it can fall below 0).
`append`	Optional. A logical indicating whether to append the output to `data` or simply return the output (default = "TRUE").
`...`	Optional. Additional **named** arguments passed to `ssm_parameters()`, such as `prefix` and `suffix`; each must be a single string. Unnamed or non-scalar arguments raise an error.

Value

A data frame containing `.data` plus six additional columns containing the SSM parameters (calculated rowwise).

See Also

Other ssm functions: `plot.circumplex_ci_accuracy()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_sem()`, `ssm_sem_parameters()`, `ssm_table()`, `summary.circumplex_ssm_id()`

Other analysis functions: `cpm_fit()`, `cpm_simulate()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_sem()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Examples

```
data("aw2009")
ssm_score(
  aw2009,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
)
```

`ssm_sem` *Perform SEM-based (latent-variable) SSM analyses*

Description

Estimate the Structural Summary Method profile that one or more external measures show against the *latent* circumplex content of a set of scales – the disattenuated analog of the correlation-based `ssm_analyze()` – from a structural equation model with the scale angles held fixed at their theoretical values. The measurement model is generated by `ssm_sem_syntax()` and fitted with **lavaan** on raw covariances; confidence intervals for all SSM parameters are constructed in-package by propagating draws of the model's free parameters through the profile and SSM transforms and applying the same percentile/circular-quantile machinery as `ssm_analyze()`. No lavaan delta-method or percentile interval is ever used for amplitude or displacement (their intervals must respect the angular branch cut, which lavaan's `:=` machinery does not).

Usage

```
ssm_sem(
  data,
  scales,
  angles = octants(),
  measures = NULL,
  grouping = NULL,
  contrast = FALSE,
  model = c("scaled", "strict"),
  invariance = NULL,
  invariance_alpha = 0.05,
  ci_method = c("mvn", "boot"),
  boots = 2000,
  interval = 0.95,
  estimator = "MLR",
  se = "robust.huber.white",
  missing = c("listwise", "fiml"),
  parallel = "no",
  ncpus = 1,
  ...
)
```

Arguments

data	Required. A data frame or matrix containing at least the circumplex scales and measures.
scales	Required. A character vector of column names, or a numeric vector of column indexes, from data that contains the circumplex scale scores. It must not select one column twice, hold a number below 1, or name a column whose name occurs more than once in data.
angles	Optional. A numeric vector containing the angular displacement of each circumplex scale included in scales, in degrees (default = octants()). The angles are fixed theoretical constants in the measurement model, never free parameters.
measures	Optional with grouping, required otherwise. A character vector (or numeric indexes) of one or more columns of data to be related to the latent circumplex content (the disattenuated correlation path). With grouping and measures = NULL, the latent MEAN path is analyzed instead: each group's model-implied latent mean profile, on the raw-score metric. (A single-group latent mean profile is not a product: factor means are not identified in one group.) A measure must not carry the name of one of the scales, since lavaan reaches variables by name.
grouping	Optional. A string naming the column of data indicating group membership. With grouping, the fixed-angle measurement model is fitted as a multi-group model under an invariance ladder (configural, then metric, then – when required – scalar), and the latent SSM profiles are reported per group. The FIRST factor level is the reference group. With measures = NULL and grouping, the latent MEAN path is analyzed (each group's model-implied latent mean profile).
contrast	Optional. A logical (default = FALSE) requesting a difference of latent SSM parameters, always second minus first with the displacement contrast in (-180, 180] degrees. Without grouping: exactly two measures (second measure minus first). With grouping: exactly two groups (second factor level minus first) and at most one measure – one measure gives the group contrast on that measure's latent profile, none gives the latent mean-path group contrast. The group contrast is invariance-gated; see invariance.
model	Optional. The measurement-model tier passed to ssm_sem_syntax(): "scaled" (default) or "strict".
invariance	Optional. The highest invariance rung to fit and REPORT ("configural", "metric", "scalar", or "strict_residuals"). NULL (default) uses the path's required rung: "metric" for the measure-profile path, "scalar" for the latent mean path. A group contrast is only computed if EVERY tested rung up through the path's required rung is retained by lavaan's own nested test (the scaled difference test under robust estimators) – a rejection at any lower rung rejects the constraints the contrast would be computed under. Rungs fitted ABOVE the required one are reported but never gate the contrast. On rejection, the returned object states the non-comparison and reports each group's separate configural profile instead (no contrast is rendered by any method). Under the strict tier the metric rung is vacuous (all loadings fixed) and is reported as such. The ladder table also reports dcfi, the change in CFI from the previous fitted rung (NA for configural and for the strict tier's vacuous metric rung), as a labeled

secondary criterion: Cheung and Rensvold's (2002) general rule rejects an invariance step when CFI drops by more than .01 (their alpha = .01). It is reported and never gates: `comparable`, the verdict, and the fit the estimation layer consumes are decided by the nested test alone, and the two criteria can legitimately disagree (a change in CFI is insensitive to sample size where the nested test is not). The retain/reject label is set **only inside the envelope that simulation covers**: exactly two groups, ML estimation, and a plain (non-robust) CFI. `print()` shows the `dcfi` and `cr` columns and a short note only inside it, and only when a rung has a `dcfi` value. Three separate things put a fit outside it. `dcfi_scope` in the returned `invariance` element records the number of groups, the estimator as lavaan reports it (`"ML"` for `"MLR"` and `"MLM"`), whether it is ML estimation and whether the CFI is plain, and the conditions that apply follow from those fields. A robust estimator – the default `"MLR"`, or `"MLM"` – makes lavaan report a robust CFI; so does `missing = "fiml"`, even under `estimator = "ML"`, so plain ML is necessary for the label but not sufficient. `"GLS"`, `"WLS"`, `"ULS"` and `"DWLS"` are not ML estimation at all, though their CFI is plain-named. And more than two groups is outside the simulation whatever the estimator. In each case the `dcfi` value stays in the returned ladder table with no label attached, and `print()` shows neither the value nor a note.

Cheung and Rensvold simulated two groups, ML estimation, multivariate normal data, and Type I error only; robust CFI variants were not in their study, so no cutoff here was validated for one.

`invariance_alpha`

Optional. The alpha level for the invariance gating decision (default = 0.05). The gate is a modeling decision with a default test, not an oracle; the invariance table is always returned so other criteria can be applied.

`ci_method` Optional. How to generate parameter replicates: `"mvn"` (default) draws from a multivariate normal with lavaan's asymptotic covariance of the free parameters (fast; one model fit); `"boot"` refits the model on `boots` bootstrap resamples via `lavaan::bootstrapLavaan()` (slow; robust to the normal approximation). Both engines feed the same in-package interval machinery.

`boots` Optional. A single positive whole number indicating how many draws or bootstrap refits to use (default = 2000).

`interval` Optional. A single number between 0 and 1 (exclusive) indicating the confidence level (default = 0.95).

`estimator` Optional. The lavaan estimator (default = "MLR": maximum likelihood with robust "Huber-White" standard errors and a scaled test statistic, the standard choice for the skewed distributions typical of circumplex scale scores). The parameter estimates are identical to `"ML"`; what changes is the covariance the `"mvn"` engine propagates (already robust via `se`) and the test statistic behind the global fit indices that `print()` reports (robust/scaled versions are used when available).

`se` Optional. The lavaan standard-error method for the fitted model (default = "robust.huber.white", the sandwich estimator). This does not affect the parameter estimates, only the covariance the `"mvn"` engine propagates: the fixed-angle measurement model is an approximation for real data, and the package's coverage validation found that sandwich-based draws keep the intervals calibrated

for the model-conditional estimand under that misspecification where the plain ML covariance undercovers (displacement ~0.88 instead of 0.95, not improving with n). Set `se = "standard"` for the classical ML covariance.

`missing` Optional. Either `"listwise"` (default; complete cases) or `"fiml"` (full-information maximum likelihood via lavaan's `missing = "ml"`).

`parallel, ncpus` Optional. Passed to `lavaan::bootstrapLavaan()` when `ci_method = "boot"` (defaults `"no"` and 1): the bootstrap refits are independent and can be distributed across cores. Results for a given `set.seed()` are reproducible regardless of these settings (the seed lavaan receives drives its own parallel-safe RNG streams). Ignored by the `"mvn"` engine.

`...` Optional. Additional arguments passed to `lavaan::cfa()` (e.g., `bounds` or estimator-control settings).

Details

The latent profile of a measure is its vector of model-implied *disattenuated* correlations with each scale's common (circumplex) content: the scale's error and unique parts are removed from the denominator, and the covariance is restricted to common content in the numerator. All latent quantities are conditional on the fixed-angle measurement model being adequate: global fit is reported by `print()`, and a poorly fitting measurement model makes the latent SSM parameters uninterpretable, not merely imprecise. The fixed angles are theoretical claims, not estimates (use `cpm_fit()` to examine an instrument's real geometry). Latent displacement is the first-harmonic direction of the saturation-modulated disattenuated profile – heterogeneous scale saturations rotate it exactly as they rotate the observed displacement; the latent layer removes the reliability modulation, nothing more. Model-implied disattenuated correlations at or beyond 1 indicate misspecification and are refused rather than summarized.

The point estimates and intervals are reported for elevation, x-value, y-value, amplitude, and displacement (no standard errors are printed anywhere, matching the package's estimate-plus-interval reporting surface). Unlike `ssm_analyze()`'s closed-form estimator, the latent transform is the ordinary-least-squares projection onto the cosine basis, so for unequally spaced angles the two functionals genuinely differ (they coincide exactly for equally spaced angles, and more generally under first- and second-harmonic balance); under OLS the fit value is a bounded R-squared in `[0, 1]` at any spacing.

With `grouping`, the latent contrast this function computes and the observed contrast that `ssm_analyze()` computes answer different questions and are not substitutes. The *observed* contrast (`ssm_analyze()` with grouping) asks whether the groups' *measured* profiles differ: it is a difference of SSM parameters computed from each group's observed scores or correlations. It confounds structural difference, differential reliability, and measurement non-invariance – that is a property of its estimand, documented rather than a defect, and it requires no invariance assumption. The *latent* contrast (`ssm_sem()` with grouping) asks whether the groups' *constructs* differ, granted the instrument measures the same thing in both groups: it is a contrast on latent SSM parameters computed under cross-group equality constraints, disattenuated and conditional on measurement invariance. When the required invariance rung is rejected the latent contrast is not "more principled" – it is misspecified, and the function therefore returns an explicit non-comparison (the verdict plus each group's separate configural profile; no contrast is computed or rendered by any method). Neither estimand replaces the other; they answer different questions and can legitimately disagree.

The displacement contrast is reported as the second group level minus the first, in `(-180, 180]` degrees, with branch-aligned circular intervals (endpoints may legitimately exceed +/-180 degrees near the boundary). Under the scaled tier the general-plane covariances are fixed to zero in all groups at all rungs (a stationarity-type assumption): a cross-group difference in a general factor's lean into the plane surfaces as misfit, and the strict tier is the tier that can express it. Under the strict tier the metric rung is vacuous (all loadings fixed) and is reported as such.

Value

A `circumplex_ssm_sem` object (a subclass of `circumplex_ssm`, so `ssm_table()` and the `ssm_plot_*` functions work on it), containing `results` (estimates and intervals), `scores` (the latent profile vectors), `details`, `call`, plus `sem` (the fitted lavaan model: the gate rung's fit for grouped analyses, or the configural fit when the gate was rejected), `invariance` (for grouped analyses: the ladder table – including the `dcfi` column and its `cr` retain/reject label, `NA` outside the criterion's validated scope – the `comparable` flag, the verdict text, the gate and required rungs, the alpha used, and `dcfi_scope` recording that scope; `NULL` for single-group analyses), and `model` (tier, generated syntax for single-group fits, and the OLS projection weights).

Inadmissible parameter draws (a nonpositive common-part or measure variance, or a disattenuated correlation at or beyond 1) are dropped whole with a warning naming the causes; if more than 5% of draws are inadmissible the analysis stops with advice to use `ci_method = "boot"` or revise the model. Degenerate profiles (flat or zero-amplitude) keep the same per-parameter `NA` contract as `ssm_analyze()`.

Reproducibility

This function consumes R's random number stream for both `ci_method` settings (`"mvn"` through the package's own draws; `"boot"` through a seed handed to lavaan's bootstrap). Call `set.seed()` immediately before `ssm_sem()` for reproducible confidence intervals.

References

Cheung, G. W., & Rensvold, R. B. (2002). Evaluating goodness-of-fit indexes for testing measurement invariance. *Structural Equation Modeling*, 9(2), 233-255. (The `dcfi` secondary criterion. Their p. 251 sentence states the direction of the -.01 rule backwards relative to their own Table 5, whose critical values are the 1% lower tails of the simulated null distributions; this package follows the simulation.)

See Also

`ssm_analyze()` for the observed-score SSM, `ssm_sem_syntax()` for the generated measurement model, and `ssm_sem_parameters()` to reuse a lavaan fit you have modified or fitted yourself.

Other ssm functions: `plot.circumplex_ci_accuracy()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem_parameters()`, `ssm_table()`, `summary.circumplex_ssm_id()`

Other analysis functions: `cpm_fit()`, `cpm_simulate()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Examples

```
data("jz2017")
set.seed(12345)
res <- ssm_sem(
  jz2017,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO"),
  measures = "NARPD",
  boots = 500
)
res
summary(res)
```

`ssm_sem_parameters` *Calculate latent SSM parameters from a fitted lavaan measurement model*

Description

The low-level adapter behind `ssm_sem()`: take an already fitted **lavaan** model of a fixed-angle circumplex measurement structure (as generated by `ssm_sem_syntax()`, possibly user-modified – e.g., a partial-invariance respecification) and compute latent SSM parameter estimates with in-package confidence intervals. Compatibility with the expected parameter structure is checked structurally (the named loading, factor-covariance, and measure-covariance parameters must be present), not by provenance.

Usage

```
ssm_sem_parameters(
  fit,
  scales,
  angles = octants(),
  measures = NULL,
  ci_method = c("mvn", "boot"),
  boots = 2000,
  interval = 0.95,
  contrast = FALSE,
  parallel = "no",
  ncpus = 1
)
```

Arguments

`fit`	Required. A fitted lavaan object whose model preserves the `ssm_sem_syntax()` parameter structure (factors `g`, `cx`, `cy`; the measures covarying with them). For `ci_method = "mvn"`, fit the model with robust (sandwich) standard errors (lavaan's

se = "robust.huber.white", ssm_sem()'s default) so the propagated covariance stays valid when the fixed-angle model is an approximation; see the se argument of ssm_sem().

scales — Required. A character vector with the scale (indicator) names, in the same order as angles. Each name must appear only once, and NA at most once.

angles — Optional. A numeric vector of the scales' theoretical angles in degrees (default = octants()). Must be the angles the model was generated with.

measures — Optional for multi-group fits, required otherwise. A character vector of the measure names; NULL on a multi-group fit selects the latent MEAN path (the fit must carry the mean structure: scale intercepts and latent means).

ci_method, boots, interval, contrast, parallel, ncpus — See ssm_sem(). Note that for a multi-group fit the CONTRAST DIRECTION (and the group labels in the output) follows the fit's own group order – lavaan's default is order of appearance in the data unless group.label was supplied at fitting time – so read the direction from the output's Group column, not from factor-level conventions.

Details

Important: multi-group fits are supported here as the partial-invariance escape hatch, and this path *bypasses* the invariance gating that ssm_sem() applies. Where ssm_sem() fits a configural-metric-scalar ladder and refuses a latent group contrast when the required rung is rejected, ssm_sem_parameters() computes the contrast from whatever multi-group fit you supply without testing invariance at all. You own the comparability claim: the groups are compared on this instrument's latent metric only to the extent the model you fitted makes them comparable.

Value

A circumplex_ssm_sem object; see ssm_sem().

Reproducibility

This function consumes R's random number stream for both ci_method settings ("mvn" through the package's own draws; "boot" through a seed handed to lavaan's bootstrap). Call set.seed() immediately before ssm_sem() for reproducible confidence intervals.

See Also

Other ssm functions: plot.circumplex_ci_accuracy(), ssm_analyze(), ssm_analyze_long(), ssm_ci_accuracy(), ssm_draws(), ssm_parameters(), ssm_parameters_id(), ssm_score(), ssm_sem(), ssm_table(), summary.circumplex_ssm_id()

Other analysis functions: cpm_fit(), cpm_simulate(), ssm_analyze(), ssm_analyze_long(), ssm_ci_accuracy(), ssm_draws(), ssm_parameters(), ssm_parameters_id(), ssm_score(), ssm_sem(), summary.circumplex_ssm_id()

Examples

```
data("jz2017")
scales <- c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
syn <- ssm_sem_syntax(scales = scales, angles = octants(), measures = "NARPD")
# Robust (sandwich) SEs so the mvn engine propagates a
# misspecification-consistent covariance (ssm_sem()'s default)
fit <- lavaan::cfa(syn, data = jz2017, se = "robust.huber.white")
set.seed(12345)
ssm_sem_parameters(fit, scales = scales, measures = "NARPD", boots = 500)
```

---

ssm_sem_syntax *Generate lavaan syntax for a fixed-angle circumplex measurement model*

Description

Emit **lavaan** model syntax for a structural-equation-model formulation of the Structural Summary Method, with the circumplex scale angles held fixed at their theoretical values. This is the syntax-generation layer of the SEM-based SSM (see the package's design notes); it is a pure function of its arguments and does **not** require **lavaan** to be installed (only fitting the emitted model does).

Usage

```
ssm_sem_syntax(
  instrument = NULL,
  scales = NULL,
  angles = NULL,
  measures = NULL,
  model = c("scaled", "strict"),
  n_groups = 1,
  invariance = c("configural", "metric", "scalar", "strict_residuals"),
  include_defined = NULL
)
```

Arguments

instrument	Optional. A `circumplex_instrument` object; its scale abbreviations and angles are used. Supply this or both `scales` and `angles`.
scales	Optional. A character vector of circumplex scale (column) names, each appearing only once, and `NA` at most once. Ignored when `instrument` is supplied.
angles	Optional. A numeric vector of scale angles in degrees, the same length as `scales`. Ignored when `instrument` is supplied.

`measures` Optional. Either `NULL` or a character vector of external measure (column) names to relate to the circumplex factors. None may be one of the scale names, whether those come from `scales` or from `instrument`.

`model` Optional. The measurement-model tier: `"scaled"` (default) or `"strict"`. See Details.

`n_groups` Optional. A single positive whole number: the number of groups for a multi-group model. `1` (default) emits the single-group model. When `n_groups >= 2` the generator emits multi-group syntax with a mean structure and the invariance constraints selected by `invariance`; fitting it requires `lavaan::cfa()` called with `group =` a grouping variable whose number of levels equals `n_groups`. Group ordering follows the factor-level order of that variable: the **first** level is the reference group, whose factor metric (and, from the scalar rung, whose latent means) is fixed.

`invariance` Optional. The cross-group invariance rung to emit when `n_groups >= 2`: `"configural"` (default), `"metric"`, `"scalar"`, or `"strict_residuals"`. Must be left at its default when `n_groups == 1` (supplying it there is an error). Under the `"strict"` tier the `"metric"` rung is vacuous (all loadings are fixed) and emits the configural structure with an explanatory comment. See the package design notes (section 6.2) for the adapted fixed-angle invariance ladder.

`include_defined` Optional. (Single-group models only: for `n_groups >= 2` the lines are unavailable and an explicit `TRUE` is ignored with a warning.) Whether to append inspection-only `:=` definitions of the covariance-metric `(e, x, y)` coordinates for each measure. `NULL` (default) emits them automatically under the `"strict"` tier when at least one measure is present (there they are linear), and omits them otherwise. `TRUE` forces emission (an error under `"scaled"`, where they would be nonlinear); `FALSE` always suppresses them.

Details

Two model tiers are available. The `"scaled"` tier frees a general saturation and a circumplex saturation per scale while fixing each scale's angle; the circumplex plane is held isotropic and orthogonal, and the general factor is held orthogonal to the plane (freeing the general-plane covariances alongside free saturations makes the model locally unidentified exactly at zero covariance, so they cannot be estimated in this tier). The `"strict"` tier fixes every loading to the unit cosine pattern `(1, cos(angle), sin(angle))` and frees the 3x3 factor covariance matrix – including the general-plane covariances, making it the tier that can model a general factor leaning into the plane. The angles enter only as evaluated cosine and sine constants; no angle is ever a free parameter.

The returned string always carries a `weights` attribute: the ordinary-least-squares projection matrix that maps a profile vector to the structural summary coordinates `(e, x, y)`. For equally spaced angles this equals the conventional closed-form estimator; for unequally spaced angles it is the least-squares projection and can differ. The emitted syntax never contains `:=` definitions for amplitude or displacement: those are nonlinear functions whose confidence intervals must be constructed in-package, not by **lavaan**'s delta method (which ignores the angular branch cut).

Value

A single character string of **lavaan** model syntax, with attributes `angles` (a `circumplex_degree` vector), `scales`, `model`, `weights` (the 3-by-p OLS projection matrix, rows `e`, `x`, `y`), `n_groups`, and (when `n_groups >= 2`) `invariance`.

See Also

`ssm_analyze()` for the observed-data SSM.

Examples

```
# Octant instrument, default scaled tier
syn <- ssm_sem_syntax(scales = paste0("s", 1:8), angles = octants())
attr(syn, "weights")
cat(syn)
```

---

ssm_table *Create HTML table from SSM results or contrasts*

Description

Take in the results of an SSM analysis and return an HTML table with the desired formatting.

Usage

```
ssm_table(ssm_object, caption = NULL, drop_xy = FALSE, render = TRUE)
```

Arguments

`ssm_object`	Required. The results output of `ssm_analyze()`.
`caption`	A string to be displayed above the table (default = NULL).
`drop_xy`	A logical indicating whether the x-value and y-value parameters should be omitted from the output (default = FALSE).
`render`	A logical indicating whether the table should be displayed in the RStudio viewer or web browser (default = TRUE).

Value

A data frame containing the information for the HTML table. As a side-effect, may also output the HTML table to the web viewer.

See Also

Other ssm functions: `plot.circumplex_ci_accuracy()`, `ssm_analyze()`, `ssm_analyze_long()`, `ssm_ci_accuracy()`, `ssm_draws()`, `ssm_parameters()`, `ssm_parameters_id()`, `ssm_score()`, `ssm_sem()`, `ssm_sem_parameters()`, `summary.circumplex_ssm_id()`

Other table functions: `html_render()`

Examples

```
# `boots` is lowered from its default of 2000 throughout these examples so
# they run quickly; a reported analysis should use the default.


# Load example data
data("jz2017")

# Create table of profile results
res <- ssm_analyze(
  jz2017,
  scales = 2:9,
  measures = c("NARPD", "ASPD"),
  boots = 200
)
ssm_table(res)

# Create table of contrast results
res <- ssm_analyze(
  jz2017,
  scales = 2:9,
  measures = c("NARPD", "ASPD"),
  contrast = TRUE,
  boots = 200
)
ssm_table(res)
```

`ssm_trajectory` *Trajectory of SSM parameters from a fitted growth model*

Description

Turn the fixed effects of the joint growth model that `ssm_growth_formula()` builds into a table of Structural Summary Method parameters at each time point, with intervals and the displacement certification. Two input shapes are accepted. The frequentist shape is `coef` and `vcov`, the fixed effects and their covariance from a glmmTMB or nlme fit. The function draws `n_draws` coefficient vectors from the multivariate normal distribution they define, the same large-sample step the package's Monte Carlo method takes. The Bayesian shape is `draws`, a matrix of posterior coefficient draws such as `as.matrix(fit)` from brms, used as given. Under either shape, each draw's `e`, `x` and `y` at time `t` are its intercept plus `t` times its slope, and the draws at each time go through `ssm_draws()` with `type = "parameters"`.

Usage

```
ssm_trajectory(
  coef,
  vcov,
```

```
  times,
  draws = NULL,
  time = "wave",
  interval = 0.95,
  n_draws = 4000,
  contrast = NULL
)

## S3 method for class 'circumplex_ssm_trajectory'
print(x, digits = 2, ...)

## S3 method for class 'circumplex_ssm_trajectory'
rbind(..., deparse.level = 1, make.row.names = TRUE, stringsAsFactors = FALSE)
```

Arguments

coef	The fixed effects: a named numeric vector. Required with vcov, and absent with draws.
vcov	The covariance matrix of coef: a symmetric numeric matrix with one row and column per element of coef, its dimnames equal to names(coef) when present. Required with coef, and absent with draws.
times	Required. A numeric vector of the time values at which to evaluate the trajectory, one output row each.
draws	Optional. A numeric matrix of coefficient draws, one row per draw and one named column per coefficient, in place of coef and vcov. When given, n_draws is ignored.
time	Optional. The name of the time column in the output, and the suffix of the slope coefficients (default "wave"), the same time given to ssm_growth_data() and ssm_growth_formula().
interval	Optional. A single number between 0 and 1 giving the level of the intervals (default = 0.95).
n_draws	Optional. The number of coefficient vectors to draw from coef and vcov (default = 4000).
contrast	Optional. NULL (default) for the default contrast, or a function of one time value returning a 3 by p numeric matrix whose rows are e, x and y and whose columns follow the order of coef or of the columns of draws.
x	An object of class "circumplex_ssm_trajectory".
digits	The number of decimal places to print (default = 2).
...	For print(), ignored (S3 consistency). For rbind(), the trajectory tables to stack.
deparse.level, make.row.names, stringsAsFactors	For rbind(), passed to rbind.data.frame().

Details

The default contrast reads six coefficient names: dve, dvx, dvy and dve:<time>, dvx:<time>, dvy:<time>, where <time> is the time argument. Those are the names the model from ssm_growth_formula()

gives its fixed effects on the long table ssm_growth_data() builds. A b_ prefix on the column names of draws is dropped, and other columns, such as brms's sd_, cor_ and lp__ columns, are ignored. A missing name is an error. A contrast function replaces the default for a model with other terms, such as a quadratic time term; it must return, for one time value, the 3 by p matrix that maps the p coefficients to e, x and y at that time, in coefficient order.

The displacement interval at a time point is wrong when x(t) and y(t) are treated as independent, which is what fitting the coordinates in separate models does. So the coef and vcov shape refuses a vcov whose implied covariance between x(t) and y(t) is exactly zero at every time in times, unless vcov is zero everywhere. This joint-fit check reads only the given times, so with a single time it sees only the covariance terms that time reaches. A draws matrix is not checked for a joint fit.

The coef and vcov shape with the default contrast also warns when vcov does not look like a fit to complete balanced data. For each of e, x and y, the negative covariance of the intercept and slope divided by the slope variance is the time at which that coordinate's fitted value has its smallest variance. A fit of the ssm_growth_formula() model to complete balanced data gives three equal values, the mean design time. The function warns when the three differ by more than sqrt(.Machine$double.eps) times a scale in units of time: the largest, over e, x and y, of the square root of the intercept variance divided by the slope variance. A change of time units changes the three values and this scale by the same factor, so it does not change whether the function warns. The scale grows with the random-intercept variance relative to the residual variance. So when that ratio is large, an unbalanced fit can give values whose spread is below the threshold, and no warning. Causes that can make the values differ include persons with different numbers of observations, fixed effects beyond that model's six, and a random-effects structure or residual-variance structure other than that model's. Equal values do not show balance. For that model, the values stay equal when every person has the same number of observations. With different numbers of observations, they also stay equal when the three coordinates have equal random-intercept variances, equal intercept covariances and equal residual variances. The function skips this balance check when a slope variance or an intercept variance is not positive, or when extreme variances make the three values or the scale overflow, or make the scale underflow to zero. The draws shape and a contrast function are not checked for balance.

The intervals from a REML fit's coef and vcov condition on its estimated variance components and are too narrow at small samples. The package's small-sample coverage oracle fit the ssm_growth_formula() model with glmmTMB to complete balanced data at five waves. Among its converged fits (those with a positive-definite Hessian), the coverage averaged over waves and over e, x, y and d was .938 to .944 at 20 to 50 persons, against a nominal .95. At 10 persons, 27.4% and 17.7% of the fits under its two simulated truths had a non-positive-definite Hessian. The "Growth Models on SSM Parameters" vignette, Section 7, gives the design, the other figures and the records, and says that no shipped correction exists. Intervals from draws summarize those draws as given.

Value

A data frame of class "circumplex_ssm_trajectory" with attribute time naming its time column and attribute input recording the input shape, "coef_vcov" or "draws", one row per value of times. Its columns are <time>; e_est, e_lci, e_uci, and the same three for x, y, a and d; and certified. The estimates and bounds are those ssm_draws() reports: medians and equal-tailed interval bounds for e, x, y and a, and the circular mean with circular quantile bounds in degrees for d. A d interval that straddles 0/360 degrees has d_lci > d_uci. certified is the displacement certification at that time, and at an uncertified time the d interval is not interpretable. Printing shows

the table rounded, marks each uncertified row, and ends with the caution for the input shape: the small-sample caution under `coef` and `vcov`, and under `draws` that the intervals summarize the draws as given. A subset that drops the attribute, or an `rbind()` of trajectory tables of different shapes, prints a caution that says the shape is not recorded. `ssm_plot_trajectory()` plots the object with no `time` argument.

See Also

Other growth functions: `ssm_growth_data()`, `ssm_growth_formula()`

Examples

```
# Fixed effects and covariance in the shape a joint fit returns
coef <- c(
  dve = 0.52, dvx = 0.61, dvy = -0.10,
  "dve:wave" = 0.00, "dvx:wave" = 0.00, "dvy:wave" = 0.065
)
vcov <- diag(c(7e-4, 3e-4, 3e-4, 5e-5, 2e-5, 2e-5))
vcov[2, 3] <- vcov[3, 2] <- 4e-5
dimnames(vcov) <- list(names(coef), names(coef))
set.seed(1)
trajectory <- ssm_trajectory(coef, vcov, times = 0:4)
trajectory
ssm_plot_trajectory(trajectory)

# The same from a matrix of coefficient draws, here drawn by hand; a brms
# fit gives one as `as.matrix(fit)`
set.seed(1)
draws <- matrix(rnorm(500 * 6), nrow = 500) %*% chol(vcov)
draws <- sweep(draws, 2, coef, "+")
colnames(draws) <- names(coef)
ssm_trajectory(times = 0:4, draws = draws)
```

---

summary.circumplex_axes_reliability

Summarize circumplex axes-reliability results

Description

Fuller display of an `axes_reliability()` object: everything `print()` shows plus the estimated variance components (with standard errors) and the global fit indices.

Usage

```
## S3 method for class 'circumplex_axes_reliability'
summary(object, digits = 3, ...)
```

Arguments

`object`	A `circumplex_axes_reliability` object.
`digits`	The number of decimal places to display (default = 3).
`...`	Not used.

Value

`object`, invisibly.

`summary.circumplex_ci_accuracy` *Summarize the accuracy of SSM confidence intervals*

Description

Print the full report of an `ssm_ci_accuracy()` run: the assessed configuration, a structure note describing the simulated population (with cautions when the structural model converged badly or fits poorly – benchmarks per Browne & Cudeck, 1993, and Hu & Bentler, 1999), the per-profile verdict blocks (coverage of elevation, amplitude, and certification-conditional displacement classified against Bradley's liberal band; the guardrail false-certification caution), and a compact table with one row per profile and amplitude condition. That table shows the coverage of each parameter, the displacement coverage when certified, the certification rate and the `Structural` flag. On a contrast row, "certified" means that both profiles were certified. The Monte Carlo standard errors, the one-sided miss rates, the interval widths, the conditional and replication counts, and the other guardrail columns are not printed; read them from the `coverage` and `guardrail` elements of the object.

Usage

```
## S3 method for class 'circumplex_ci_accuracy'
summary(object, digits = 3, ...)
```

Arguments

`object`	A `circumplex_ci_accuracy` object from `ssm_ci_accuracy()`.
`digits`	Number of digits to which table entries are rounded (default = 3).
`...`	Currently ignored.

Value

The object, invisibly.

References

Browne, M. W., & Cudeck, R. (1993). Alternative ways of assessing model fit. In K. A. Bollen & J. S. Long (Eds.), *Testing structural equation models* (pp. 136-162). Sage.

Hu, L., & Bentler, P. M. (1999). Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural Equation Modeling, 6*(1), 1-55.

`summary.circumplex_cpm` *Summarize a circular process model fit*

Description

Fuller display of a `cpm_fit()` object: adds the correlation-function weights, the full set of fit indices, a residual summary (the largest absolute residual and the pair it belongs to), and all boundary/identification diagnostics and the lower-discrepancy note in plain language. When the confidence intervals are analytic, prints a coverage caution calibrated by simulation: unconditionally when the sample size is modest (N < 2000, where Wald intervals mis-covered for every configuration studied), and from N = 2000 up to but not including N = 50000 when the fitted solution shows a boundary or weak-identification marker (Heywood communality, boundary harmonic removed, small correlation-function weight, ill-conditioned Hessian, or competing near-tied optima), the regime where they mis-covered even at large N (see `cpm_fit()`). The vignette section *When a fit sits at a boundary* (`vignette("cpm-boundary-fits")`) glosses each marker and gives the interpretation and next steps when one fires. When the confidence intervals are bootstrap, any fired markers are instead listed in a descriptive note at every sample size; the note also states that what has been measured about the markers covers analytic intervals only (and not every marker was measured), so they are not validated as predictors of the bootstrap intervals.

Usage

```
## S3 method for class 'circumplex_cpm'
summary(object, digits = 3, ...)
```

Arguments

`object`	A `circumplex_cpm` object.
`digits`	The number of decimal places to display (default = 3).
`...`	Not used.

Value

`object`, invisibly.

`summary.circumplex_ssm_id` *Summarize per-person SSM parameters at the group level*

Description

Aggregate a per-person SSM parameter table (from `ssm_parameters_id()`) into group-level summaries, using circular statistics for displacement: arithmetic means are meaningless for angles, so displacement is summarized by its circular mean (the direction of the summed unit vectors) and the mean resultant length (a 0 to 1 measure of directional concentration).

Usage

```
## S3 method for class 'circumplex_ssm_id'
summary(object, ...)
```

Arguments

object	Required. An object of class "circumplex_ssm_id" created by ssm_parameters_id().
...	Ignored (S3 consistency).

Details

Persons with undefined (NA) displacement are stripped before the circular aggregation – n_na_d reports how many – while the arithmetic means of the other parameters use all persons with defined values. Two aggregation caveats apply. (1) The circular mean of per-person displacements weights every person's direction equally; it is a *different quantity* from the displacement of the group mean profile (e.g., from ssm_analyze()), which weights persons by amplitude – on heterogeneous samples the two can differ substantially. (2) By the triangle inequality, the amplitude of the group mean profile is at most the mean per-person amplitude (a_mean), strictly smaller when directions disperse; relatedly, the mean resultant length d_res falls below 1 as directions disperse.

Value

A one-row data frame with columns n (persons), n_na_d (persons with undefined displacement, excluded from the circular summaries), e_mean, x_mean, y_mean, a_mean (arithmetic means), d_mean (circular mean of displacement, degrees in [0, 360], the 0/360 pole reported as 360), and d_res (mean resultant length in [0, 1]; NA when no displacement is defined, and undefined direction at zero resultant reports d_mean = NA).

See Also

Other ssm functions: plot.circumplex_ci_accuracy(), ssm_analyze(), ssm_analyze_long(), ssm_ci_accuracy(), ssm_draws(), ssm_parameters(), ssm_parameters_id(), ssm_score(), ssm_sem(), ssm_sem_parameters(), ssm_table()

Other analysis functions: cpm_fit(), cpm_simulate(), ssm_analyze(), ssm_analyze_long(), ssm_ci_accuracy(), ssm_draws(), ssm_parameters(), ssm_parameters_id(), ssm_score(), ssm_sem(), ssm_sem_parameters()

Examples

```
data("aw2009")
res <- ssm_parameters_id(
  aw2009,
  scales = c("PA", "BC", "DE", "FG", "HI", "JK", "LM", "NO")
)
summary(res)
```

summary.circumplex_structure *Summarize circumplex-structure test results*

Description

Fuller display of a `fit_structure()` object: adds the interpretive cutoffs behind each classification, the estimated angle and communality of each scale, and the analysis settings.

Usage

```
## S3 method for class 'circumplex_structure'
summary(object, digits = 3, ...)
```

Arguments

`object`	A `circumplex_structure` object.
`digits`	The number of decimal places to display (default = 3).
`...`	Not used.

Value

`object`, invisibly.

theme_circumplex *Circumplex canvas theme*

Description

The **ggplot2** theme applied to the circumplex canvas built by `ggcircumplex()`. It is built on `ggplot2::theme_minimal()` so that the amplitude rings, displacement spokes, and labels drawn by `coord_circumplex()` are themed panel furniture that respond to further theming. Apply it to a custom circumplex plot, and add `+ theme_*()` or `+ theme()` on top to restyle the canvas.

Usage

```
theme_circumplex(base_size = 12)
```

Arguments

`base_size`	A single positive number giving the base font size (in pt) for the theme (default = 12).

Value

A **ggplot2** theme object, to be added to a plot with `+`.

See Also

Other circumplex layers: coord_circumplex(), geom_ssm_arc(), geom_ssm_ellipse(), geom_ssm_path(), geom_ssm_point(), ggcircumplex(), scale_x_circumplex()

Examples

```
# Restyle the canvas with a larger base font
ggcircumplex(octants()) + theme_circumplex(base_size = 16)
```

Index

∗ **analysis functions**
  cpm_fit, 16
  cpm_simulate, 21
  ssm_analyze, 52
  ssm_analyze_long, 58
  ssm_ci_accuracy, 60
  ssm_draws, 64
  ssm_parameters, 70
  ssm_parameters_id, 72
  ssm_score, 80
  ssm_sem, 82
  ssm_sem_parameters, 87
  summary.circumplex_ssm_id, 97
∗ **circumplex layers**
  coord_circumplex, 15
  geom_ssm_arc, 24
  geom_ssm_ellipse, 26
  geom_ssm_path, 27
  geom_ssm_point, 29
  ggcircumplex, 30
  scale_x_circumplex, 47
  theme_circumplex, 99
∗ **datasets**
  aw2009, 5
  jz2017, 35
  raw_iipsc, 46
  simulated_growth, 50
  simulated_items, 51
  simulated_occasions, 52
∗ **growth functions**
  ssm_growth_data, 67
  ssm_growth_formula, 68
  ssm_trajectory, 92
∗ **instrument functions**
  anchors, 3
  instruments, 33
  items, 34
  norms, 36
  scales, 46
∗ **ssm functions**
  plot.circumplex_ci_accuracy, 40
  ssm_analyze, 52
  ssm_analyze_long, 58
  ssm_ci_accuracy, 60
  ssm_draws, 64
  ssm_parameters, 70
  ssm_parameters_id, 72
  ssm_score, 80
  ssm_sem, 82
  ssm_sem_parameters, 87
  ssm_table, 91
  summary.circumplex_ssm_id, 97
∗ **structure functions**
  fit_structure, 22
∗ **table functions**
  html_render, 32
  ssm_table, 91
∗ **tidying functions**
  ipsatize, 33
  norm_standardize, 37
  score, 48
  self_standardize, 49
∗ **visualization functions**
  plot.circumplex_ci_accuracy, 40
  ssm_plot_circle, 73
  ssm_plot_contrast, 75
  ssm_plot_curve, 76
  ssm_plot_trajectory, 78

anchors, 3
anchors(), *33*, *35*, *37*, *46*
angle_unwrap, 4
aw2009, 5
axes_reliability, 6
axes_reliability(), *44*, *51*, *95*

boot, *54*

coord_circumplex, 15

coord_circumplex(), *24–27*, *29–32*, *42*, *47*, *74*, *99*, *100*
cpm_fit, 16
cpm_fit(), *21*, *22*, *24*, *41*, *42*, *44*, *56*, *59–63*, *65*, *71*, *73*, *81*, *85*, *86*, *88*, *97*, *98*
cpm_simulate, 21
cpm_simulate(), *20*, *56*, *59*, *63*, *65*, *71*, *73*, *81*, *86*, *88*, *98*

fit_structure, 22
fit_structure(), *6*, *14*, *42*, *43*, *45*, *99*

geom_ssm_arc, 24
geom_ssm_arc(), *15*, *16*, *27*, *29*, *30*, *32*, *47*, *100*
geom_ssm_ellipse, 26
geom_ssm_ellipse(), *16*, *25*, *29*, *30*, *32*, *47*, *66*, *100*
geom_ssm_path, 27
geom_ssm_path(), *16*, *25*, *27*, *30*, *32*, *47*, *74*, *80*, *100*
geom_ssm_point, 29
geom_ssm_point(), *15*, *16*, *25*, *27*, *29*, *32*, *47*, *66*, *100*
ggcircumplex, 30
ggcircumplex(), *16*, *24–27*, *29*, *30*, *41–43*, *47*, *99*, *100*
ggplot2::arrow(), *28*
ggplot2::coord_radial(), *16*
ggplot2::geom_path(), *28*
ggplot2::scale_x_continuous(), *47*
ggplot2::theme_minimal(), *99*

html_render, 32
html_render(), *91*

instruments, 33
instruments(), *4*, *35*, *37*, *46*
ipsatize, 33
ipsatize(), *23*, *24*, *38*, *49*, *50*
items, 34
items(), *4*, *33*, *37*, *46*

jz2017, 35

lavaan::bootstrapLavaan(), *84*, *85*
lavaan::cfa(), *85*, *90*

norm_standardize, 37
norm_standardize(), *34*, *49*, *50*
norms, 36
norms(), *4*, *33*, *35*, *38*, *46*

octants, 39
octants(), *6*, *17*, *50–52*

PANO, 39
plot(), *23*
plot.circumplex_ci_accuracy, 40
plot.circumplex_ci_accuracy(), *56*, *59*, *63*, *65*, *71*, *73*, *75–77*, *80*, *81*, *86*, *88*, *91*, *98*
plot.circumplex_cpm, 41
plot.circumplex_structure, 42
poles, 43
print(), *23*, *95*
print.circumplex_axes_reliability, 44
print.circumplex_cpm, 44
print.circumplex_cpm(), *18*
print.circumplex_growth_formula (ssm_growth_formula), 68
print.circumplex_ssm_trajectory (ssm_trajectory), 92
print.circumplex_structure, 45

quadrants, 45

raw_iipsc, 46
rbind.circumplex_ssm_trajectory (ssm_trajectory), 92
rbind.data.frame(), *93*

scale_x_circumplex, 47
scale_x_circumplex(), *16*, *25*, *27*, *29*, *30*, *32*, *100*
scales, 46
scales(), *4*, *33*, *35*, *37*
score, 48
score(), *6*, *34*, *38*, *50*
self_standardize, 49
self_standardize(), *34*, *38*, *49*
set.seed(), *23*
simulated_growth, 50
simulated_growth_origin (simulated_growth), 50
simulated_items, 51
simulated_occasions, 52
ssm_analyze, 52, *71*

ssm_analyze(), *19, 20*, *22, 23*, *40*, *58–61*, *63*, *65, 66*, *71*, *73, 74*, *78, 79*, *81, 82*, *85, 86*, *88*, *91*, *98*
ssm_analyze_long, 58
ssm_analyze_long(), *20*, *22*, *40*, *52*, *56*, *63*, *65*, *71*, *73, 74*, *78, 79*, *81*, *86*, *88*, *91*, *98*
ssm_ci_accuracy, 60
ssm_ci_accuracy(), *20*, *22*, *40*, *56*, *59*, *65*, *71*, *73*, *81*, *86*, *88*, *91*, *96*, *98*
ssm_draws, 64
ssm_draws(), *20*, *22*, *27*, *40*, *56*, *59*, *63*, *66*, *71*, *73*, *79*, *81*, *86*, *88*, *91, 92*, *94*, *98*
ssm_ellipse_data, 66
ssm_ellipse_data(), *27*
ssm_growth_data, 67
ssm_growth_data(), *50*, *68, 69*, *93–95*
ssm_growth_formula, 68
ssm_growth_formula(), *50*, *67, 68*, *92–95*
ssm_parameters, 70, *81*
ssm_parameters(), *20*, *22*, *40*, *56*, *59*, *63*, *65*, *72, 73*, *81*, *86*, *88*, *91*, *98*
ssm_parameters_id, 72
ssm_parameters_id(), *20*, *22*, *40*, *56*, *59*, *63*, *65*, *68*, *71*, *81*, *86*, *88*, *91*, *97, 98*
ssm_plot_circle, 73
ssm_plot_circle(), *15*, *32*, *40*, *76, 77*, *80*
ssm_plot_contrast, 75
ssm_plot_contrast(), *40*, *75*, *77*, *79, 80*
ssm_plot_curve, 76
ssm_plot_curve(), *40*, *75, 76*, *80*
ssm_plot_trajectory, 78
ssm_plot_trajectory(), *40*, *52*, *75–77*, *95*
ssm_score, *70*, 80
ssm_score(), *20*, *22*, *40*, *56*, *59*, *63*, *65*, *71–73*, *86*, *88*, *91*, *98*
ssm_sem, 82
ssm_sem(), *20*, *22*, *40*, *56*, *59*, *63*, *65*, *71*, *73*, *81*, *87, 88*, *91*, *98*
ssm_sem_parameters, 87
ssm_sem_parameters(), *20*, *22*, *40*, *56*, *59*, *63*, *65*, *71*, *73*, *81*, *86*, *91*, *98*
ssm_sem_syntax, 89
ssm_sem_syntax(), *82, 83*, *86, 87*
ssm_table, 91
ssm_table(), *33*, *40*, *56*, *59*, *63*, *65*, *71*, *73*, *81*, *86*, *88*, *98*
ssm_trajectory, 92
ssm_trajectory(), *50*, *68, 69*, *79*
summary(), *10*, *23*
summary.circumplex_axes_reliability, 95
summary.circumplex_ci_accuracy, 96
summary.circumplex_ci_accuracy(), *62*
summary.circumplex_cpm, 97
summary.circumplex_cpm(), *18*
summary.circumplex_ssm_id, 97
summary.circumplex_ssm_id(), *20*, *22*, *40*, *56*, *59*, *63*, *65*, *71*, *73*, *81*, *86*, *88*, *91*
summary.circumplex_structure, 99

theme_circumplex, 99
theme_circumplex(), *16*, *25*, *27*, *29, 30*, *32*, *47*