Package 'janusplot'

May 8, 2026

Type Package

Title Asymmetric Smoothed-Association Matrices via GAM Fits

Version 0.1.0

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_j ~ s(x_i))' and the lower triangle 'gam(x_i ~ s(x_j))'. 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 24-category shape taxonomy quantify the directional difference and qualitative form of each fitted smooth.

License GPL (>= 3)

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, stats, cli (>= 3.6.0), lifecycle, rlang (>= 1.1.0)

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

VignetteBuilder knitr

RoxygenNote 7.3.3

Config/testthat/edition 3

Config/testthat/parallel true

Config/Needs/website pkgdown

LazyData true

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 2026-04-28 18:30:08 UTC

Contents

janusplot 2
janusplot_data 8
janusplot_shape_cutoffs 10
janusplot_shape_hierarchy 11
janusplot_shape_metrics 13
janusplot_shape_sensitivity 14
janusplot_shape_sensitivity_plot 16
janusplot_shape_sensitivity_shapes 17
janusplot_shape_sensitivity_summary 18
shape_sensitivity_demo 18

Index **20**

`janusplot` *Asymmetric smoothed-association matrix*

Description

[Experimental]

Render a pairwise, asymmetric matrix of smoothed associations between numeric variables. Each cell [i, j] where `i != j` shows the fitted spline from `mgcv::gam()`:

- Upper triangle (`i < j`): `gam(x_j ~ s(x_i) + <adjust>)`.
- Lower triangle (`i > j`): `gam(x_i ~ s(x_j) + <adjust>)`.
- Diagonal: blank panel when labels live on the border (default), or a variable-name label when `labels = "diagonal"`.

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

Usage

```
janusplot(
  data,
  vars = NULL,
  adjust = NULL,
  method = "REML",
  k = -1L,
  bs = "tp",
  order = c("original", "hclust", "alphabetical"),
  show_data = TRUE,
  show_ci = TRUE,
  display = c("fit", "d1", "d2"),
  derivative_ci = c("none", "pointwise", "simultaneous"),
  derivative_ci_nsim = 1000L,
  n_grid = NULL,
  colour_by = c("pearson", "spearman", "kendall", "edf", "deviance_gap", "none"),
  fill_by = NULL,
  palette = NULL,
  annotations = c("edf", "A"),
  shape_cutoffs = janusplot_shape_cutoffs(),
  show_shape_legend = TRUE,
  glyph_style = c("ascii", "unicode"),
  labels = c("border", "diagonal", "none"),
  diagonal = c("auto", "blank", "name", "density"),
  label_srt = 45,
  label_cex = 1,
  signif_glyph = TRUE,
  show_asymmetry = NULL,
  na_action = c("pairwise", "complete"),
  parallel = FALSE,
  with_data = FALSE,
  text_scale_diag = 1,
  text_scale_off_diag = 1,
  show_glossary = TRUE,
  glossary_scale = 1,
  ...
)
```

Arguments

`data`	A data frame with numeric columns to include.
`vars`	Character vector of column names to use. `NULL` (default) uses all numeric columns in `data`. Non-numeric columns trigger an error listing offenders.
`adjust`	A one-sided formula RHS giving additional covariates and/or random effects to include in every pairwise GAM. For example, `adjust = ~ s(age) + s(site, bs = "re")` fits `gam(y ~ s(x) + s(age) + s(site, bs = "re"))` for each pair. Default `NULL` fits unadjusted pairwise smooths.

`method` Smoothing-parameter estimation method passed to `mgcv::gam()`. Default `"REML"` per mgcv recommendation.

`k` Integer, or named list mapping variable names to integers. Basis dimension for `s()`. Default `-1L` (mgcv's automatic choice).

`bs` Basis type for `s()`. Default `"tp"` (thin plate).

`order` One of `"original"` (default), `"hclust"` (reorder by hierarchical clustering of Pearson correlations), or `"alphabetical"`.

`show_data` Logical. If `TRUE` (default), overlay raw data points (low alpha) behind each spline. Only applies when `display = "fit"`; derivative panels never overlay raw data.

`show_ci` Logical. If `TRUE` (default), overlay the 95% confidence envelope from `predict(gam, se.fit = TRUE)` on the fit panel (i.e. when `display = "fit"`). CI rendering on derivative panels is controlled separately by `derivative_ci`.

`display` One of `"fit"` (default), `"d1"`, or `"d2"`. Selects which single quantity is rendered in every off-diagonal cell of the matrix.

- `"fit"` — the fitted smooth $\hat{f}(x)$; default, behaviour identical to the pre-derivative release.
- `"d1"` — the first derivative $\hat{f}'(x)$ of the fitted smooth. Zero crossings localise turning points of $\hat{f}$.
- `"d2"` — the second derivative $\hat{f}''(x)$. Zero crossings localise inflection points of $\hat{f}$.

A single matrix shows a single quantity by design: stacked multi-panel cells crowd the matrix at any realistic variable count. To compare fit against derivative, render two or three `janusplot()` calls side-by-side; each call keeps its own `with_data = TRUE` summary table tagged with the `display` column.

Orders $k \geq 3$ are not exposed — higher-order derivatives of penalised regression splines amplify noise and rarely carry usable signal at realistic sample sizes. See `vignette("janusplot")` for the theoretical justification and applied use-cases.

`derivative_ci` One of `"none"` (default), `"pointwise"`, or `"simultaneous"`. Controls whether — and how — a 95% confidence ribbon is drawn underneath the derivative curve when `display %in% c("d1", "d2")`. Ignored when `display = "fit"`.

- `"none"` — no ribbon. The curve and the zero reference line are all you see. Default, because pointwise ribbons overshoot nominal coverage as a joint region and can invite over-reading of local features.
- `"pointwise"` — 95% pointwise ribbon from $\sqrt{\mathrm{diag}(DV_pD^\top)}$ (Wood 2017 §7.2.4). Valid marginally; not a simultaneous statement.
- `"simultaneous"` — 95% simultaneous band via the Monte Carlo construction of Ruppert, Wand & Carroll (2003) popularised for GAMs by Simpson (2018, *Frontiers Ecol. Evol.* 6:149): draw B samples $\tilde{\boldsymbol{\beta}} \sim \mathcal{N}(\hat{\boldsymbol{\beta}}, V_p)$, compute $\max_x |D_i(\tilde{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}})|/\mathrm{se}_i$, and use the $(1-\alpha)$ quantile as a critical multiplier on the pointwise SE. Valid for feature localisation ("where is $\hat{f}'(x)$ significantly non-zero").

`derivative_ci_nsim` Integer. Number of Monte Carlo samples used when `derivative_ci = "simultaneous"`. Default `1000L` — a compromise between coverage accuracy (Simpson 2018 uses

10000) and CPU budget across every pair in a medium-sized matrix. Ignored for any other `derivative_ci`.

`n_grid` Integer or `NULL`. Number of equally-spaced points used to evaluate each fitted smooth (and its derivatives). Default `NULL` resolves to `100` when `display = "fit"` and `200` otherwise, because finite-difference second derivatives visibly degrade below ~ 150 points on moderate-k smooths. Supplying `n_grid` directly overrides both defaults. Larger grids shift the numerical shape-metric values (M, C, turning / inflection counts) slightly because they are computed on this same grid. Shapes and asymmetry are the primary reading; `M`, `C` and the counts are secondary diagnostics and the grid-induced drift is tolerable.

`colour_by` One of `"pearson"` (default), `"spearman"`, `"kendall"`, `"edf"`, `"deviance_gap"`, or `"none"`. Encodes the per-cell fill colour by the chosen scalar. Correlation choices use a diverging palette with limits `c(-1, 1)` and a shared `corr` colour-bar title; `"edf"` and `"deviance_gap"` use a sequential palette labelled by the metric.

`fill_by` Deprecated alias for `colour_by`. If supplied emits a single soft deprecation warning and is forwarded to `colour_by`.

`palette` Character. Colour palette for the cell fill scale. Defaults to `"RdBu"` when `colour_by` is a correlation and `"viridis"` otherwise. Sequential choices: `"viridis"`, `"magma"`, `"inferno"`, `"plasma"`, `"cividis"`, `"mako"`, `"rocket"`, `"turbo"` (not CB-safe), `"YlOrRd"`, `"YlGnBu"`, `"Blues"`, `"Greens"`. Diverging choices: `"RdYlBu"`, `"RdBu"`, `"PuOr"`, `"Spectral"` (not CB-safe). Passing a sequential palette while `colour_by` is a correlation silently upgrades to the default diverging palette.

`annotations` Character vector, a subset of `c("edf", "A", "shape", "code")`. Controls which corner annotations appear on each off-diagonal cell:

- `"code"` — 2-letter ASCII shape code, **top-left** corner.
- `"A"` and `"edf"` — asymmetry index and effective degrees of freedom, stacked **bottom-left**.
- `"shape"` — shape glyph (Unicode or ASCII per `glyph_style`), **bottom-right** corner.

Default `c("edf", "A")`. `"code"` and `"shape"` occupy distinct corners so both can be requested together. See `janusplot_shape_hierarchy()` for the full code list.

`shape_cutoffs` Named list of classification thresholds used to map the continuous shape indices into discrete `shape_category` labels; see `janusplot_shape_cutoffs()`.

`show_shape_legend` Logical. If `TRUE` (default), attach a standing shape-types legend plate below the matrix that illustrates every category in the taxonomy as a canonical thumbnail spline. Independent of `annotations`.

`glyph_style` One of `"ascii"` (default) or `"unicode"`. Controls how cell shape glyphs render when `"shape"` is included in `annotations`. Default is `"ascii"` for maximum portability across typesetting pipelines; switch to `"unicode"` only when the target font is known to cover the curve glyph set.

`labels` One of `"border"` (default), `"diagonal"`, or `"none"`. Controls where variable names are rendered:

- "border" — names along the top (rotated per label_srt) and left margins of the matrix; diagonal cells are left blank. Mirrors corrplot's tl.pos = "lt" convention.
- "diagonal" — names centred on the diagonal cells (the pre-0.1 layout).
- "none" — labels suppressed entirely; diagonal cells blank.

diagonal — One of "auto" (default), "blank", "name", or "density". Controls what is rendered in the diagonal cells of the matrix.

- "auto" — preserves the historical behaviour: variable name when labels = "diagonal", blank otherwise.
- "blank" — empty bordered panel (uniform grid reading).
- "name" — variable name centred in the cell, bold.
- "density" — kernel density of the variable filled in translucent grey, with a rug of raw values along the bottom edge. Mirrors the GGally::ggpairs convention; surfaces tail weight, bimodality, and support clipping that the pairwise smooths alone cannot reveal. Variable names should come from the border (labels = "border", the default) when this mode is on.

label_srt — Numeric. Rotation (degrees) of top labels when labels = "border". Default 45; set to 0 for horizontal or 90 for vertical. Ignored when labels != "border".

label_cex — Positive numeric multiplier on the border-label font size. Default 1. Ignored when labels = "none".

signif_glyph — Logical. If TRUE (default), annotate cells with · / * / ** reflecting the smooth's F-test p-value.

show_asymmetry — Deprecated. Use annotations instead ("A" %in% annotations). When supplied, a soft deprecation warning fires and the argument is merged into annotations.

na_action — One of "pairwise" (default; per-cell complete observations) or "complete" (listwise; all cells use the same rows).

parallel — Logical. If TRUE, use future.apply::future_mapply() to fit pairs in parallel. Requires the future.apply package and a user-configured future::plan(). Default FALSE.

with_data — Logical. If TRUE, return a two-element list list(plot, data) where data is a flat per-cell summary (one row per off-diagonal cell) of everything the plot displays. The data element is always a plain data.frame (base R — no data.table dependency). Default FALSE — in which case only the ggplot is returned.

text_scale_diag — Positive numeric multiplier applied to the diagonal variable-name labels. Default 1. Diagonal labels additionally auto-shrink for long variable names (nchar(var) > 10) so they fit the cell regardless of this value.

text_scale_off_diag — Positive numeric multiplier applied to all off-diagonal annotations (n / EDF readouts, significance glyphs, asymmetry-index labels). Default 1. Use < 1 when cells are small and the annotations crowd the fit line; use > 1 for presentation plots.

show_glossary — Logical. If TRUE (default), attach a multi-line caption below the matrix describing the on-plot abbreviations (n, EDF, A, fill encoding, significance glyphs). Only keys actually displayed are listed.

`glossary_scale` Positive numeric multiplier on the glossary caption font size. Default 1.

`...` Additional arguments passed to `mgcv::gam()`.

Value

If `with_data = FALSE` (default), a ggplot2::ggplot object (via `patchwork::wrap_plots()`) carrying a top-of-matrix title that names the displayed quantity (`"Direct fit"`, `"First derivative f'"`, or `"Second derivative f''"`). If `with_data = TRUE`, a list with two elements: `plot` (the ggplot) and `data` (a tidy table with columns `var_x`, `var_y`, `position`, `n_used`, `edf`, `pvalue`, `signif`, `dev_exp`, `asymmetry_index`, `cor_pearson`, `cor_spearman`, `cor_kendall`, `tie_ratio`, `monotonicity_index`, `convexity_index`, `n_turning_points`, `n_inflections`, `flat_range_ratio`, `shape_category`, `colour_value`, `display`, one row per off-diagonal cell). The `display` column tags which quantity the call rendered, so separate calls for fit / d1 / d2 yield comparable, stackable tables. Derivative *curves* themselves (grid of x, fitted $\hat{f}^{(k)}$, SE) live on `janusplot_data()` — see there.

See Also

`janusplot_data()` for the raw per-cell fits + metrics.

Other smooth-associations: `janusplot_data()`

Examples

```
# Minimal runnable example — 3 variables, 6 asymmetric pairwise GAM fits.
janusplot(mtcars[, c("mpg", "hp", "wt")])


# Heteroscedastic DGP: Pearson r is ~ 0.9 but the inverse fit is
# clearly non-linear, yielding asymmetry index > 0.5.
set.seed(2026L)
n  <- 200L
x1 <- stats::runif(n, 0, 10)
x2 <- x1 + stats::rnorm(n, sd = 0.2 * x1)
janusplot(data.frame(x1 = x1, x2 = x2, x3 = stats::rnorm(n)))

# A single matrix renders a single quantity. To compare the fit
# against its derivatives, render three calls and place them
# side-by-side; each call's title makes the quantity explicit.
set.seed(2026L)
xs <- stats::runif(300L, -3, 3)
df <- data.frame(
  x  = xs,
  y1 = sin(xs)  + stats::rnorm(300L, sd = 0.3),
  y2 = xs^2     + stats::rnorm(300L, sd = 0.6)
)
janusplot(df, display = "fit")
janusplot(df, display = "d1")
janusplot(df, display = "d2")

# Simultaneous CI bands on a derivative panel, per Simpson (2018).
janusplot(df, display = "d1", derivative_ci = "simultaneous")
```

janusplot_data *Raw GAM fits and per-cell metrics for a smoothed-association matrix*

Description

[Experimental]

Companion to janusplot() returning the raw list of GAM fits plus per-cell metrics (EDF, F-test p-value, deviance explained, asymmetry index, pairwise correlations, shape descriptors) without constructing the ggplot. Useful for custom rendering or downstream analysis.

Usage

```
janusplot_data(
  data,
  vars = NULL,
  adjust = NULL,
  method = "REML",
  k = -1L,
  bs = "tp",
  na_action = c("pairwise", "complete"),
  parallel = FALSE,
  keep_fits = FALSE,
  derivatives = integer(),
  derivative_ci = c("pointwise", "none", "simultaneous"),
  derivative_ci_nsim = 1000L,
  n_grid = NULL,
  shape_cutoffs = janusplot_shape_cutoffs(),
  ...
)
```

Arguments

data	A data frame with numeric columns to include.
vars	Character vector of column names to use. NULL (default) uses all numeric columns in data. Non-numeric columns trigger an error listing offenders.
adjust	A one-sided formula RHS giving additional covariates and/or random effects to include in every pairwise GAM. For example, adjust = ~ s(age) + s(site, bs = "re") fits gam(y ~ s(x) + s(age) + s(site, bs = "re")) for each pair. Default NULL fits unadjusted pairwise smooths.
method	Smoothing-parameter estimation method passed to mgcv::gam(). Default "REML" per mgcv recommendation.
k	Integer, or named list mapping variable names to integers. Basis dimension for s(). Default -1L (mgcv's automatic choice).

`bs` Basis type for `s()`. Default `"tp"` (thin plate).

`na_action` One of `"pairwise"` (default; per-cell complete observations) or `"complete"` (listwise; all cells use the same rows).

`parallel` Logical. If `TRUE`, use `future.apply::future_mapply()` to fit pairs in parallel. Requires the `future.apply` package and a user-configured `future::plan()`. Default `FALSE`.

`keep_fits` Logical. If `TRUE`, retain full `mgcv::gam()` model objects in the return (large memory footprint for `k` above ~15). Default `FALSE` — retains summary metrics and prediction grids only.

`derivatives` Integer vector of derivative orders to compute on every pair (subset of `1:2`). Default `integer()` — no derivatives. Unlike `janusplot()`, the data companion can return multiple orders from a single call for programmatic analysis; pass `c(1L, 2L)` to surface both.

`derivative_ci` One of `"none"` (default), `"pointwise"`, or `"simultaneous"`. Controls whether — and how — a 95% confidence ribbon is drawn underneath the derivative curve when `display %in% c("d1", "d2")`. Ignored when `display = "fit"`.

- `"none"` — no ribbon. The curve and the zero reference line are all you see. Default, because pointwise ribbons overshoot nominal coverage as a joint region and can invite over-reading of local features.
- `"pointwise"` — 95% pointwise ribbon from $\sqrt{\mathrm{diag}(DV_pD^\top)}$ (Wood 2017 §7.2.4). Valid marginally; not a simultaneous statement.
- `"simultaneous"` — 95% simultaneous band via the Monte Carlo construction of Ruppert, Wand & Carroll (2003) popularised for GAMs by Simpson (2018, *Frontiers Ecol. Evol.* 6:149): draw B samples $\tilde{\boldsymbol{\beta}} \sim \mathcal{N}(\hat{\boldsymbol{\beta}}, V_p)$, compute $\max_x |D_i(\tilde{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}})|/\mathrm{se}_i$, and use the $(1-\alpha)$ quantile as a critical multiplier on the pointwise SE. Valid for feature localisation ("where is $\hat{f}'(x)$ significantly non-zero").

`derivative_ci_nsim` Integer. Number of Monte Carlo samples used when `derivative_ci = "simultaneous"`. Default `1000L` — a compromise between coverage accuracy (Simpson 2018 uses 10000) and CPU budget across every pair in a medium-sized matrix. Ignored for any other `derivative_ci`.

`n_grid` Integer or `NULL`. Number of equally-spaced points used to evaluate each fitted smooth (and its derivatives). Default `NULL` resolves to `100` when `display = "fit"` and `200` otherwise, because finite-difference second derivatives visibly degrade below ~ 150 points on moderate-k smooths. Supplying `n_grid` directly overrides both defaults. Larger grids shift the numerical shape-metric values (M, C, turning / inflection counts) slightly because they are computed on this same grid. Shapes and asymmetry are the primary reading; `M`, `C` and the counts are secondary diagnostics and the grid-induced drift is tolerable.

`shape_cutoffs` Named list of classification thresholds used to map the continuous shape indices (`monotonicity_index`, `convexity_index`) into discrete `shape_category` labels. Defaults from `janusplot_shape_cutoffs()`.

`...` Additional arguments passed to `mgcv::gam()`.

Value

A list with components:

`vars` Character vector of variables used, in plotted order.

`pairs` List of per-pair results. Each element has `i`, `j`, `var_i`, `var_j`, `fit_yx`, `fit_xy` (NULL if `keep_fits = FALSE`), `pred_yx`, `pred_xy` (data frames with `x`, `fit`, `se`, `lo`, `hi`), `edf_yx`, `edf_xy`, `pvalue_yx`, `pvalue_xy`, `dev_exp_yx`, `dev_exp_xy`, `n_used`, `asymmetry_index`, plus Pearson / Spearman / Kendall correlations (`cor_pearson`, `cor_spearman`, `cor_kendall`), the maximum tie ratio across x and y (`tie_ratio`), and per-direction shape descriptors (`monotonicity_index_yx`, `convexity_index_yx`, `monotonicity_index_xy`, `convexity_index_xy`, `n_turning_yx`, `n_inflect_yx`, `n_turning_xy`, `n_inflect_xy`, `shape_yx`, `shape_xy`). When `derivatives` is non-empty, each pair additionally carries `deriv_yx` and `deriv_xy`, each a named list keyed by order (`"1"`, `"2"`) whose entries are data frames with columns `x`, `fit`, `se`, `lo`, `hi`, `ci_type` matching the schema of `pred_yx` / `pred_xy`. The `ci_type` column records whether the `lo` / `hi` columns are `"pointwise"` (default), `"simultaneous"` (Ruppert–Wand–Carroll / Simpson 2018 critical-multiplier bands), or `"none"`. When `derivative_ci = "simultaneous"`, each derivative frame also carries a `"crit_multiplier"` attribute giving the MC-derived critical multiplier used. See `janusplot_shape_metrics()` for the definition of the monotonicity and convexity indices.

`call` Match call.

See Also

`janusplot()` for the ggplot front-end, `janusplot_shape_metrics()` for the shape-metric primitives.

Other smooth-associations: `janusplot()`

Examples

```
# Per-pair fits + metrics on a small mtcars slice
out <- janusplot_data(mtcars[, c("mpg", "hp", "wt")])
out$pairs[[1L]]$asymmetry_index
out$pairs[[1L]]$cor_spearman
out$pairs[[1L]]$shape_yx
```

---

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`).

`mono_nonmono` `|M|` below this is considered non-monotone (default `0.3`).

`mono_s` `|M|` threshold for labelling an S-shape (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`).

`curv_strong` `|C|` threshold for a U-shape / inverted-U shape (default `0.5`).

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

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` 24-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).

Usage

```
janusplot_shape_hierarchy()
```

Value

A data frame with 24 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.

`shape_category` One of 24 labels from `janusplot_shape_hierarchy()` dispatched on `(n_turning_points, n_inflecti` with `(monotonicity_index, convexity_index)` disambiguation for the monotone case.

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 24-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 (24-category) and archetype (7-family) levels.

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 = NULL,
  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()
)
```

Arguments

shapes	Character vector of ground-truth names from janusplot_shape_sensitivity_shapes(). Default NULL → all 14.
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().

Value

A 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 (24-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.

See Also

janusplot_shape_sensitivity_summary(), janusplot_shape_sensitivity_plot(), janusplot_shape_sensitivity
shape_sensitivity_demo.

Other shape-sensitivity: janusplot_shape_sensitivity_plot(), janusplot_shape_sensitivity_shapes(), janusplot_shape_sensitivity_summary()

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)
janusplot_shape_sensitivity_summary(res, level = "archetype")
```

---

janusplot_shape_sensitivity_plot

Visualise a shape-sensitivity sweep

Description

[Experimental]

Produce one of four diagnostic plots from the raw data frame returned by janusplot_shape_sensitivity():

`"confusion_fine"` 24 x 24 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.

Usage

```
janusplot_shape_sensitivity_plot(
  results,
  type = c("confusion_fine", "confusion_archetype", "accuracy_grid", "recovery_curves")
)
```

Arguments

`results`	Data frame from janusplot_shape_sensitivity() or the precomputed shape_sensitivity_demo.
`type`	One of `"confusion_fine"`, `"confusion_archetype"`, `"accuracy_grid"`, or `"recovery_curves"`.

Value

A ggplot2::ggplot object.

See Also

Other shape-sensitivity: janusplot_shape_sensitivity(), janusplot_shape_sensitivity_shapes(), janusplot_shape_sensitivity_summary()

Examples

```
data("shape_sensitivity_demo", package = "janusplot")
janusplot_shape_sensitivity_plot(shape_sensitivity_demo,
                                 "recovery_curves")
```

---

janusplot_shape_sensitivity_shapes

Canonical ground-truth shapes for the sensitivity study

Description

[Experimental]

Return the names of every canonical ground-truth shape that janusplot_shape_sensitivity() can simulate from. Fourteen shapes spanning five archetypes (monotone_linear, monotone_curved, unimodal, wave, multimodal). The chaotic and degenerate archetypes are out of scope (no realistic deterministic generator).

Usage

```
janusplot_shape_sensitivity_shapes()
```

Value

Character vector of length 14 — the generator names.

See Also

janusplot_shape_sensitivity(), janusplot_shape_hierarchy().

Other shape-sensitivity: janusplot_shape_sensitivity(), janusplot_shape_sensitivity_plot(), janusplot_shape_sensitivity_summary()

Examples

```
janusplot_shape_sensitivity_shapes()
```

janusplot_shape_sensitivity_summary

Summarise a shape-sensitivity sweep

Description

[Experimental]

Aggregate the raw output of `janusplot_shape_sensitivity()` into a per-cell mean-accuracy table at either the fine (24-category) or archetype (7-family) level.

Usage

```
janusplot_shape_sensitivity_summary(results, level = c("fine", "archetype"))
```

Arguments

`results`	Data frame returned by `janusplot_shape_sensitivity()`.
`level`	One of `"fine"` (default) or `"archetype"`.

Value

A data frame with columns `truth`, `n`, `sigma`, `accuracy`.

See Also

Other shape-sensitivity: `janusplot_shape_sensitivity()`, `janusplot_shape_sensitivity_plot()`, `janusplot_shape_sensitivity_shapes()`

Examples

```
data("shape_sensitivity_demo", package = "janusplot")
head(janusplot_shape_sensitivity_summary(shape_sensitivity_demo,
                                         level = "archetype"))
```

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.

Design:

- **Shapes** (6, one per non-degenerate archetype): 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 with 2160 rows and 14 columns — see the "Value" section of janusplot_shape_sensitivity() for the column schema.

See Also

janusplot_shape_sensitivity(), janusplot_shape_sensitivity_plot(), janusplot_shape_sensitivity_summary

Examples

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

Index

∗ **datasets**
  shape_sensitivity_demo, 18
∗ **shape-metrics**
  janusplot_shape_cutoffs, 10
  janusplot_shape_hierarchy, 11
  janusplot_shape_metrics, 13
∗ **shape-sensitivity**
  janusplot_shape_sensitivity, 14
  janusplot_shape_sensitivity_plot, 16
  janusplot_shape_sensitivity_shapes, 17
  janusplot_shape_sensitivity_summary, 18
∗ **smooth-associations**
  janusplot, 2
  janusplot_data, 8

ggplot2::ggplot, *7*, *17*

janusplot, 2, *10*
janusplot(), *8*, *10, 11*, *14*
janusplot_data, *7*, 8
janusplot_data(), *7*, *14*
janusplot_shape_cutoffs, 10, *12*, *14*
janusplot_shape_cutoffs(), *5*, *9*, *13–15*
janusplot_shape_hierarchy, *11*, 11, *14*
janusplot_shape_hierarchy(), *5*, *14*, *17*
janusplot_shape_metrics, *11, 12*, 13
janusplot_shape_metrics(), *10*, *14, 15*
janusplot_shape_sensitivity, 14, *17, 18*
janusplot_shape_sensitivity(), *16–19*
janusplot_shape_sensitivity_plot, *16*, 16, *17, 18*
janusplot_shape_sensitivity_plot(), *16*, *19*
janusplot_shape_sensitivity_shapes, *16, 17*, 17, *18*
janusplot_shape_sensitivity_shapes(), *15, 16*
janusplot_shape_sensitivity_summary, *16, 17*, 18
janusplot_shape_sensitivity_summary(), *16*, *19*

mgcv::gam(), *2*, *4*, *7–9*, *13*

patchwork::wrap_plots(), *7*

shape_sensitivity_demo, *14*, *16*, 18