Package 'janusplot'

November 14, 2026

Type Package

Title Asymmetric Smoothed-Association Matrices via GAM Fits

Version 1.0.0

Description Render a pairwise, asymmetric smoothed-association matrix of continuous variables. Each cell shows the fitted spline from an 'mgcv' generalised additive model, with the upper triangle displaying 'gam(x_i ~ s(x_j))' and the lower triangle 'gam(x_j ~ s(x_i))'. Unlike Pearson's correlation matrix, the visualisation is intentionally asymmetric, revealing heteroscedasticity, leverage, and directional non-linearity that a single scalar correlation hides. An asymmetry index and a 27-category shape taxonomy quantify the directional difference and qualitative form of each fitted smooth.

License MIT + file LICENSE

URL https://github.com/max578/janusplot, https://max578.github.io/janusplot/

BugReports https://github.com/max578/janusplot/issues

Encoding UTF-8

Language en-AU

Depends R (>= 4.3.0)

Imports mgcv (>= 1.9.0), ggplot2 (>= 3.5.0), patchwork (>= 1.1.0), grid, grDevices, stats, tools, cli (>= 3.6.0), lifecycle, rlang (>= 1.1.0)

Suggests agridat, future, future.apply, kernR (>= 0.8.0), knitr, MASS, palmerpenguins, rmarkdown, testthat (>= 3.0.0), vdiffr (>= 1.0.0), withr

Additional_repositories https://max578.r-universe.dev

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website pkgdown

LazyData true

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Max Moldovan [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0001-9680-8474>)

Maintainer Max Moldovan <max.moldovan@adelaide.edu.au>

Repository CRAN

Date/Publication 2026-11-14 04:00:02 UTC

Contents

janusplot 2
janusplot_compare 5
janusplot_curves 7
janusplot_direction 8
janusplot_fit 12
janusplot_pair 17
janusplot_reference 19
janusplot_shape_cutoffs 21
janusplot_shape_hierarchy 22
janusplot_shape_metrics 23
janusplot_shape_sensitivity 25
janusplot_style 28
plot.janusplot_compare 32
plot.janusplot_fit 34
shape_sensitivity_demo 37

Index **38**

`janusplot` *Asymmetric smoothed-association matrix*

Description

[Stable]

Fits every ordered pair of numeric variables with `janusplot_fit()` and draws the matrix with `plot.janusplot_fit()` in one call. Each cell `[r, c]` off the diagonal shows the fitted spline of `gam(x_r ~ s(x_c) + <adjust>)` from `mgcv::gam()` or `mgcv::bam()`: the row variable on the vertical axis against the column variable on the horizontal axis, as in `graphics::pairs()`. The diagonal is blank; the variable names sit on the border and a key in the top-left corner shows the convention.

The two triangles differ by design: the asymmetry reveals heteroscedasticity, leverage and directional non-linearity that a single scalar correlation hides.

Usage

```
janusplot(
  data,
  vars = NULL,
  adjust = NULL,
  colour_by = "nonlinearity",
  background_by = NULL,
  annotations = NULL,
  order = "original",
  display = "fit",
  group = NULL,
  focus_by = NA_character_,
  focus_threshold = "q90",
  axes = "original",
  style = janusplot_style(),
  ...
)
```

Arguments

`data`	A data frame. Column names need not be syntactic; fits run under internal aliases and every result reports the original names.
`vars`	Character vector of numeric column names to fit, in matrix order. `NULL` (the default) takes every numeric column of `data`. Identical or exactly collinear columns are not fitted against each other; their cells are listed in `flags$failed`.
`adjust`	One-sided formula of covariates and random effects added to every pairwise model, for example `~ s(age) + s(site, bs = "re")`. `NULL` (the default) fits unadjusted smooths. Character and logical `adjust` columns are converted to factor before fitting. Random-effect smooths are excluded from prediction, so the curve and its band are population-level. With the identity link (the Gaussian default) the curve is the smooth averaged over the adjustment covariates of the data, and the plots draw each point as a partial residual: the observation with its own adjustment replaced by that average, so points and curve share one level (`style$adjusted_points`). With another link, numeric covariates are held at their mean and fixed factors at their most frequent level, and points stay raw.
`colour_by`	The quantity painted on each cell: `"nonlinearity"` (the default; EDF minus 1 of the cell's smooth, sequential from 0), `"asymmetry"` (the pair's index A in `[0, 1]`, the same in both cells), `"pearson"`, `"spearman"` or `"kendall"` (the pair's correlation on a diverging scale over `[-1, 1]`), `"deviance_gap"` (deviance explained by the smooth minus the straight line's) or `"none"`.
`background_by`	The correlation that tints each cell's background: `"pearson"`, `"spearman"`, `"kendall"`, or `"none"` for no background. Left unset, it is `"pearson"` for a marginal fit and `"none"` for a fit with `adjust`, whose curves are adjusted while the correlations are marginal; set it to show the marginal correlation behind adjusted curves. Independent of `colour_by`. It cannot be combined with `janusplot_style(colour_channel = "fill")`, which paints the same area.

annotations — Character vector, a subset of `c("A", "edf", "signif", "code", "shape", "k_warn", "n", "r", "formula")`, written in the strip of every cell (or in its corners under `janusplot_style(annotation_position = "corners")`): the asymmetry index, the EDF, the significance glyph of the smooth F-test (*** p < .001, ** p < .01, * p < .05, . p < .1, from the adjusted p-value when the fit has `p_adjust` other than `"none"`), the two-letter shape code, the shape glyph, a warning sign on cells flagged by the k-check or capped at the distinct predictor count, the rows used, the `background_by` correlation of the pair (Pearson when `background_by = "none"`), printed `"r -0.03"` (`"marginal r -0.03"` for a fit with `adjust`), and the cell's formula, printed `Temp ~ Ozone`. `NULL` (the default) picks by matrix size: `c("A", "signif")` up to twelve variables, `"signif"` from thirteen, where 5-point strips of neighbouring cells would touch; `character()` writes none.

order — Order of the variables in the matrix: `"original"` (the default, the order of `x$vars`), `"hclust"` (hierarchical clustering of absolute Pearson correlations) or `"alphabetical"`. Drawing only; the fit is unchanged.

display — What every off-diagonal cell shows: `"fit"` (the default), `"d1"` (the first derivative of the smooth, whose zero crossings are turning points) or `"d2"` (the second derivative, whose zero crossings are inflections). A derivative view needs the fit to carry that order in `x$settings$derivatives`.

group — Name of a factor, character or logical column of `x$data` whose levels colour the raw points, with one legend below the matrix, or `NULL` (the default). Drawing only; the column never enters the model. The column must exist in `x$data` and have at most `style$group_max_levels` levels.

focus_by — `NA` (the default, no filter) or one of `"asymmetry"`, `"edf"`, `"non_linearity"` (EDF minus 1) and `"k_flag"`. Cells whose metric falls below `focus_threshold` are washed in grey at `style$focus_dim_alpha`; the matrix keeps its shape. A visual filter only; the fit and its table are unchanged.

focus_threshold — A quantile string, `"q90"` by default (taken over the off-diagonal cells), or a single numeric cutoff. Ignored when `focus_by` is `NA`.

axes — Scale of the drawn axes: `"original"` (the default), `"standardised"` (`(x - mean) / sd` per variable, border label `"var (z)"`), `"centred"` (`x - mean`, label `"var (centred)"`) or `"rank"` (empirical rank, label `"rank(var)"`). Drawing only; the same transform is applied to the points, the curve and the band.

style — A `janusplot_style()` object holding every appearance setting.

... — Model arguments of `janusplot_fit()`, further arguments to `mgcv::gam()` or `mgcv::bam()`, and for one release the 0.2.0 appearance arguments listed in Details.

Details

Model settings (`k`, `bs`, `engine`, `method`, `na_action`, `n_grid`, `derivatives`, `derivative_ci`, `derivative_ci_nsim`, `p_adjust`, `shape_cutoffs`, `k_check_thresholds`, `auto_refit_k`, `k_max_iter`, `discrete`, `nthreads`, `parallel`, `keep_fits`) pass through `...` to `janusplot_fit()`, which forwards anything else to mgcv. Appearance settings live in `janusplot_style()` and are passed as `style`. To draw the same fit more than once, call `janusplot_fit()` and `plot()` it.

The 0.2.0 appearance arguments of `janusplot()` are accepted through `...` for one release, each with a deprecation warning, and are removed in 1.1.0: `show_data`, `show_ci`, `palette`, `show_shape_legend`, `glyph_style`, `labels`, `diagonal`, `label_srt`, `show_glossary`, `compact`, `compact_threshold`, `compact_levels` and `focus_dim_alpha` are applied to `style`; `label_cex`, `text_scale_diag`, `text_scale_off_diag` and `glossary_scale` are mapped to the `labels`, `diagonal`, `strip` and `glossary` elements of `janusplot_style(text_size = )` at the matrix's variable count, close to but not exactly the 0.2.0 sizes; `fill_by` is mapped to `colour_by`; `show_asymmetry` and `signif_glyph` are mapped to `annotations`; `labels = "diagonal"` is drawn as `"border"` with a message; `with_data = TRUE` returns `list(plot, data)` with `data` the table of `summary(janusplot_fit(...))`; `save_as`, `save_width`, `save_height` and `save_dpi` write the plot once with `ggplot2::ggsave()`. Arguments after `adjust` must be named: the positional order of 0.2.0 is not shimmed.

Value

A patchwork object, which is a ggplot, as described in `plot.janusplot_fit()`. With the deprecated `with_data = TRUE`, a list with elements `plot` (that object) and `data` (the per-cell table of `summary.janusplot_fit()`).

See Also

`janusplot_fit()` for the object behind the plot, `plot.janusplot_fit()` for the view arguments, `janusplot_style()` for the appearance settings.

Other smooth-associations: `janusplot_compare()`, `janusplot_curves()`, `janusplot_direction()`, `janusplot_fit()`, `janusplot_pair()`, `janusplot_reference()`, `janusplot_style()`, `plot.janusplot_compare()`, `plot.janusplot_fit()`

Examples

```
# Three variables, six asymmetric pairwise GAM fits.
janusplot(mtcars[, c("mpg", "hp", "wt")], engine = "gam")

# Other colourings, groups and the derivative displays
# (display = "d1", "d2") are shown in vignette("janusplot").
```

---

janusplot_compare *Compare two fits of the same variables cell by cell*

Description

[Stable]

Matches every directed cell of two `janusplot_fit()` objects over the same variables and reports how the EDF, the asymmetry index and the shape labels move from the first fit to the second. The typical use puts a marginal fit against one adjusted for a grouping variable, which shows how sensitive each fitted shape is to that adjustment. Nothing is refitted.

Usage

```
janusplot_compare(fit_a, fit_b, labels = c("A", "B"))

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

Arguments

fit_a, fit_b	janusplot_fit objects over the same set of variables. fit_a is the reference (often the marginal fit), fit_b the comparison (often the adjusted fit).
labels	Character vector of length 2 naming fit_a and fit_b in printed and plotted output; two different, non-empty strings. c("A", "B") by default.
x	A janusplot_compare object.
...	Ignored by the print() method.

Details

Cells are matched by variable names, so the two fits may store their variables and pairs in different orders. Differences are the second fit minus the first. A shape or archetype change is NA when the fit failed on either side, where the label is "indeterminate" or NA.

The comparison reports sensitivity to adjustment. A changed label says the fitted shape depends on the adjustment; it does not by itself say why.

Value

A list of class janusplot_compare with components:

table Data frame with one row per directed cell, in the row order of summary(fit_a): var_x (predictor), var_y (response), position ("upper" or "lower" in fit_a), n_used_a, n_used_b, edf_a, edf_b, delta_edf, A_a, A_b, delta_A (the pair's asymmetry index), shape_a, shape_b, shape_changed (logical), archetype_a, archetype_b, archetype_changed (logical), trend_a, trend_b and trend_changed (logical). The trend is "increasing" or "decreasing" for a monotone shape (by the sign of the monotonicity index), otherwise "non_monotone" or "degenerate".

fit_a, fit_b The two fits as given.

labels The two labels.

print() returns x invisibly.

See Also

plot.janusplot_compare() for the matrix with both curves, janusplot_fit() for the fits.

Other smooth-associations: janusplot(), janusplot_curves(), janusplot_direction(), janusplot_fit(), janusplot_pair(), janusplot_reference(), janusplot_style(), plot.janusplot_compare(), plot.janusplot_fit()

Examples

```
set.seed(2026L)
g <- rep(c("g1", "g2", "g3"), each = 40L)
mu <- c(g1 = 0, g2 = 2, g3 = 4)[g]
d <- data.frame(x = mu + stats::rnorm(120L), g = factor(g))
d$y <- -1.5 * mu + 0.8 * (d$x - mu) + stats::rnorm(120L, sd = 0.4)
d$z <- sin(d$x) + stats::rnorm(120L, sd = 0.3)
vars <- c("x", "y", "z")
fit_a <- janusplot_fit(d, vars, engine = "gam", derivatives = integer())
fit_b <- janusplot_fit(d, vars, adjust = ~ g, engine = "gam",
                       derivatives = integer())
cmp <- janusplot_compare(fit_a, fit_b,
                         labels = c("marginal", "group-adjusted"))
cmp
cmp$table[, c("var_x", "var_y", "edf_a", "edf_b", "shape_changed")]
```

---

janusplot_curves *Fitted curves of a janusplot fit as a data frame*

Description

[Experimental]

Returns the curves a janusplot_fit() object stores, one row per grid point of every directed cell, for drawing with ggplot2 or any other tool. The fitted curves come with their 95 percent band; the derivative curves, when the fit computed them, with theirs. Nothing is refitted.

Usage

```
janusplot_curves(fit, x = NULL, y = NULL, what = c("fit", "d1", "d2"))
```

Arguments

fit	A janusplot_fit object from janusplot_fit().
x, y	Two variable names of fit$vars to return both directions of that pair only, or NULL (the default) for every pair.
what	"fit" (the default) for the fitted curves, "d1" or "d2" for the first or second derivative, which need that order in fit$settings$derivatives.

Details

A cell var_y ~ s(var_x) is identified by var_x and var_y, and by position as in summary(): "upper" when var_y comes first in fit$vars. The fitted curve is on the response scale and its band is mapped from the link scale, as in the plots. Under adjust with the identity link, the curve is averaged over the adjustment covariates (see janusplot_fit()). Cells that failed to fit have no rows.

Value

A data frame with columns `var_x`, `var_y`, `position`, `x`, `fit`, `se` (on the link scale), `lo` and `hi`, one row per grid point of each directed cell.

See Also

`summary.janusplot_fit()` for the per-cell numbers.

Other smooth-associations: `janusplot()`, `janusplot_compare()`, `janusplot_direction()`, `janusplot_fit()`, `janusplot_pair()`, `janusplot_reference()`, `janusplot_style()`, `plot.janusplot_compare()`, `plot.janusplot_fit()`

Examples

```
fit <- janusplot_fit(airquality[, c("Ozone", "Temp", "Wind")],
                     engine = "gam")
curves <- janusplot_curves(fit, "Temp", "Ozone")
head(curves)

# Both directions of the pair on shared axes.
library(ggplot2)
ggplot(curves, aes(x, fit, colour = var_y)) +
  geom_ribbon(aes(ymin = lo, ymax = hi, fill = var_y), alpha = 0.2,
              colour = NA) +
  geom_line() +
  facet_wrap(~ var_y, scales = "free")
```

---

janusplot_direction *Test whether the residual-independence condition licenses a direction*

Description

[Experimental]

`janusplot_pair()` draws a pair twice – `y ~ s(x)` on the left and `x ~ s(y)` on the right – and the fit summarises the difference between the two directions as an asymmetry index. That index is a *diagnostic*: it says the two directions of fit are not interchangeable, not that either one is the causal direction.

This function runs the test that turns the picture into a claim. Under the additive-noise model `y = f(x) + e` with `e` independent of `x`, the model holds in at most one direction for all but a small set of exceptional joint distributions. So a direction is supported when

- the residuals of the fit in that direction are independent of the putative cause, **and**
- the residuals of the fit in the opposite direction are not.

Independence is assessed with the kernel (HSIC) permutation test in `kernR::hsic_test()`. Anything other than the clean one-sided pattern – both directions admitting an additive-noise model, neither doing so, or the known non-identifiable linear-Gaussian case – returns `"undecided"`. The smaller of the two p-values is never read as a direction.

Usage

```
janusplot_direction(
  fit,
  x,
  y,
  alpha = 0.05,
  n_permutations = 500L,
  linear_tol = 0.01,
  gaussian_alpha = 0.01,
  seed = NULL,
  n_min = 30L,
  n_max = 2000L
)

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

Arguments

fit	A janusplot_fit object from janusplot_fit().
x	Character. Name of the first variable, one of fit$vars – the putative cause under the forward hypothesis. For print(), a janusplot_direction object.
y	Character. Name of the second variable, one of fit$vars – the putative effect under the forward hypothesis. Names need not be syntactic: names with spaces or symbols are returned and reported as supplied.
alpha	Numeric in (0, 1). Significance level for the two residual-independence tests. Residuals are called independent when the HSIC p-value exceeds alpha. Default 0.05.
n_permutations	Integer. Permutations used to build each HSIC null distribution. Must be at least 99 – the smallest attainable p-value is 1 / (n_permutations + 1). Default 500L.
linear_tol	Numeric in (0, 1). A fitted smooth counts as a straight line when 1 - cor(fitted, x)^2 falls below this value. Default 0.01.
gaussian_alpha	Numeric in (0, 1). Significance level for the four Shapiro-Wilk normality tests used by the linear-Gaussian detector. Default 0.01.
seed	Integer or NULL. Seed for the HSIC permutation draws. The reverse direction is seeded at seed + 1 so the two null distributions are not drawn from the same permutation sequence. Also seeds the row subsample taken when the complete cases exceed n_max; with seed = NULL that subsample uses seed 2026, so it is reproducible in both cases. The subsample does not change the global random-number state.
n_min	Integer. Smallest number of complete cases on which a verdict is given. Below it the function returns "undecided" with reason_code = "n_below_min" and reason = "n below n_min", without fitting or testing. Default 30L. Samples below 20 complete cases are refused with an error whatever n_min is.

n_max	Integer. Largest number of rows used for the fits and the HSIC tests, which build n by n kernel matrices. Above it a random subsample of n_max rows is drawn (see seed). Default 2000L. The HSIC verdict depends on n: with more rows the test detects smaller departures from independence, so verdicts at different n are not directly comparable.
...	Ignored.

Details

What the fit supplies:

The rows are those the pair was fitted on in fit: complete on x, y and the adjust variables, so n equals the pair's n_used. Each direction is fitted with the basis dimension that direction used in fit (k_used_yx or k_used_xy of the pair) and the basis type fit$settings$bs, so the verdict rests on the smooths the matrix draws.

Both directions are fitted with mgcv::gam(). A fit made with engine = "gam" passes its method; a fit made with engine = "bam" carries a mgcv::bam() method ("fREML" by default), so the test uses the mgcv::gam() default "REML" instead. The method used is returned as method.

The test is bivariate. When fit was made with an adjust formula, the adjust terms do not enter the two direction fits and a message says so; only the rows they select carry over.

When fit keeps its models (keep_fits = TRUE), was made with engine = "gam" and no adjust formula, and n does not exceed n_max, the residuals, EDFs and fitted values of the two kept models are used without refitting. The result is the same as a refit.

The linear-Gaussian exception:

Identifiability fails exactly when the relationship is linear and both the cause and the noise are Gaussian: a bivariate Gaussian is reproduced perfectly by a linear model with Gaussian noise in either direction, so the residual-independence condition holds both ways and carries no directional information. This case is detected before the two independence tests are read, and returns "undecided" with reason_code = "linear_gaussian". It is flagged when *both* of the following hold:

- **Linearity.** Each fitted smooth departs from a straight line by less than linear_tol, measured as 1 - cor(fitted, x)^2 – the share of the fitted variation that no straight line in the predictor can reproduce.
- **Gaussianity.** Shapiro-Wilk applied to x, y, and both residual vectors fails to reject normality at gaussian_alpha in all four cases.

gaussian_alpha defaults to 0.01, not 0.05: the consequence of the flag is abstention, so the threshold is set to withhold a directional claim unless the data argue clearly against Gaussianity.

Reading the result:

A "forward" or "reverse" verdict states that the condition licensing that direction holds on this sample. It is not a proof of causation: the additive-noise model assumes no unobserved common cause, no feedback, and additive noise, and none of those assumptions is testable from the pair alone.

Value

A list of class janusplot_direction, printed as the verdict, its reason, the two p-values and the rows used:

`verdict` One of `"forward"` (`x` drives `y`), `"reverse"` (`y` drives `x`), or `"undecided"`.

`reason_code` Machine-readable ground for the verdict: `"forward_only"`, `"reverse_only"`, `"both_independent"`, `"neither_independent"`, `"linear_gaussian"`, or `"n_below_min"`. For `"n_below_min"` the p-values, statistics, EDFs, curvatures and normality p-values are `NA`.

`reason` One-paragraph statement of the same ground.

`x`, `y` The variable names, as supplied.

`forward`, `reverse` The two hypotheses as printable strings, e.g. `"x -> y"`.

`p_forward`, `p_reverse` HSIC permutation p-values for residual-versus-cause independence in each direction.

`statistic_forward`, `statistic_reverse` The two observed HSIC statistics.

`independent_forward`, `independent_reverse` Whether each p-value exceeds `alpha`.

`edf_forward`, `edf_reverse` Effective degrees of freedom of each fitted smooth.

`curvature_forward`, `curvature_reverse` Departure of each smooth from a straight line; see Details.

`normality_p` Named numeric vector of the four Shapiro-Wilk p-values (`x`, `y`, `residuals_forward`, `residuals_reverse`).

`linear_gaussian` `TRUE` when the non-identifiable linear-Gaussian case was detected.

`n` Complete cases on `x` and `y`.

`n_used` Rows used for the fits and tests: `n`, or `n_max` when `n` exceeds it.

`alpha`, `gaussian_alpha`, `linear_tol`, `n_permutations`, `n_min`, `n_max` The settings the verdict was reached under.

`method` The `mgcv::gam()` smoothing-parameter method of the two direction fits.

`print()` returns its argument invisibly.

References

Hoyer, P. O., Janzing, D., Mooij, J. M., Peters, J., & Scholkopf, B. (2009). Nonlinear causal discovery with additive noise models. *Advances in Neural Information Processing Systems*, **21**, 689-696.

Gretton, A., Fukumizu, K., Teo, C. H., Song, L., Scholkopf, B., & Smola, A. J. (2008). A kernel statistical test of independence. *Advances in Neural Information Processing Systems*, **20**, 585-592.

Shimizu, S., Hoyer, P. O., Hyvarinen, A., & Kerminen, A. (2006). A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, **7**, 2003-2030.

See Also

`janusplot_fit()` for the fit, `janusplot_pair()` for the two directions drawn side by side.

Other smooth-associations: `janusplot()`, `janusplot_compare()`, `janusplot_curves()`, `janusplot_fit()`, `janusplot_pair()`, `janusplot_reference()`, `janusplot_style()`, `plot.janusplot_compare()`, `plot.janusplot_fit()`

Examples

```
if (requireNamespace("kernR", quietly = TRUE)) {
  # x drives y through a non-linear function with non-Gaussian noise:
  # the additive-noise model is identifiable here.
  set.seed(2026L)
  n <- 200L
  x <- stats::runif(n, -3, 3)
  y <- x^3 / 5 + stats::runif(n, -1, 1)
  fit <- janusplot_fit(data.frame(x = x, y = y), engine = "gam")

  res <- janusplot_direction(fit, "x", "y", n_permutations = 199L,
                             seed = 2026L)
  res
  c(forward = res$p_forward, reverse = res$p_reverse)

  # The same pair read the other way round names the other direction.
  janusplot_direction(fit, "y", "x", n_permutations = 199L,
                      seed = 2026L)$verdict
}
```

`janusplot_fit` *Fit every ordered pair of a smoothed-association matrix*

Description

[Stable]

Fits `gam(x_j ~ s(x_i) + <adjust>)` once for every ordered pair of numeric variables and returns a `janusplot_fit` object holding the per-pair results, a per-cell summary table, a cleaned copy of the data used and the fit settings. `janusplot()` draws the matrix; `summary()` returns the per-cell table.

Usage

```
janusplot_fit(
  data,
  vars = NULL,
  adjust = NULL,
  k = -1L,
  bs = "tp",
  engine = c("bam", "gam"),
  method = NULL,
  na_action = c("pairwise", "complete"),
  n_grid = NULL,
  derivatives = 1:2,
  derivative_ci = c("pointwise", "none", "simultaneous"),
  derivative_ci_nsim = 1000L,
  p_adjust = c("none", "BH", "holm"),
  shape_cutoffs = janusplot_shape_cutoffs(),
```

```
  k_check_thresholds = NULL,
  auto_refit_k = FALSE,
  k_max_iter = 2L,
  discrete = FALSE,
  nthreads = 1L,
  parallel = FALSE,
  keep_fits = FALSE,
  ...
)

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

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

Arguments

data	A data frame. Column names need not be syntactic; fits run under internal aliases and every result reports the original names.
vars	Character vector of numeric column names to fit, in matrix order. NULL (the default) takes every numeric column of data. Identical or exactly collinear columns are not fitted against each other; their cells are listed in flags$failed.
adjust	One-sided formula of covariates and random effects added to every pairwise model, for example ~ s(age) + s(site, bs = "re"). NULL (the default) fits unadjusted smooths. Character and logical adjust columns are converted to factor before fitting. Random-effect smooths are excluded from prediction, so the curve and its band are population-level. With the identity link (the Gaussian default) the curve is the smooth averaged over the adjustment covariates of the data, and the plots draw each point as a partial residual: the observation with its own adjustment replaced by that average, so points and curve share one level (style$adjusted_points). With another link, numeric covariates are held at their mean and fixed factors at their most frequent level, and points stay raw.
k	Basis dimension for s(): a single integer, or a named list of integers keyed by variable name. -1L (the default) leaves the choice to mgcv. Each cell's basis is capped at the number of distinct predictor values in its rows, and an auto_refit_k refit stops one below that number; capped cells are listed in flags$k_capped and recorded as k_requested, k_used and k_capped.
bs	Basis type for s(); "tp" (thin plate) by default.
engine	"bam" (the default) fits with mgcv::bam(), "gam" with mgcv::gam(). The two differ in their default method and by about one to three percent in EDF on the same data.
method	Smoothing-parameter estimation method passed to mgcv. NULL (the default) resolves to "fREML" for engine = "bam" and "REML" for engine = "gam".
na_action	"pairwise" (the default) fits each cell on the rows complete for its own variables and the adjust columns; "complete" first drops every row with a missing value in any fitted or adjusting column. Infinite values count as missing and are

reported in a warning. A pair sharing fewer than 5 complete rows is not fitted and is listed in `flags$sparse`; an error is raised only when no pair reaches 5 rows.

`n_grid` Number of equally spaced points at which each smooth and its derivatives are evaluated, at least 10. `NULL` (the default) resolves to `200L` when `derivatives` is non-empty and `100L` otherwise.

`derivatives` Integer vector of derivative orders to compute for every cell, a subset of `1:2`. The default `1:2` computes both; `integer()` computes none.

`derivative_ci` The band stored with each derivative curve: `"pointwise"` (the default), `"none"` or `"simultaneous"`. Simultaneous bands use the Monte Carlo critical multiplier of Ruppert, Wand and Carroll (2003) as applied to GAMs by Simpson (2018).

`derivative_ci_nsim` Number of Monte Carlo draws for `derivative_ci = "simultaneous"`; `1000L` by default and at least 100.

`p_adjust` Multiplicity adjustment of the smooth F-test p-values across the `k(k - 1)` directed cells: `"none"` (the default), `"BH"` or `"holm"`, applied once with `stats::p.adjust()`. Raw and adjusted values are both kept.

`shape_cutoffs` Named list of classification thresholds, from `janusplot_shape_cutoffs()`, mapping the shape indices to a `shape_category` label.

`k_check_thresholds` Named list with entries `edf_ratio`, `k_index` and `p` for the basis-dimension diagnostic that follows `mgcv::k.check()`. `NULL` (the default) uses `edf_ratio = 0.9`, `k_index = 1` and `p = 0.05`.

`auto_refit_k` Logical. When `TRUE`, a cell flagged by the k-check is refitted with a doubled basis until the flag clears, the distinct-value cap is reached or `k_max_iter` doublings have run. The diagnostic is computed either way; the default is `FALSE`.

`k_max_iter` Maximum number of doublings per cell when `auto_refit_k = TRUE`; `2L` by default.

`discrete` Logical. Passes mgcv's covariate discretisation to `mgcv::bam()`; `FALSE` by default and ignored for `engine = "gam"`.

`nthreads` Integer. Threads used inside each `mgcv::bam()` fit; `1L` by default, which avoids oversubscription under `parallel = TRUE`. Ignored for `engine = "gam"`.

`parallel` Logical. When `TRUE`, pairs are fitted with `future.apply::future_mapply()` under the user's `future::plan()`; `FALSE` by default.

`keep_fits` Logical. When `TRUE`, the mgcv model objects are kept in `$fits` and as `fit_yx` / `fit_xy` of every pair; `FALSE` (the default) keeps metrics and prediction grids only.

`...` Further arguments passed to `mgcv::gam()` or `mgcv::bam()`. The `print()` and `summary()` methods ignore them. Under a `family` with a non-identity link the curve and its band are drawn on the response scale, while the derivatives, the shape indices and the shape label describe the linear predictor (the link scale). The two agree on monotone labels; a curvature label can differ, for example a log-linear Poisson mean is `linear_up` on the link scale and convex on the response scale. A `family` applies to every cell of the matrix, so fit count or binary columns in their own call, not beside continuous ones.

x	A janusplot_fit object.
object	A janusplot_fit object.

Details

Suffixes name the direction of fit. _xy is vars[i] ~ s(vars[j]) for i < j, the upper-triangle cell at row i, column j; _yx is vars[j] ~ s(vars[i]), the lower-triangle cell. In $table each row is one directed cell with var_x the predictor and var_y the response.

First and second derivatives of every smooth are computed at fit time by default, so n_grid = NULL resolves to 200 evaluation points. The shape indices M and C, the turning-point and inflection counts and the flat-range ratio are computed on this grid.

The smooth F-test p-values of the directed cells form one family. p_adjust adjusts them once with stats::p.adjust() and the result is stored beside the raw values as association_p_adj. A failed cell has no p-value and stays out of the family, so the family is the cells that produced a p-value, at most k(k - 1). Under "none" the adjusted columns equal the raw ones.

directional_complexity_yx is (EDF_yx - EDF_xy) / (EDF_yx + EDF_xy) in [-1, 1], positive when the fit of var_j on var_i is the more complex direction; directional_complexity_xy is its negative and asymmetry_index its magnitude. A is unreliable when either direction had its basis dimension capped (k_capped_yx or k_capped_xy), either at the first fit or when an auto_refit_k refit reached the cap (k_at_cap in the k-check columns).

shape_borderline is the smallest absolute gap between a shape index and a classification cutoff whose crossing by that index alone would change the shape_category label, on the index's own scale: M and C against the monotonicity and curvature cutoffs, the flat-range ratio against the flat cutoff. A small value marks a label that a slightly different smooth would have moved. Only flat and the monotone labels of a curve with no turning point and no inflection carry a margin. Every other label is decided by the turning-point and inflection counts and has NA: the margin does not measure how near a count is to changing.
$data is the cleaned copy of the columns used: the fitted variables, the adjust variables and every factor, character or logical column of data, under the original column names, after infinite values are set to NA and character or logical adjust columns are converted to factor. It keeps every row under na_action = "pairwise" and the listwise-complete rows under "complete"; a cell's raw points are its two columns on the rows complete for them and the adjust columns.

Value

A list of class janusplot_fit with components, in order:

vars Character vector of the variables fitted, in matrix order.

pairs List with one element per unordered pair i < j. Each element carries i, j, var_i, var_j, fit_yx, fit_xy (NULL unless keep_fits = TRUE), the prediction grids pred_yx, pred_xy (data frames with x, fit, se, lo, hi on the response scale, se on the link scale, and fit_link when the link is not the identity), the partial residuals partial_yx, partial_xy (numeric, one per row the cell was fitted on; NULL without adjust or with a non-identity link), edf_yx, edf_xy, association_p_yx, association_p_xy, association_p_adj_yx, association_p_adj_xy, dev_exp_yx, dev_exp_xy, deviance_gap_yx, deviance_gap_xy, n_used, asymmetry_index, k_capped_yx, k_capped_xy, directional_complexity_yx, directional_complexity_xy, the correlations cor_pearson, cor_spearman, cor_kendall and tie_ratio, the shape indices monotonicity_index_*, convexity_index_*, the counts n_turning_*, n_inflect_*,

the labels shape_*, shape_code_*, shape_archetype_*, shape_monotonic_*, shape_linear_*, the margins shape_borderline_yx, shape_borderline_xy (NA for an indeterminate or count-decided label), the derivative curves deriv_yx, deriv_xy (lists keyed "1", "2" of data frames with x, fit, se, lo, hi, ci_type), the prediction references prediction_reference_yx, prediction_reference_xy, the k-check lists k_check_yx, k_check_xy, k_requested_*, k_used_*, engine, method, error_yx and error_xy.

table Data frame with one row per directed cell: var_x, var_y, position ("upper" if var_y is first), n_used, edf, association_p, association_p_adj, signif (glyph from the adjusted p-value), dev_exp, deviance_gap, asymmetry_index, k_capped_yx, k_capped_xy, directional_complexity (signed for this cell), the correlations, the shape indices, counts and labels, shape_borderline, the k-check columns k_prime, k_index, k_p, k_flag, k_check_status, k_initial, k_final, k_iterations, k_at_cap, k_requested, k_used, k_capped, engine, method and error. summary() returns this table.

data The cleaned copy of the columns used, as described in Details.

adjust The adjust formula, or NULL.

settings Named list of the fit settings as resolved: engine, method, k, bs, na_action, n_grid, derivatives, derivative_ci, derivative_ci_nsim, p_adjust, shape_cutoffs, k_check_thresholds, auto_refit_k, k_max_iter, family and link (names, as in stats::family()), discrete and nthreads.

flags List with failed (data frame of var_x, var_y, reason for cells that carry no fit), k_capped and k_flagged (character vectors of cells as "var_y ~ s(var_x)") and sparse (data frame of var_i, var_j, n_shared for pairs with fewer than 5 shared rows).

fits When keep_fits = TRUE, the list of mgcv objects keyed "i_j" for the cell vars[j] ~ s(vars[i]); otherwise NULL.

call The matched call.

print() returns x invisibly; summary() returns the table component.

See Also

janusplot() for the matrix, janusplot_shape_metrics() for the shape indices, janusplot_shape_cutoffs() for the classification thresholds.

Other smooth-associations: janusplot(), janusplot_compare(), janusplot_curves(), janusplot_direction(), janusplot_pair(), janusplot_reference(), janusplot_style(), plot.janusplot_compare(), plot.janusplot_fit()

Examples

```
fit <- janusplot_fit(mtcars[, c("mpg", "hp", "wt")], engine = "gam")
fit
summary(fit)[, c("var_x", "var_y", "edf", "association_p",
                 "shape_category")]
fit$pairs[[1L]]$asymmetry_index
```

janusplot_pair *Draw both directions of pairs of a janusplot fit*

Description

[Stable]

Draws the two directional fits of one or more pairs of a janusplot_fit() object. Each pair is a block of two panels side by side: y ~ s(x) on the left and x ~ s(y) on the right, each with its raw points, band, curve, the model formula as the title and a strip giving the pair's asymmetry index A, the EDF of that direction, the pair's Pearson correlation r, its significance glyph and the rows used (annotations picks among them). Nothing is refitted.

Usage

```
janusplot_pair(
  fit,
  x,
  y,
  display = c("fit", "d1", "d2", "both"),
  group = NULL,
  annotations = NULL,
  arrange = c("beside", "stacked"),
  nrow = NULL,
  ncol = NULL,
  style = janusplot_style()
)
```

Arguments

fit	A janusplot_fit object from janusplot_fit().
x	Character vector of the predictors of the left panels, each one of fit$vars; or a data frame or character matrix with two columns, predictor then response, one row per pair, with y missing.
y	Character vector of the responses of the left panels, the same length as x, element i paired with x[i] and different from it.
display	What the panels show: "fit" (the default) the fitted curves over the data, "d1" or "d2" the first or second derivative of each smooth, which need that order in fit$settings$derivatives, and "both" the two curves overlaid in one z-score panel.
group	Name of a factor, character or logical column of fit$data whose levels colour the raw points, with a legend below the panels, or NULL (the default). Drawing only; the column never enters the model.
annotations	Character vector choosing what the strip of each panel shows under display = "fit", a subset of the annotations of plot.janusplot_fit(). NULL (the default) is c("A", "edf", "r", "signif", "n"); character() writes no strip. The derivative and overlay displays keep their own strips.

arrange	How the two panels of a pair sit: "beside" (the default) puts y ~ s(x) left of x ~ s(y), "stacked" puts it above; see Details. The overlay of display = "both" has one panel per pair and ignores it.
nrow, ncol	Number of pair blocks per stack and number of stacks, or NULL (the default) to derive them as described in Details. When both are given, nrow * ncol must cover every pair.
style	A janusplot_style() object holding every appearance setting.

Details

The left panel of a block always shows y ~ s(x), whichever order the fit stored the pair in. Titles name the model in the column names of the data, with the fit's adjust terms appended, for example Ozone ~ s(Temp) + s(Month, bs = "re"); the curve and band are then population-level, as in janusplot_fit(). The significance glyph is read from the adjusted p-value when the fit has p_adjust other than "none". With style$pair_background = TRUE the panels are tinted by the pair's Pearson correlation as the matrix background is, with the same background_palette and background_strength; the default leaves them plain, since no colour bar is drawn. A fit with adjust draws no background: its curves are adjusted, while the correlation is marginal.

Many pairs are laid out as blocks stacked top to bottom, with the stacks side by side: nrow is the number of blocks per stack and ncol the number of stacks. Give one and the other follows from the number of pairs; give neither and up to five pairs form one stack, more pairs ceiling(n / 5) stacks. Fourteen pairs with nrow = 5L give stacks of five, five and four blocks; stacks that would stay empty are dropped. One pair is a single row of two panels. Every panel shares the text sizes of style$text_size.

arrange = "stacked" turns each block on its side: y ~ s(x) above x ~ s(y). Several pairs then sit side by side, with every forward panel in one row and every reverse panel in the row below, which is the layout for comparing the same direction across pairs. Here ncol is the number of blocks per row and nrow the number of rows of blocks; give neither and up to five pairs form one row.

display = "both" draws one panel per pair in the z-score plane of the pair's rows: x standardised on the horizontal axis and y on the vertical. y ~ s(x) is drawn in style$curve_colour and x ~ s(y) in orange, with its axes swapped into the same plane, so the two curves can be compared directly. One key below the panels names each colour: by its formula for one pair, as vertical on horizontal and horizontal on vertical for several. style$identity_line = TRUE adds the line y = x, which in this plane is the standard-deviation line: when both curves are flatter than it along their own predictor axis, each variable regresses towards the mean of the other. The overlay draws the observed points, also under adjust, because the partial residuals of the two directions differ.

Under adjust with the identity link, the side-by-side panels draw partial residuals (see adjusted_points in janusplot_style()).

Points are drawn in grey, coloured by group with one legend below the panels, or as a light grey binned density when the pair has more rows than style$density_threshold and no group is given.

Value

A patchwork object, which is a ggplot: the blocks of panels in their stacks, the key of the overlay colours for display = "both" and the group legend underneath when group is given.

See Also

janusplot_fit() for the object, plot.janusplot_fit() for the matrix, janusplot_direction() for the residual-independence test of a pair.

Other smooth-associations: janusplot(), janusplot_compare(), janusplot_curves(), janusplot_direction(), janusplot_fit(), janusplot_reference(), janusplot_style(), plot.janusplot_compare(), plot.janusplot_fit()

Examples

```
fit <- janusplot_fit(airquality[, c("Ozone", "Temp", "Wind")],
                     engine = "gam")
janusplot_pair(fit, "Temp", "Ozone")

# Several pairs at once take vectors of x and y, or a two-column data
# frame; vignette("janusplot") shows one such call and display = "both".
```

---

janusplot_reference *Bootstrap reference for the asymmetry index of a pair*

Description

[Experimental]

The asymmetry index A of a pair varies a great deal from sample to sample, and it is not zero for a straight line: a straight line in one direction gives a curved conditional mean in the other whenever the predictor is not Gaussian, and noise lets either smooth bend. janusplot_reference() measures how large A gets from those sources alone. For each pair and each direction it simulates the fitted smooth of that direction with additive noise, keeping the observed predictor, the adjustment and the residual distribution, refits both directional smooths as janusplot_fit() did, and records A.

Usage

```
janusplot_reference(
  fit,
  x = NULL,
  y = NULL,
  B = 99L,
  seed = NULL,
  parallel = FALSE
)

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

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

Arguments

fit	A janusplot_fit object from janusplot_fit().
x, y	Variable names of the pairs to reference: two equal-length character vectors, element i of x paired with element i of y; or a two-column data frame in x with y missing, as for janusplot_pair(). NULL for both (the default) takes every pair of the fit.
B	Positive integer, the number of replicates per mechanism; 99L by default. Use 499 or more for a reported p-value.
seed	Integer seed for the replicates, or NULL (the default) to draw from the current random-number stream. With a seed, the stream in the calling session is restored afterwards.
parallel	Logical. TRUE runs the replicates through future.apply under the current future plan; FALSE (the default) runs them sequentially.
...	Ignored.

Details

For the mechanism x -> y, the cell y ~ s(x) (with the fit's adjust terms) is refitted to the pair's rows. Each replicate draws y* = fitted + e*, with e* resampled from that fit's centred residuals, and keeps x and the adjustment covariates as observed. Both directional smooths, y* ~ s(x) and x ~ s(y*), are then fitted with the engine, method, basis and basis dimension of the original cells, and A* is computed from their EDFs. The mechanism y -> x swaps the roles. When the fitted smooth is a straight line, the mechanism is a straight line with additive noise.

p_value is the share of replicates with A* at least the observed A, computed as (1 + #{A* >= A}) / (1 + B_ok). A large value says that the smooth of that direction, with additive noise of constant spread and these marginals, readily gives an A this large: the index adds nothing to the two fitted smooths. A small value under both mechanisms says neither direction with additive noise reproduces the asymmetry, as with a spread that changes along the predictor. Neither result identifies a causal direction. A_ref_q95 shows how large A gets by chance at this sample size.

The reference needs the Gaussian family with the identity link. It resamples residuals, so the simulated responses are continuous; for a variable with many tied values (see tie_ratio in summary()) the reference is approximate. Each pair costs 4 * B + 2 model fits, so the function is meant for a few chosen pairs; parallel = TRUE spreads the replicates over a future plan.

Value

A data frame of class janusplot_reference with one row per pair and mechanism: var_x, var_y (the pair), mechanism ("var_x -> var_y" or "var_y -> var_x"), n_used, A (the observed index), A_ref_median, A_ref_q95 (the median and 95th percentile of A*), p_value and B_ok (replicates that fitted). The replicate values are kept in the attribute "draws". print() adds a one-line reading per pair; plot() draws the reference distributions with the observed A marked.

See Also

janusplot_fit() for the index, janusplot_direction() for a residual-independence test of the direction of a pair.

Other smooth-associations: janusplot(), janusplot_compare(), janusplot_curves(), janusplot_direction(), janusplot_fit(), janusplot_pair(), janusplot_style(), plot.janusplot_compare(), plot.janusplot_fit()

Examples

```
# A straight line with a uniform predictor: the reverse smooth bends,
# so A is well above zero, and the reference shows the line explains it.
set.seed(1L)
x <- runif(200L, -2, 2)
d <- data.frame(x = x, y = x + rnorm(200L, sd = 0.4))
fit <- janusplot_fit(d, engine = "gam")
ref <- janusplot_reference(fit, B = 49L, seed = 2L)
ref
plot(ref)
```

---

janusplot_shape_cutoffs

Default cutoff thresholds for shape_category *classification*

Description

[Experimental]

Returns the named list of thresholds used to map the continuous monotonicity (M) and convexity (C) indices (plus inflection counts) into a discrete shape_category. Expose so callers can override individual thresholds or pass a fully custom list to janusplot() / janusplot_shape_metrics().

Usage

```
janusplot_shape_cutoffs(...)
```

Arguments

...	Optional named overrides to merge into the defaults.

Value

A named list with numeric thresholds:

mono_strong |M| threshold for a strictly monotone smooth (default 0.9).

mono_mod |M| threshold for a curved-but-monotone smooth (default 0.5).

curv_low |C| below this is considered near-linear curvature (default 0.2).

curv_mod |C| threshold for a clearly curved monotone (default 0.5).

flat range(fit) / sd(y) below this is called flat (default 0.05).

The cutoffs mono_nonmono, mono_s and curv_strong are deprecated. The classifier never read them: non-monotone and S-shaped curves are identified from the turning-point and inflection counts, and peak against trough from the type of the extremum. Passing them raises a deprecation warning and they are dropped from the returned list.

See Also

Other shape-metrics: janusplot_shape_hierarchy(), janusplot_shape_metrics()

Examples

```
janusplot_shape_cutoffs()
janusplot_shape_cutoffs(curv_mod = 0.6, flat = 0.02)
```

---

janusplot_shape_hierarchy *Shape-category taxonomy table*

Description

[Experimental]

Return the full janusplot shape taxonomy as a data frame with four hierarchy columns plus presentation fields. The taxonomy is the single source of truth consumed by the classifier, the cell renderer, the legend plate, and the janusplot_data() output.

Hierarchy columns (finest -> coarsest):

category 27-way fine label (linear_up, skewed_peak, bimodal, ...). Computed per cell by janusplot().

code Unique two-letter ASCII shorthand (safe on any font or typesetting pipeline) – e.g. lu for linear_up.

archetype Seven-family grouping: monotone_linear, monotone_curved, unimodal, wave, multimodal, chaotic, degenerate.

monotonic Three-way coarse classification: monotone / non_monotone / degenerate.

linear Binary: linear / non_linear / degenerate.

The broader tiers (linear/non-linear, monotone/non-monotone) are textbook calculus; the archetype layer maps cleanly to shape-constrained regression vocabulary (Pya & Wood 2015; Meyer 2008) and to dose-response shape categories (Calabrese 2008; Calabrese & Baldwin 2001). The (T, I) dispatch underlying each fine category is a coarsened Morse-theoretic critical-point classification (Milnor 1963).

Curves with a single turning point are split by the type of that extremum, read from the sign of the first derivative on either side of it (negative then positive is a minimum, positive then negative a maximum), and then by the inflection count: a maximum gives inverted_u, skewed_peak, broad_peak or rippled_peak for 0, 1, 2 or 3+ inflections; a minimum gives u_shape, skewed_trough, broad_trough or rippled_trough. The taxonomy is closed under reflecting y (replacing y with -y maps every single-extremum category onto its mirror), which is why each peak category has a trough counterpart.

Usage

```
janusplot_shape_hierarchy()
```

Value

A data frame with 27 rows and columns `category`, `code`, `archetype`, `monotonic`, `linear`, `glyph`, `ascii`, `label`, `gloss`.

References

Calabrese, E. J. (2008). Hormesis: why it is important to toxicology and toxicologists. *Environmental Toxicology and Chemistry*, **27**(7), 1451-1474.

Meyer, M. C. (2008). Inference using shape-restricted regression splines. *Annals of Applied Statistics*, **2**(3), 1013-1033.

Milnor, J. (1963). *Morse Theory*. Princeton University Press.

Pya, N., & Wood, S. N. (2015). Shape constrained additive models. *Statistics and Computing*, **25**(3), 543-559.

See Also

Other shape-metrics: `janusplot_shape_cutoffs()`, `janusplot_shape_metrics()`

Examples

```
tax <- janusplot_shape_hierarchy()
head(tax[, c("category", "code", "archetype", "monotonic", "linear")])
# Count how many categories live in each archetype
table(tax$archetype)
```

---

janusplot_shape_metrics

Shape metrics for a fitted univariate smooth

Description

[Experimental]

Compute the continuous monotonicity and convexity indices, inflection and turning-point counts, and rule-based shape category for a fitted univariate smooth. Works on either a per-pair fit object returned from the janusplot internal machinery or a freshly fitted `mgcv::gam()` with a single `s()` term.

Both indices are bounded in `[-1, 1]` and weighted by the empirical density of the predictor:

- `monotonicity_index` (paper symbol `M`). Let `f` be the fitted smooth evaluated on a dense grid of `n_grid` equally-spaced points across the predictor range, `f'` its numerical first derivative, and `w` the empirical density of the predictor on the same grid with `sum(w) = 1`. Then `monotonicity_index = sum(w * f') / sum(w * |f'|) in [-1, 1]`. `+1` is strictly increasing, `-1` strictly decreasing, `0` non-monotone.

- `convexity_index` (paper symbol `C`). With `f''` the numerical second derivative on the same grid, `convexity_index = sum(w * f'') / sum(w * |f''|) in [-1, 1]`. `+1` is globally convex (bowl-up), `-1` globally concave (bowl-down), `0` inflection-dominated (S-curve, sine, flat).

Both indices are scale-invariant (replacing `y -> a*y + b` leaves them unchanged) and density-weighted so they describe the smooth *where the data actually live*, not extrapolated tails.

Usage

```
janusplot_shape_metrics(
  fit,
  x_name = NULL,
  newdata = NULL,
  n_grid = 200L,
  cutoffs = janusplot_shape_cutoffs()
)
```

Arguments

`fit`	Either a list returned by a janusplot pair-fit helper (must contain `pred` and `raw`), or a fitted `mgcv::gam()` with a single `s(x)` term.
`x_name`	Character. Column name of the predictor when `fit` is a `mgcv::gam()` object. Ignored for pair-fit lists.
`newdata`	Optional data frame supplying the raw predictor values used for density weighting when `fit` is a `mgcv::gam()` object. If `NULL`, the model frame is used.
`n_grid`	Integer. Prediction grid length when `fit` is a `mgcv::gam()` object. Default `200L`.
`cutoffs`	Named list of classification thresholds; see `janusplot_shape_cutoffs()`. Default uses package defaults.

Value

A named list with components:

`monotonicity_index` `M` in `[-1, 1]`. See Description.

`convexity_index` `C` in `[-1, 1]`. See Description.

`n_turning_points` Integer count of lobe-mass-weighted sign changes of `f'`. Equals the number of interior extrema.

`n_inflections` Integer count of lobe-mass-weighted sign changes of `f''`.

`flat_range_ratio` `range(f) / sd(y)` – small values indicate a degenerate flat smooth.

`extremum_type` `"minimum"` or `"maximum"` when exactly one turning point is counted (sign of `f'` negative then positive, or positive then negative, across it); `NA` otherwise.

`shape_category` One of 27 labels from `janusplot_shape_hierarchy()` dispatched on `(n_turning_points, n_inflecti` with `(monotonicity_index, convexity_index)` disambiguation for the monotone case and `extremum_type` separating the peak family (`inverted_u`, `skewed_peak`, `broad_peak`, `rippled_peak`) from the trough family (`u_shape`, `skewed_trough`, `broad_trough`, `rippled_trough`) for a single turning point.

See Also

janusplot_shape_cutoffs(), janusplot(), janusplot_data().

Other shape-metrics: janusplot_shape_cutoffs(), janusplot_shape_hierarchy()

Examples

```
# On a fitted gam
set.seed(2026L)
n  <- 200L
x  <- stats::runif(n, 0, 10)
y  <- log1p(x) + stats::rnorm(n, sd = 0.3)
d  <- data.frame(x = x, y = y)
fit <- mgcv::gam(y ~ s(x), data = d, method = "REML")
janusplot_shape_metrics(fit, x_name = "x", newdata = d)
```

---

janusplot_shape_sensitivity

Shape-recognition sensitivity study

Description

[Experimental]

Run a full-factorial sensitivity sweep for the janusplot 27-category shape classifier. For each combination of ground-truth shape, sample size n, noise level sigma, and replicate, the sweep:

1. Generates n points from the noiseless canonical curve on [0, 1] + Gaussian noise with SD = sigma (fraction of the y-range, so signal-to-noise is comparable across shapes).
2. Fits mgcv::gam(y ~ s(x), method = "REML").
3. Runs janusplot_shape_metrics() to classify the fitted smooth.
4. Records correctness at both the fine (27-category) and archetype (7-family) levels.

The result is a data frame of class janusplot_sensitivity. summary() aggregates it to a per-cell accuracy table and plot() draws one of four diagnostics.

The function is the package-native implementation of simulation/scripts/scenario_4_shape_recognition.R. A small precomputed dataset is shipped as shape_sensitivity_demo for downstream examples without requiring users to re-run the sweep.

Usage

```
janusplot_shape_sensitivity(
  shapes = c("linear_up", "linear_down", "convex_up", "concave_up", "convex_down",
    "concave_down", "s_shape", "u_shape", "inverted_u", "skewed_peak", "broad_peak",
      "wave", "bimodal", "bi_wave"),
  n_grid = c(50L, 100L, 200L, 500L),
  sigma_grid = c(0.02, 0.05, 0.1, 0.2, 0.4),
  n_rep = 200L,
```

```
  cutoffs = janusplot_shape_cutoffs(),
  parallel = FALSE,
  seed = 2026L,
  verbose = interactive()
)

## S3 method for class 'janusplot_sensitivity'
summary(object, level = c("fine", "archetype"), ...)

## S3 method for class 'janusplot_sensitivity'
plot(
  x,
 type = c("confusion_fine", "confusion_archetype", "accuracy_grid", "recovery_curves"),
  ...
)
```

Arguments

shapes	Character vector of ground-truth shape names, any subset of the 14 canonical shapes in the default. They span five archetypes (monotone_linear, monotone_curved, unimodal, wave, multimodal); chaotic and degenerate have no realistic deterministic generator. The default is every shape, and NULL is equivalent to the default.
n_grid	Integer vector of sample sizes. Default c(50L, 100L, 200L, 500L).
sigma_grid	Numeric vector of noise levels (fraction of the y-range). Default c(0.02, 0.05, 0.10, 0.20, 0.40).
n_rep	Integer. Replicates per cell. Default 200L.
cutoffs	Named list of classification thresholds; see janusplot_shape_cutoffs().
parallel	Logical. If TRUE and future.apply is installed, dispatch replicates in parallel. The caller is responsible for configuring future::plan() (e.g. future::plan(future::multisession)
seed	Integer. Base seed – each fit uses seed + row_index so results are reproducible and cell-permutation-invariant.
verbose	Logical. Print progress messages to the console. Default is interactive().
object, x	A janusplot_sensitivity data frame returned by janusplot_shape_sensitivity(), or a plain data frame with the same columns.
level	The level at which summary() scores correctness: "fine" (default, the 27-category label) or "archetype" (the 7-family label).
...	Unused; present for compatibility with the generics.
type	The diagnostic plot() draws: "confusion_fine" 27 x (|shapes|) confusion matrix at the fine category level – rows = ground truth, columns = predicted, cells coloured by P(pred | truth). "confusion_archetype" 7 x 7 confusion matrix at the archetype level. "accuracy_grid" per-shape heatmap of archetype-level accuracy across the (n, sigma) design.

"recovery_curves" accuracy as a function of sigma, one line per sample size, faceted by shape.

Value

janusplot_shape_sensitivity() returns a data frame of class c("janusplot_sensitivity", "data.frame") with one row per fit. Columns:

truth Ground-truth shape name.

n Sample size for this fit.

sigma Noise level for this fit.

seed RNG seed used.

predicted Classifier output at the fine (27-category) level.

correct Logical – does predicted == truth?

archetype_truth Expected archetype for truth.

archetype_pred Archetype of predicted.

archetype_correct Logical – archetype-level correctness.

monotonicity_index Monotonicity index M (see janusplot_shape_metrics()).

convexity_index Convexity index C (see janusplot_shape_metrics()).

n_turn, n_inflect Recovered turning-point and inflection counts.

error "gam_fit_failed" when mgcv::gam() errored; NA otherwise.

summary() returns a data frame with columns truth, n, sigma and accuracy, one row per cell of the design, where accuracy is the mean correctness at level.

plot() returns a ggplot2::ggplot object.

See Also

shape_sensitivity_demo, janusplot_shape_hierarchy().

Examples

```
# Tiny-run smoke test (< 2 seconds): 3 shapes x 2 n x 2 sigma x 5 reps.
res <- janusplot_shape_sensitivity(
  shapes     = c("linear_up", "u_shape", "wave"),
  n_grid     = c(100L, 200L),
  sigma_grid = c(0.05, 0.20),
  n_rep      = 5L,
  verbose    = FALSE
)
head(res)
summary(res, level = "archetype")

# The precomputed demo carries the class, so the methods apply directly.
data("shape_sensitivity_demo", package = "janusplot")
head(summary(shape_sensitivity_demo, level = "archetype"))
plot(shape_sensitivity_demo, "recovery_curves")
```

janusplot_style *Appearance settings for janusplot views*

Description

[Stable]

Collects every appearance knob of `plot.janusplot_fit()`, `janusplot_pair()` and `janusplot()` into one validated object. Model settings live in `janusplot_fit()`, what the matrix encodes lives in `plot()`'s `colour_by`, `background_by` and `annotations`, and how it is drawn lives here.

Usage

```
janusplot_style(
  colour_channel = c("border", "tile", "fill"),
  palette = NULL,
  background_palette = "BuOr",
  background_strength = 0.35,
  annotation_position = c("strip", "corners"),
  show_data = TRUE,
  adjusted_points = c("partial", "raw"),
  identity_line = FALSE,
  show_ci = TRUE,
  point_size = 0.4,
  point_alpha = 0.25,
  curve_colour = "#08306b",
  curve_width = 2,
  band_alpha = 0.2,
  border_width = NULL,
  density_threshold = 2000L,
  group_palette = "Okabe-Ito",
  group_max_levels = 8L,
  labels = c("border", "none"),
  label_srt = 45,
  diagonal = c("auto", "blank", "name", "density"),
  glyph_style = c("ascii", "unicode"),
  show_shape_legend = "auto",
  show_glossary = TRUE,
  show_key = TRUE,
  legend_position = c("right", "bottom"),
  pair_background = FALSE,
  text_size = "auto",
  base_family = "",
  compact = c("auto", "always", "never"),
  compact_threshold = 12L,
  compact_levels = NULL,
  focus_dim_alpha = 0.25
```

```
)

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

Arguments

colour_channel	Where the colour_by value is painted: "border" (the default) colours the panel border, "tile" fills a small bottom-right corner tile and leaves the border grey, "fill" washes the whole cell at alpha 0.45 and needs background_by = "none".
palette	Name of the colour palette for the colour_by scale, or NULL (the default) for "RdBu" when colour_by is a correlation and "viridis" otherwise. Sequential choices: "viridis", "magma", "inferno", "plasma", "cividis", "mako", "rocket", "turbo" (not colour-blind safe), "YlOrRd", "YlGnBu", "Blues", "Greens". Diverging choices: "RdYlBu", "RdBu", "PuOr", "Spectral" (not colour-blind safe). A sequential palette under a correlation colour_by is replaced by "RdBu".
background_palette	Diverging palette of the correlation background, one of "BuOr" (the default: blue negative, orange positive; its two sides stay the furthest apart under simulated red-green and blue-yellow colour-vision deficiency), "RdBu" (blue negative, red positive), "RdYlBu", "PuOr" and "Spectral".
background_strength	Number in (0, 1], 0.35 by default: how far from white towards the palette end the background of a cell with correlation of magnitude one is drawn. A cell with correlation r is drawn abs(r) * background_strength of the way, as an opaque colour.
annotation_position	Where the annotations of plot() are written: "strip" (the default) in a thin strip above the data area, "corners" inside the panel where 0.2.0 placed them (A, EDF, r and n bottom-left, significance top-right, shape code top-left, shape glyph bottom-right).
show_data	Logical. TRUE (the default) draws the raw points of each cell behind the curve on fit cells; derivative cells never carry points.
adjusted_points	Points of a fit with adjust: "partial" (the default) draws partial residuals, each observation with its own adjustment replaced by the average adjustment, so the points sit around the curve as they do without adjustment; "raw" draws the observed values. Partial residuals need the identity link; with another link the points stay raw.
identity_line	Logical. TRUE adds the dashed line y = x to the z-score panels of janusplot_pair() with display = "both". There it is the standard-deviation line: a curve flatter than it, read along its own predictor axis, regresses towards the mean. FALSE (the default) leaves it out.
show_ci	Logical. TRUE (the default) draws the 95 percent band around the curve, and around a derivative curve when the fit stored one.

`point_size` Size of the raw points; `0.4` by default. `janusplot_pair()` draws the points at this size. The matrix views draw them without an outline and shrink them with the variable count, as they shrink the text: at the default size a point is about 1.25 points across at four variables and 0.85 at eight.

`point_alpha` Opacity of the raw points in `[0, 1]`; `0.25` by default.

`curve_colour` Colour of the fitted curve and its band; the dark blue `"#08306b"` by default.

`curve_width` Width of the fitted curve in points; `2` by default.

`band_alpha` Opacity of the band in `[0, 1]`; `0.20` by default.

`border_width` Width in points of a border that carries a colour, or `NULL` (the default) for the width scaled with the variable count described in Details.

`density_threshold`
Positive integer. A cell whose row count exceeds it draws the points as a binned density (`ggplot2::geom_bin2d()` with 40 bins) instead of points; `2000L` by default. Cells coloured by `group` always draw points.

`group_palette` Colours for the levels of `plot()`'s `group` column: the name of a palette known to `grDevices::palette.colors()` (`"Okabe-Ito"` by default) or a character vector of colours with at least one colour per level.

`group_max_levels`
Positive integer. The largest number of levels a `group` column may have; `8L` by default.

`labels` Where variable names are drawn: `"border"` (the default) along the top and left margins, `"none"` nowhere.

`label_srt` Rotation in degrees of the top border labels; `45` by default.

`diagonal` Content of the diagonal cells: `"auto"` (the default) and `"density"` draw the kernel density of the variable with a rug of its raw values, plus one density per level when the plot has a `group`; `"blank"` leaves them empty and `"name"` writes the variable name.

`glyph_style` `"ascii"` (the default) or `"unicode"` for the shape glyphs, the derivative order labels and the k-check warning sign.

`show_shape_legend`
`"auto"` (the default) attaches the shape-categories legend below the matrix when the cells carry the two-letter shape code it decodes (`annotations` includes `"code"`); `TRUE` always attaches it, `FALSE` never.

`show_glossary` Logical. `TRUE` (the default) attaches a caption describing the abbreviations and encodings on the plot.

`show_key` Logical. `TRUE` (the default) draws in the top-left corner of the matrix a two-by-two key with the first two variables, showing that every cell is `row ~ s(column)`. It needs `labels = "border"`.

`legend_position`
Where the matrix views put the colour bars: `"right"` (the default) in a column beside the matrix, which suits a wide screen; `"bottom"` as horizontal bars under the matrix, beside the group legend, which suits a figure one page wide. Under `"bottom"` the key moves from the corner to the left of the bars, so the top border holds the variable names alone and the cells take the full width.

pair_background
: Logical. TRUE tints the panels of janusplot_pair() by the pair's Pearson correlation, as background_by = "pearson" tints the matrix. FALSE (the default) leaves them plain, because the pair view draws no colour bar; add "r" to annotations to print the correlation in the strip. A fit with adjust draws no pair background.

text_size
: "auto" (the default), a single base size in points, a named list of element sizes, or NULL for no text; see Details.

base_family
: Font family of every text, "" (the default) for the device's sans-serif font.

compact
: Scale-aware content suppression as the variable count grows: "auto" (the default) climbs the tier ladder at compact_threshold, "always" forces at least tier 1, "never" keeps every cell at tier 0. Tier 1 drops the points and compacts the strip, tier 2 drops the band, tier 3 drops the curve and shows the shape glyph alone.

compact_threshold
: Integer of at least 2. The variable count at which tier 1 starts under compact = "auto"; 12L by default.

compact_levels
: Optional named list with entries t1, t2, t3 giving the variable counts at which each tier starts, with t1 < t2 < t3. NULL (the default) uses compact_threshold, compact_threshold + 6 and compact_threshold + 13.

focus_dim_alpha
: Opacity in [0, 1] of the grey wash on cells outside the focus_by selection; 0.25 by default.

x
: A janusplot_style object.

...
: Ignored by the print() method.

Details

The default cell has a thick panel border carrying the colour_by value, a background tinted by the pair's signed correlation (white at zero, background_strength of the way to the palette's negative or positive end at minus or plus one), a thin strip above the data area with the annotations in small grey text, grey points behind a dark blue curve with a 20 percent band, and, for cells with more than density_threshold rows, a light grey binned density in place of the points. A key in the empty top-left corner states the convention of every cell, row ~ s(column).

text_size sets every text of the view. "auto" scales the strip and the diagonal names with the variable count and keeps the strip at 5 points or more. A single number is a base size in points: the strip and the axis text are 0.8 of it, the border labels 1.0, the diagonal names 1.2, the legends and the glossary 0.9, the corner key 0.75 and the title 1.3. A named list sets elements one by one: strip (the strip and the in-panel annotations), labels (the border variable names), diagonal (the names on the diagonal), legend (the colour bars and the group legend), key (the corner key), glossary (the caption), title (the title) and axis (the axes of the pair view). Each element is a size in points, "auto", or NULL, which removes that text; elements not named stay "auto". text_size = NULL removes every text.

Widths are in points, as for text: a width w is drawn as the ggplot2 linewidth w / ggplot2::.pt. border_width = NULL scales the border with the variable count: 6.8 points (linewidth 2.4) at four variables, thicker for fewer, thinner for more, and never below 2.8 points (linewidth 1.0).

`colour_channel = "fill"` with `colour_by = "pearson"` and `background_by = "none"` reproduces the 0.2.0 look.

Value

A named list of class `janusplot_style` holding the 33 values above, with `text_size` stored as given. `print()` lists the values that differ from the defaults and returns `x` invisibly.

See Also

`plot.janusplot_fit()` and `janusplot()`, which take the object as `style`.

Other smooth-associations: `janusplot()`, `janusplot_compare()`, `janusplot_curves()`, `janusplot_direction()`, `janusplot_fit()`, `janusplot_pair()`, `janusplot_reference()`, `plot.janusplot_compare()`, `plot.janusplot_fit()`

Examples

```
janusplot_style()
janusplot_style(colour_channel = "tile", point_alpha = 0.4)

# Every text from one base size; no glossary text and no corner key.
janusplot_style(text_size = list(strip = 7, glossary = NULL, key = NULL))

fit <- janusplot_fit(mtcars[, c("mpg", "hp", "wt")], engine = "gam")
# A figure for a printed page: legends under the matrix, level column
# names, and no title or glossary, which the figure caption replaces.
plot(fit, style = janusplot_style(
  legend_position = "bottom", label_srt = 0, show_glossary = FALSE,
  text_size = list(title = NULL)
))
```

`plot.janusplot_compare` *Draw two fits of the same variables in one matrix*

Description

[Stable]

Draws the matrix of a `janusplot_compare()` object with both fitted curves in every off-diagonal cell: the first fit as a grey dashed curve, the second as the solid curve with its band over the second fit's raw points. A red border marks the cells whose trend changed (rising, falling or non-monotone), and the strip above each cell gives the EDF of both fits. Nothing is refitted.

Usage

```
## S3 method for class 'janusplot_compare'
plot(
  x,
  group = NULL,
  order = c("original", "hclust", "alphabetical"),
  style = janusplot_style(),
  ...
)
```

Arguments

x	A janusplot_compare object.
group	Name of a factor, character or logical column of xfit_bdata whose levels colour the raw points, with one legend below the matrix, or NULL (the default). Drawing only. The column must exist in xfit_bdata and have at most style$group_max_levels levels.
order	Order of the variables in the matrix: "original" (the default, the order of xfit_bvars), "hclust" (hierarchical clustering of absolute Pearson correlations in xfit_bdata) or "alphabetical".
style	A janusplot_style() object. The second fit's curve and band take curve_colour and band_alpha; the points, density threshold, labels, diagonal, glyph style and compact tiers apply as in plot.janusplot_fit().
...	Must be empty. An unknown name is an error.

Details

The layout follows plot.janusplot_fit(): every cell [r, c] shows vars[r] ~ s(vars[c]), over the variables of the second fit, with the key of that convention in the top-left corner, or under the matrix with janusplot_style(legend_position = "bottom"). There is no colour scale and no background. The caption keys the two curves by x$labels when style$show_glossary is TRUE.

Value

A patchwork object, which is a ggplot: the matrix with its border labels, the group legend when group is given, the shape-categories legend when style$show_shape_legend is TRUE ("auto" leaves it off: the compare cells carry no shape code), a title naming the two fits and the key caption when style$show_glossary is TRUE.

See Also

janusplot_compare() for the object, plot.janusplot_fit() for the matrix of one fit.

Other smooth-associations: janusplot(), janusplot_compare(), janusplot_curves(), janusplot_direction(), janusplot_fit(), janusplot_pair(), janusplot_reference(), janusplot_style(), plot.janusplot_fit()

Examples

```
set.seed(2026L)
g <- rep(c("g1", "g2", "g3"), each = 40L)
mu <- c(g1 = 0, g2 = 2, g3 = 4)[g]
d <- data.frame(x = mu + stats::rnorm(120L), g = factor(g))
d$y <- -1.5 * mu + 0.8 * (d$x - mu) + stats::rnorm(120L, sd = 0.4)
d$z <- sin(d$x) + stats::rnorm(120L, sd = 0.3)
vars <- c("x", "y", "z")
fit_a <- janusplot_fit(d, vars, engine = "gam", derivatives = integer())
fit_b <- janusplot_fit(d, vars, adjust = ~ g, engine = "gam",
                       derivatives = integer())
cmp <- janusplot_compare(fit_a, fit_b,
                         labels = c("marginal", "group-adjusted"))
plot(cmp, group = "g")
```

---

plot.janusplot_fit *Draw the matrix of a janusplot fit*

Description

[Stable]

Draws the asymmetric matrix of a janusplot_fit() object: one cell per directed pair, the fitted curve with its band over the raw points, the variable names along the top and left borders. Nothing is refitted; every number on the plot is in the object.

Usage

```
## S3 method for class 'janusplot_fit'
plot(
  x,
  colour_by = c("nonlinearity", "asymmetry", "pearson", "spearman", "kendall",
    "deviance_gap", "none"),
  background_by = c("pearson", "spearman", "kendall", "none"),
  annotations = NULL,
  order = c("original", "hclust", "alphabetical"),
  display = c("fit", "d1", "d2"),
  group = NULL,
  focus_by = NA_character_,
  focus_threshold = "q90",
  axes = c("original", "standardised", "centred", "rank"),
  style = janusplot_style(),
  ...
)
```

Arguments

x	A janusplot_fit object.
colour_by	The quantity painted on each cell: "nonlinearity" (the default; EDF minus 1 of the cell's smooth, sequential from 0), "asymmetry" (the pair's index A in [0, 1], the same in both cells), "pearson", "spearman" or "kendall" (the pair's correlation on a diverging scale over [-1, 1]), "deviance_gap" (deviance explained by the smooth minus the straight line's) or "none".
background_by	The correlation that tints each cell's background: "pearson", "spearman", "kendall", or "none" for no background. Left unset, it is "pearson" for a marginal fit and "none" for a fit with adjust, whose curves are adjusted while the correlations are marginal; set it to show the marginal correlation behind adjusted curves. Independent of colour_by. It cannot be combined with janusplot_style(colour_channel = "fill"), which paints the same area.
annotations	Character vector, a subset of c("A", "edf", "signif", "code", "shape", "k_warn", "n", "r", "formula"), written in the strip of every cell (or in its corners under janusplot_style(annotation_position = "corners")): the asymmetry index, the EDF, the significance glyph of the smooth F-test (*** p < .001, ** p < .01, * p < .05, . p < .1, from the adjusted p-value when the fit has p_adjust other than "none"), the two-letter shape code, the shape glyph, a warning sign on cells flagged by the k-check or capped at the distinct predictor count, the rows used, the background_by correlation of the pair (Pearson when background_by = "none"), printed "r -0.03" ("marginal r -0.03" for a fit with adjust), and the cell's formula, printed Temp ~ Ozone. NULL (the default) picks by matrix size: c("A", "signif") up to twelve variables, "signif" from thirteen, where 5-point strips of neighbouring cells would touch; character() writes none.
order	Order of the variables in the matrix: "original" (the default, the order of x$vars), "hclust" (hierarchical clustering of absolute Pearson correlations) or "alphabetical". Drawing only; the fit is unchanged.
display	What every off-diagonal cell shows: "fit" (the default), "d1" (the first derivative of the smooth, whose zero crossings are turning points) or "d2" (the second derivative, whose zero crossings are inflections). A derivative view needs the fit to carry that order in x$settings$derivatives.
group	Name of a factor, character or logical column of x$data whose levels colour the raw points, with one legend below the matrix, or NULL (the default). Drawing only; the column never enters the model. The column must exist in x$data and have at most style$group_max_levels levels.
focus_by	NA (the default, no filter) or one of "asymmetry", "edf", "non_linearity" (EDF minus 1) and "k_flag". Cells whose metric falls below focus_threshold are washed in grey at style$focus_dim_alpha; the matrix keeps its shape. A visual filter only; the fit and its table are unchanged.
focus_threshold	A quantile string, "q90" by default (taken over the off-diagonal cells), or a single numeric cutoff. Ignored when focus_by is NA.
axes	Scale of the drawn axes: "original" (the default), "standardised" ((x - mean) / sd per variable, border label "var (z)"), "centred" (x - mean, label

	`"var (centred)"`) or `"rank"` (empirical rank, label `"rank(var)"`). Drawing only; the same transform is applied to the points, the curve and the band.
`style`	A `janusplot_style()` object holding every appearance setting.
`...`	Must be empty. An unknown name is an error, so a misspelt style knob cannot pass silently.

Details

Every off-diagonal cell `[r, c]` shows `vars[r] ~ s(vars[c])`: the variable named left of the row is the response on the vertical axis and the variable named above the column is the predictor on the horizontal axis, as in `graphics::pairs()`. Cells `[r, c]` and `[c, r]` are the two directions of one pair and differ by design. A key in the top-left corner shows the convention with the first two variables (`style$show_key`).

The `colour_by` value is painted on the border, a corner tile or the whole cell per `style$colour_channel`, with one colour bar per matrix. The `background_by` correlation tints the panel behind the points, band and curve: white at zero and `style$background_strength` of the way to the negative or positive end of `style$background_palette` at minus or plus one, with its own colour bar over `[-1, 1]`. A strip above each cell carries the `annotations`; the asymmetry index belongs to the pair and is written once, above the cell in the upper triangle. The diagonal shows the kernel density of each variable with a rug of its values, and with `group` one density per level, scaled by the level's share of the rows so the coloured curves sum to the grey pooled density (`style$diagonal`).

The default sizes are set for a figure one page wide (170 mm). At one journal column (85 mm), draw at most four variables with `janusplot_style(text_size = 6, show_shape_legend = FALSE, show_glossary = FALSE)`.

The colour bars sit in a column to the right of the matrix, which suits a wide screen. For a printed page, `janusplot_style(legend_position = "bottom", label_srt = 0)` puts the bars and the key under the matrix, so the cells take the full width, and `show_glossary = FALSE` leaves the explanation to the figure caption.

`colour_by = "edf"` is accepted for one release and mapped to `"nonlinearity"` with a deprecation message.

Value

A patchwork object, which is a ggplot: the matrix with its border labels and corner key, the colour bar when `colour_by` is not `"none"`, the background colour bar when `background_by` is not `"none"`, the group legend when `group` is given, the shape-categories legend when `style$show_shape_legend` is `TRUE`, or `"auto"` and `annotations` include `"code"`, a title naming the displayed quantity and the glossary caption when `style$show_glossary` is `TRUE`.

See Also

`janusplot_fit()` for the object, `janusplot_style()` for the appearance settings, `janusplot()` for the one-call form.

Other smooth-associations: `janusplot()`, `janusplot_compare()`, `janusplot_curves()`, `janusplot_direction()`, `janusplot_fit()`, `janusplot_pair()`, `janusplot_reference()`, `janusplot_style()`, `plot.janusplot_compare()`

Examples

```
fit <- janusplot_fit(mtcars[, c("mpg", "hp", "wt")], engine = "gam")
plot(fit)

# Redrawing with another colour_by, annotations, group or display
# needs no refit; vignette("janusplot") shows each of them.
```

---

shape_sensitivity_demo

Precomputed shape-recognition sensitivity results (demo)

Description

Raw output from a small-footprint invocation of janusplot_shape_sensitivity(). Shipped so users can explore the sensitivity API and regenerate every figure in the shape-recognition-sensitivity vignette without having to re-run the sweep themselves. Regenerated via data-raw/shape_sensitivity_demo.R, which runs the design below through janusplot_shape_sensitivity(), so the object carries the janusplot_sensitivity class and summary() and plot() apply to it directly.

Design:

- **Shapes** (6, across 5 of the 7 archetypes – u_shape and inverted_u both roll up to unimodal): linear_up, concave_up, u_shape, inverted_u, wave, bimodal.
- **Sample sizes** (3): c(100, 200, 500).
- **Noise levels** (4): c(0.05, 0.10, 0.20, 0.40) fraction of y-range.
- **Replicates**: 30.
- **Total fits**: 2160.
- **Seed**: 2026.

Usage

```
shape_sensitivity_demo
```

Format

A data frame of class c("janusplot_sensitivity", "data.frame") with 2160 rows and 14 columns – see the "Value" section of janusplot_shape_sensitivity() for the column schema.

See Also

janusplot_shape_sensitivity() and its summary() and plot() methods.

Examples

```
data("shape_sensitivity_demo", package = "janusplot")
head(shape_sensitivity_demo)
plot(shape_sensitivity_demo, "recovery_curves")
```

Index

∗ **datasets**
shape_sensitivity_demo, 37
∗ **shape-metrics**
janusplot_shape_cutoffs, 21
janusplot_shape_hierarchy, 22
janusplot_shape_metrics, 23
∗ **shape-sensitivity**
janusplot_shape_sensitivity, 25
∗ **smooth-associations**
janusplot, 2
janusplot_compare, 5
janusplot_curves, 7
janusplot_direction, 8
janusplot_fit, 12
janusplot_pair, 17
janusplot_reference, 19
janusplot_style, 28
plot.janusplot_compare, 32
plot.janusplot_fit, 34

future, *20*

ggplot2::geom_bin2d(), *30*
ggplot2::ggplot, *27*
ggplot2::ggsave(), *5*
graphics::pairs(), *2*, *36*
grDevices::palette.colors(), *30*

janusplot, 2
janusplot(), *6*, *8*, *11*, *12*, *16*, *19*, *21*, *22*, *25*, *28*, *32*, *33*, *36*
janusplot_compare, 5
janusplot_compare(), *5*, *8*, *11*, *16*, *19*, *21*, *32*, *33*, *36*
janusplot_curves, 7
janusplot_curves(), *5*, *6*, *11*, *16*, *19*, *21*, *32*, *33*, *36*
janusplot_data(), *25*
janusplot_direction, 8
janusplot_direction(), *5*, *6*, *8*, *16*, *19–21*, *32*, *33*, *36*
janusplot_fit, 12
janusplot_fit(), *2*, *4–9*, *11*, *17–21*, *28*, *32–34*, *36*
janusplot_pair, 17
janusplot_pair(), *5*, *6*, *8*, *11*, *16*, *20*, *21*, *28–33*, *36*
janusplot_reference, 19
janusplot_reference(), *5*, *6*, *8*, *11*, *16*, *19*, *32*, *33*, *36*
janusplot_shape_cutoffs, 21
janusplot_shape_cutoffs(), *14*, *16*, *23–26*
janusplot_shape_hierarchy, 22
janusplot_shape_hierarchy(), *22*, *24*, *25*, *27*
janusplot_shape_metrics, 23
janusplot_shape_metrics(), *16*, *21–23*, *25*, *27*
janusplot_shape_sensitivity, 25
janusplot_shape_sensitivity(), *37*
janusplot_style, 28
janusplot_style(), *4–6*, *8*, *11*, *16*, *18*, *19*, *21*, *33*, *36*

mgcv::bam(), *2*, *4*, *10*, *13*, *14*
mgcv::gam(), *2*, *4*, *10*, *11*, *13*, *14*, *23*, *24*
mgcv::k.check(), *14*

patchwork, *5*, *18*, *33*, *36*
plot.janusplot_compare, 32
plot.janusplot_compare(), *5*, *6*, *8*, *11*, *16*, *19*, *21*, *32*, *36*
plot.janusplot_fit, 34
plot.janusplot_fit(), *2*, *5*, *6*, *8*, *11*, *16*, *17*, *19*, *21*, *28*, *32*, *33*
plot.janusplot_reference *(*janusplot_reference*)*, 19
plot.janusplot_sensitivity *(*janusplot_shape_sensitivity*)*,

25
print.janusplot_compare
(janusplot_compare), 5
print.janusplot_direction
(janusplot_direction), 8
print.janusplot_fit (janusplot_fit), 12
print.janusplot_reference
(janusplot_reference), 19
print.janusplot_style
(janusplot_style), 28

shape_sensitivity_demo, 25, 27, 37
stats::family(), 16
stats::p.adjust(), 14, 15
summary.janusplot_fit (janusplot_fit),
12
summary.janusplot_fit(), 5, 8
summary.janusplot_sensitivity
(janusplot_shape_sensitivity),
25