

Package ‘ggextreme’

October 6, 2026

Type Package

Title Bar Chart Races and Interactive Plots for Clinical Research

Version 0.1.0

Description Presentation quality charts built on 'ggplot2' that the package itself does not provide. The bar chart race interpolates values on a uniform time grid, ranks every frame on its own values and eases bars into new positions, so a reordering field stays readable. Frames are ordinary 'ggplot2' objects laid out on a fixed pixel grid, which keeps the axis and label column from drifting between them, and are encoded to GIF or MP4. Circular images, such as the bundled country flags, can be placed at the end of each bar. Causal diagrams are drawn as directed acyclic graphs whose nodes and arrows each carry a rationale and references, shown on hover and opened with clickable links on click, through 'ggiraph'; the same diagram is also available as a static 'ggplot2' object. Network plots for network meta-analysis are drawn from arm level data, with the baseline characteristics and outcomes of every arm shown side by side on click. Forest plots for 'metafor' and 'meta' fits carry each study's record and risk of bias traffic lights, and replay a cumulative meta-analysis as an animation; funnel plots shade where each study would be significant and carry the pooled estimate without it, trim and fill and the tests for small-study effects; league tables for 'netmeta' fits show the direct and indirect evidence behind every estimate. Kaplan-Meier plots read survival and the hazard ratio at any time under the pointer, beside a risk table and proportional hazards tests, and swimmer plots give each patient a lane with their responses, progression and death. Nomograms of regression models, from linear and generalized linear models to mixed, Cox, parametric survival, ordinal and multinomial models, have a handle per predictor and compute each prediction with its confidence interval in the page. Choropleth maps of the world or of any 'sf' map step or play through the years, with several measures side by side for the same year. Causal diagrams can also show which paths between an exposure and an outcome an adjustment set leaves open, by the backdoor criterion; Kaplan-Meier plots give the restricted mean survival time up to a horizon the reader can move; league tables and network plots show where each network estimate's

evidence comes from; and swimmer plots can carry a waterfall of best change and each patient's course beside the lanes. Four explorers put a threshold or an assumption in the reader's hands: the cutoff of a diagnostic test, with what it means for 1,000 people at any prevalence; the strength of unmeasured confounding, with E-values; the choices of a multiverse of analyses; and the threshold that defines a responder. Plots after CINeMA judge the confidence in each estimate of a network meta-analysis in six domains, with every judgment's reason and source, and show what lies behind them: each study's contribution, the estimates against a movable range of little difference, direct against indirect evidence, and a league table and a network marked with the judgments.

License MIT + file LICENSE

Copyright See file inst/COPYRIGHTS for the bundled Lato font, country flag artwork and world map.

Encoding UTF-8

LazyData true

Depends R (≥ 4.1)

Imports gdtools ($\geq 0.5.0$), ggiraph ($\geq 0.9.6$), ggplot2 ($\geq 3.5.0$), grDevices, grid, htmltools, htmlwidgets, igraph, jsonlite, parallel, ragg, rlang, scales, splines, stats, systemfonts ($\geq 1.2.4$), tools, utils

Suggests av, geepack, gifski, glmmTMB, gt, knitr, lme4, logistf, magick, MASS, meta, metadat, metafor, mgcv, netmeta, nlme, nnet, ordinal, quantreg, rmarkdown, rms, rsvg, sf, survival, testthat ($\geq 3.0.0$)

Config/testthat/edition 3

VignetteBuilder knitr

URL <https://choxos.github.io/ggextreme/>,
<https://github.com/choxos/ggextreme>

BugReports <https://github.com/choxos/ggextreme/issues>

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Ahmad Sofi-Mahmudi [aut, cre, cph]

Maintainer Ahmad Sofi-Mahmudi <a.sofimahmudi@gmail.com>

Repository CRAN

Date/Publication 2026-10-06 14:50:02 UTC

Contents

animate_choropleth	3
------------------------------	---

<i>animate_choropleth</i>	3
---------------------------	---

<i>animate_km</i>	4
<i>animate_meta</i>	6
<i>animate_race</i>	7
<i>cinema_clinical</i>	8
<i>cinema_contribution</i>	9
<i>cinema_incoherence</i>	11
<i>cinema_judge</i>	13
<i>cinema_league</i>	16
<i>cinema_network</i>	19
<i>clefts_qci</i>	20
<i>clefts_qci_world</i>	21
<i>cleft_dag</i>	22
<i>ggcausal</i>	23
<i>ggchoropleth</i>	25
<i>ggdiagnostic</i>	27
<i>ggfunnel</i>	28
<i>ggkm</i>	30
<i>ggleague</i>	33
<i>ggmeta</i>	34
<i>ggmultiverse</i>	36
<i>ggnma</i>	38
<i>ggnomogram</i>	40
<i>gggrace</i>	42
<i>ggresponder</i>	45
<i>ggsensitivity</i>	46
<i>ggswimmer</i>	48
<i>graph_widget</i>	50
<i>nomogram_predict</i>	52
<i>psoriasis_nma</i>	52
<i>race_flags</i>	53
<i>race_flag_codes</i>	54
<i>race_frame</i>	55
<i>race_palette</i>	55
<i>race_size</i>	56
<i>theme_race</i>	57

Index	58
--------------	-----------

<i>animate_choropleth</i>	<i>Animate a choropleth map</i>
---------------------------	---------------------------------

Description

Plays the years of a [ggchoropleth\(\)](#) map, easing each region from one year's shade to the next, and writes the result to a GIF or MP4.

Usage

```
animate_choropleth(
  x,
  file = "map.gif",
  step = 0.6,
  end_pause = 2,
  fps = 20,
  res = 150,
  loop = TRUE,
  cores = max(1L, parallel::detectCores() - 1L),
  quiet = FALSE
)
```

Arguments

x	A map from <code>ggchoropleth()</code> .
file	Output path, ending in .gif or .mp4.
step	Seconds from one time to the next.
end_pause	Seconds to hold the final frame.
fps	Frames per second.
res	Output resolution in pixels per inch.
loop	Loop the GIF. Ignored for video.
cores	Number of cores to draw frames on.
quiet	Suppress the progress bar.

Value

file, invisibly.

Examples

```
m <- ggchoropleth(subset(clefts_qci_world, year >= 2015), iso3, year,
  values = c(QCI = "qci"))
animate_choropleth(m, tempfile(fileext = ".gif"), step = 0.3, fps = 6,
  res = 60, cores = 1)
```

 animate_km

Animate a Kaplan-Meier plot

Description

Draws the curves of a `ggkm()` plot over follow-up, as though the trial were being watched, and writes the result to a GIF or MP4. The numbers at risk appear as each break is reached and the time is shown large behind the curves.

Usage

```
animate_km(  
  x,  
  file = "km.gif",  
  duration = 5,  
  end_pause = 2,  
  fps = 30,  
  res = 150,  
  loop = TRUE,  
  cores = max(1L, parallel::detectCores() - 1L),  
  quiet = FALSE  
)
```

Arguments

x	A plot from <code>ggkm()</code> .
file	Output path, ending in .gif or .mp4.
duration	Seconds to draw the full follow-up.
end_pause	Seconds to hold the final frame.
fps	Frames per second.
res	Output resolution in pixels per inch.
loop	Loop the GIF. Ignored for video.
cores	Number of cores to draw frames on.
quiet	Suppress the progress bar.

Value

file, invisibly.

Examples

```
if (requireNamespace("survival", quietly = TRUE)) {  
  colon <- subset(survival::colon, etype == 2)  
  km <- ggkm(survival::Surv(time / 365.25, status) ~ rx, data = colon,  
             xlab = "Years")  
  animate_km(km, tempfile(fileext = ".gif"), duration = 2, fps = 8,  
             cores = 1)  
}
```

animate_meta

*Animate a cumulative meta-analysis***Description**

Replays a meta-analysis one study at a time, in the order of the data, and writes it to a GIF or MP4. Each study fades in as it is added, and the pooled diamond eases to its new position over swap seconds, the same motion as `ggrace()`, before holding for hold seconds.

Usage

```
animate_meta(
  x,
  file = "meta.gif",
  time = NULL,
  hold = 0.8,
  swap = 0.45,
  end_pause = 2,
  fps = 30,
  res = 150,
  loop = TRUE,
  cores = max(1L, parallel::detectCores() - 1L),
  quiet = FALSE
)
```

Arguments

x	A forest plot from <code>ggmeta()</code> .
file	Output path, ending in .gif or .mp4.
time	Optional labels, one per study, shown large behind the plot as each study is added, such as the year of publication.
hold	Seconds to hold on each step.
swap	Seconds the pooled estimate takes to move to its new value.
end_pause	Seconds to hold the final frame.
fps	Frames per second.
res	Output resolution in pixels per inch.
loop	Loop the GIF. Ignored for video.
cores	Number of cores to draw frames on.
quiet	Suppress the progress bar.

Value

file, invisibly.

Examples

```
if (requireNamespace("metafor", quietly = TRUE)) {
  dat <- metafor::escalc(measure = "RR", ai = tpos, bi = tneg,
                        ci = cpos, di = cneg, data = metadat::dat.bcg,
                        slab = paste(author, year))
  dat <- dat[order(dat$year), ]
  fit <- metafor::rma(yi, vi, data = dat)
  animate_meta(ggmeta(fit), tempfile(fileext = ".gif"), time = dat$year,
              fps = 10, cores = 1)
}
```

animate_race	<i>Render a bar chart race to a file</i>
--------------	--

Description

Draws every frame with 'ragg' and encodes them. The encoder follows the file extension: .gif uses 'gifski' and falls back to 'magick', anything else is treated as video and uses 'av', falling back to an ffmpeg binary on the search path.

Usage

```
animate_race(
  x,
  file = "race.mp4",
  loop = TRUE,
  cores = max(1L, parallel::detectCores() - 1L),
  quiet = FALSE
)
```

Arguments

x	A ggrace object from ggrace() .
file	Output path, ending in .gif or .mp4.
loop	Loop the GIF. Ignored for video.
cores	Number of cores to draw frames on. Frames are independent, so this scales close to linearly. Forced to 1 on Windows, where parallel::mclapply() cannot fork.
quiet	Suppress the progress bar. Progress is not reported when drawing on more than one core.

Value

file, invisibly.

cinema_clinical

*Draw network estimates against a range of little difference***Description**

Draws every network estimate of a network meta-analysis with its confidence interval and prediction interval against a range of little difference, whose lower and upper limits the reader can move on their own. Beside each estimate, in words, is what its confidence interval and its prediction interval are compatible with: an important benefit, little difference or an important harm of the first treatment against the second, or more than one of these; and the imprecision and heterogeneity judgments that CINeMA's rules give at these limits.

Usage

```
cinema_clinical(
  x,
  ...,
  reference = NULL,
  prediction = TRUE,
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

x	Judgments from <code>cinema_judge()</code> that include a threshold, or a network meta-analysis from <code>netmeta::netmeta()</code> , judged with the arguments in ...
...	When x is a netmeta fit, arguments for <code>cinema_judge()</code> : threshold, the pre-specified limits of little difference, which is required, and <code>small_values</code> , order and pooled.
reference	Optionally, one treatment: only its comparisons with the others are drawn, such as every treatment against placebo.
prediction	Draw the prediction intervals, when the model has them.
title, caption	Title above the plot and note below it. A caption you give is added above the default notes.
family	Font family. The package ships Lato and registers it on load.

Details

In the widget, two sliders move the lower and the upper limit; the prespecified limits stay marked with dashed lines and a button returns to them. Every reading, judgment and the summary under the plot follow. Under the estimates, a threshold sensitivity strip for each limit shows, row by row, where each reading changes as that limit moves with the other held where it is: a change of color is a change of reading. Clicking or tapping a strip moves that limit there; the sliders do the same from the keyboard. A switch hides the prediction intervals.

The limits apply to every comparison as written, the first treatment against the second, in the order set by `cinema_judge()`. A prediction interval shows where the effect in a comparable new setting is expected to lie, not the effect for an individual patient, and with few studies the heterogeneity behind it is poorly estimated. The plot describes what each interval is compatible with; it never calls a comparison equivalent.

Value

An object of class `cinema_clinical`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `readings` gives, for each comparison drawn, what its intervals are compatible with and the imprecision and heterogeneity judgments at the prespecified limits.

Sources

Nikolakopoulou A, Higgins JPT, Papakonstantinou T, et al. CINeMA: an approach for assessing confidence in the results of a network meta-analysis. *PLoS Medicine* 2020;17(4):e1003082. doi:10.1371/journal.pmed.1003082

Papakonstantinou T, Nikolakopoulou A, Higgins JPT, Egger M, Salanti G. CINeMA: software for semiautomated assessment of the confidence in the results of network meta-analysis. *Campbell Systematic Reviews* 2020;16:e1080. doi:10.1002/cl2.1080

Papakonstantinou T, Nikolakopoulou A, Rucker G, et al. Estimating the contribution of studies in network meta-analysis: paths, flows and streams. *F1000Research* 2018;7:610.

Examples

```
if (requireNamespace("netmeta", quietly = TRUE) &&
    requireNamespace("meta", quietly = TRUE)) {
  pw <- meta::pairwise(treat = treatment, event = pasi75_r,
                      n = pasi75_n, studlab = study,
                      data = psoriasis_nma, sm = "OR")
  nma <- netmeta::netmeta(pw, common = FALSE)
  cinema_clinical(nma, threshold = c(0.8, 1.25),
                 small_values = "undesirable")
}
```

<code>cinema_contribution</code>	<i>Draw where each network estimate's evidence comes from, study by study</i>
----------------------------------	---

Description

Draws the contribution of every study to every estimate of a network meta-analysis, from `netmeta::netcontrib(x, study = TRUE)`: one bar per comparison, split into studies as wide as their share of the estimate and colored by each study's risk of bias or indirectness, with the studies grouped low, moderate and high as CINeMA draws them; a comparison by study matrix of the same numbers, whose squares mark with an outline the studies that compare that pair head to head; and a small network.

Usage

```
cinema_contribution(
  x,
  ...,
  color = NULL,
  positions = NULL,
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

<code>x</code>	Judgments from <code>cinema_judge()</code> made with study judgments, or a network meta-analysis from <code>netmeta::netmeta()</code> , judged with the arguments in
<code>...</code>	When <code>x</code> is a <code>netmeta</code> fit, arguments for <code>cinema_judge()</code> : the study judgments <code>rob</code> and <code>indirectness</code> , at least one of them, and optionally <code>contributions</code> , <code>small_values</code> , <code>order</code> and <code>pooled</code> .
<code>color</code>	Which judgment colors the studies at first, " <code>rob</code> " or " <code>indirectness</code> ". Defaults to risk of bias when it was given.
<code>positions</code>	Optional data frame placing the treatments of the small network by hand, with columns <code>treatment</code> , <code>x</code> and <code>y</code> and <code>y</code> pointing up, as in <code>ggmma()</code> . Defaults to a circle.
<code>title, caption</code>	Title above the plot and note below it. A caption you give is added above the default notes.
<code>family</code>	Font family. The package ships Lato and registers it on load.

Details

In the widget, a switch colors the studies by risk of bias or by indirectness. Selecting a comparison, from its label, the matrix or the menu, lights its bar and matrix row and widens each line of the small network by the share of the estimate that flows through it. Selecting a study, from the matrix, its button or the panel, dims the others and marks the comparisons it randomized. Clicking a part of a bar lists the studies with that judgment in that estimate, with their shares and the reasons for their judgments. The labels and matrix headers can be reached with the keyboard, and a sentence under the controls says what the selection shows.

Contributions say where an estimate's information comes from, not how trustworthy it is. Dropping a study would change the whole fit, so there is deliberately no switch that removes one and rescales the bars.

Value

An object of class `cinema_contribution`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `contributions` holds the share of each estimate from each study.

Sources

Nikolakopoulou A, Higgins JPT, Papakonstantinou T, et al. CINeMA: an approach for assessing confidence in the results of a network meta-analysis. PLoS Medicine 2020;17(4):e1003082. doi:10.1371/journal.pmed.1003082

Papakonstantinou T, Nikolakopoulou A, Higgins JPT, Egger M, Salanti G. CINeMA: software for semiautomated assessment of the confidence in the results of network meta-analysis. Campbell Systematic Reviews 2020;16:e1080. doi:10.1002/cl2.1080

Papakonstantinou T, Nikolakopoulou A, Rucker G, et al. Estimating the contribution of studies in network meta-analysis: paths, flows and streams. F1000Research 2018;7:610.

Examples

```
if (requireNamespace("netmeta", quietly = TRUE) &&
    requireNamespace("meta", quietly = TRUE)) {
  pw <- meta::pairwise(treat = treatment, event = pasi75_r,
                      n = pasi75_n, studlab = study,
                      data = psoriasis_nma, sm = "OR")
  nma <- netmeta::netmeta(pw, common = FALSE)
  # Illustrative judgments, invented for this example.
  rob <- data.frame(
    study = c("CLEAR", "ERASURE", "FEATURE", "FIXTURE", "JUNCTURE"),
    judgment = c("high", "low", "some concerns", "low", "some concerns")
  )
  cinema_contribution(nma, rob = rob, small_values = "undesirable",
                     caption = "Illustrative judgments, not published assessments.")
}
```

cinema_incoherence

Draw direct, indirect and network estimates side by side

Description

Draws, for every comparison of a network meta-analysis, the direct estimate from the studies that compare the pair head to head, the indirect estimate from the rest of the network and the network estimate that combines them, as separated by `netmeta::netsplit()`. Beside them is the inconsistency factor, the ratio of the direct to the indirect estimate for a ratio measure or their difference otherwise, with its confidence interval and p-value, and the incoherence judgment that CINeMA's rules give.

Usage

```
cinema_incoherence(x, ..., title = NULL, caption = NULL, family = "Lato")
```

Arguments

<code>x</code>	Judgments from <code>cinema_judge()</code> , or a network meta-analysis from <code>netmeta::netmeta()</code> , judged with the arguments in
<code>...</code>	When <code>x</code> is a <code>netmeta</code> fit, arguments for <code>cinema_judge()</code> , such as <code>threshold</code> , which shades the range of little difference and lets the rule judge comparisons whose test gives <code>p</code> of 0.10 or less, <code>split</code> , <code>small_values</code> , <code>order</code> and <code>contributions</code> , which list the studies behind each indirect estimate.
<code>title, caption</code>	Title above the plot and note below it. A caption you give is added above the default notes.
<code>family</code>	Font family. The package ships Lato and registers it on load.

Details

A comparison with only direct or only indirect evidence cannot be checked this way. It is kept visibly apart, marked as not assessable locally, because the absence of a test is not agreement; CINeMA then judges it from the global design by treatment interaction test, given under the plot. Both tests have low power, above all with few studies, so a large `p`-value is weak evidence that direct and indirect evidence agree, and the interval of the inconsistency factor shows how large a disagreement the data still allow.

Hovering over a comparison gives its numbers; clicking it, or pressing Enter on it, opens a panel that says in words how far direct and indirect evidence could disagree, lists the studies behind each estimate and gives the reason for the judgment.

Value

An object of class `cinema_incoherence`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `comparisons` holds the direct, indirect and network estimates and the inconsistency factors on the scale of the effect, and `global` the design by treatment test.

Sources

Nikolakopoulou A, Higgins JPT, Papakonstantinou T, et al. CINeMA: an approach for assessing confidence in the results of a network meta-analysis. *PLoS Medicine* 2020;17(4):e1003082. doi:10.1371/journal.pmed.1003082

Papakonstantinou T, Nikolakopoulou A, Higgins JPT, Egger M, Salanti G. CINeMA: software for semiautomated assessment of the confidence in the results of network meta-analysis. *Campbell Systematic Reviews* 2020;16:e1080. doi:10.1002/cl2.1080

Papakonstantinou T, Nikolakopoulou A, Rucker G, et al. Estimating the contribution of studies in network meta-analysis: paths, flows and streams. *F1000Research* 2018;7:610.

Examples

```
if (requireNamespace("netmeta", quietly = TRUE) &&
    requireNamespace("meta", quietly = TRUE)) {
  pw <- meta::pairwise(treat = treatment, event = pasi75_r,
                      n = pasi75_n, studlab = study,
```

```

      data = psoriasis_nma, sm = "OR")
nma <- netmeta::netmeta(pw, common = FALSE)
cinema_incoherence(nma, threshold = 1.25, small_values = "undesirable")
}

```

cinema_judge

Judge confidence in the results of a network meta-analysis

Description

Applies the rules of CINeMA, Confidence in Network Meta-Analysis (Nikolakopoulou et al. 2020; Papakonstantinou et al. 2020), to every comparison of a network meta-analysis. Each comparison gets a judgment of no concerns, some concerns or major concerns in each of six domains: within-study bias, reporting bias, indirectness, imprecision, heterogeneity and incoherence, each with its reason in words and a note of whether it was computed by a rule or given by you. The domains are kept side by side and never added into a score.

Usage

```

cinema_judge(
  x,
  rob = NULL,
  indirectness = NULL,
  reporting = NULL,
  threshold = NULL,
  rule = "average",
  judgments = NULL,
  small_values = NULL,
  order = NULL,
  pooled = NULL,
  contributions = NULL,
  split = NULL
)

```

Arguments

x A network meta-analysis from `netmeta::netmeta()`.

rob, indirectness Study level judgments of risk of bias and of indirectness: data frames with a column study, naming every study in the network once, a column judgment, and optionally a column reason in words. A judgment is low, moderate or high, written as "low", "some concerns" (or "moderate" or "unclear") and "high", as "l", "m" and "h", or as 1, 2 and 3, the codes the CINeMA web application reads.

reporting	Your judgment of reporting bias, which cannot be computed from the data: a data frame with a column judgment, "undetected" or "suspected" ("strongly suspected" is also read), an optional column reason, and the comparison it applies to, either as treat1 and treat2 or as comparison (such as "A:B" or "A vs B"). A data frame with a single row and no comparison applies to every comparison.
threshold	The limits of the range of little difference, on the scale of the effect (an odds ratio, say, not its logarithm), for the first treatment of each comparison against the second: one number, such as 1.25, for a range symmetric about no effect (0.8 to 1.25 here), or two numbers for independent lower and upper limits, which must lie on either side of no effect. A threshold of no effect itself, 1 for a ratio or 0 for a difference, treats any effect as important. Imprecision and heterogeneity, and incoherence when its test gives p of 0.10 or less, need it.
rule	How the study judgments are summarized for each comparison: "average", "majority" or "highest". Give two, such as c("average", "highest"), for different rules for within-study bias and for indirectness. See the section on rules.
judgments	Domain judgments you made yourself, such as those exported from the CINeMA web application, which replace the computed ones. Either wide, with the comparison (treat1 and treat2, or comparison) and one column per domain, named like the domains ("Within-study bias", "Reporting bias", "Indirectness", "Imprecision", "Heterogeneity", "Incoherence"), optional columns such as "Imprecision reason" and optional "Confidence rating" and "Reason(s) for downgrading" columns; or long, with the comparison and columns domain, judgment and optionally reason. A missing or empty judgment leaves the computed one in place.
small_values	Whether small values of the effect are "desirable", as for mortality, or "undesirable", as for a response. It sets the ranking, and so the order of the treatments, and which side of the range is a benefit. Defaults to the setting stored in x, which is worth checking.
order	The order of the treatments. Each comparison is written with the treatment that comes first in this order first, and the limits in threshold apply in that direction. Defaults to the P-score ranking, best first.
pooled	Which model to use, "random" or "common". Defaults to the random effects model when x has one.
contributions	The contribution of each study to each estimate: an object from <code>netmeta::netcontrib(x, study = TRUE)</code> , or TRUE to compute it. By default it is computed when rob or indirectness is given, which takes a few seconds for a network of a few dozen studies.
split	The direct and indirect estimates: an object from <code>netmeta::netsplit()</code> , or NULL to compute them with its default method, back-calculation. Give one computed with <code>method = "SIDDE"</code> to use that method instead.

Details

The result feeds the plots of the family, so they share one set of estimates, contributions and judgments: `cinema_contribution()`, `cinema_clinical()`, `cinema_incoherence()` and `cinema_league()`.

Each of them also accepts a netmeta fit and the arguments of this function.

Value

An object of class `cinema`, a list whose main fields are judgments, one row per comparison and domain with the level (0, 1 or 2 for no, some or major concerns, NA when not judged), the judgment in words, the reason and its source; comparisons, the network, direct and indirect estimates, prediction intervals and inconsistency factors on the scale of the effect; contributions, the share of each estimate from each study; studies, the study judgments; threshold, rule, global (the design by treatment test) and fit. It prints as a table of the judgments.

Rules

Every rule below is the one CINeMA implements, as published; where the papers leave a detail open the choice made here is stated.

- **Within-study bias** and **indirectness** combine the study judgments with the percentage contribution of each study to each estimate (Papakonstantinou et al. 2018), from `netmeta::netcontrib()`. The *majority* rule takes the level with the largest total contribution, the more serious level on a tie; the *average* rule scores low 1, moderate 2 and high 3, averages the scores weighted by contribution and rounds, halves up; the *highest* rule takes the most serious level among the studies that contribute more than 0.0001 percent. Low, moderate and high become no, some and major concerns.
- **Reporting bias** is your judgment; CINeMA suggests suspected or undetected, and suspected is shown as some concerns.
- **Imprecision** compares the confidence interval with the range of little difference. There are no concerns when the interval lies wholly within the range, or wholly on the side of no effect that the point estimate is on; some concerns when it crosses no effect but not the limit on the other side; and major concerns when it passes that limit, so that it holds important effects in both directions.
- **Heterogeneity** judges the prediction interval by the same rule. There are no concerns when it reaches the same step as the confidence interval, some concerns when it reaches one step further and major concerns when it reaches two (Table 4 of Papakonstantinou et al. 2020; this reproduces every scenario in Figure 3 of Nikolakopoulou et al. 2020). A common effect model has no prediction interval, so heterogeneity is then not judged.
- **Incoherence**, for a comparison with direct and indirect evidence, uses the test of the difference between them from `netmeta::netsplit()` (SIDE). With p above 0.10 there are no concerns. Otherwise the areas below, within and above the range of little difference are compared: when both confidence intervals reach the same areas there are no concerns, when they differ in one area some concerns, and when they differ in two or three major concerns. A comparison with only direct or only indirect evidence cannot be tested locally, and takes its judgment from the global design by treatment interaction test of `netmeta::decomp.design()`: major concerns below 0.05, some from 0.05 to 0.10 and no concerns above; when the network has no closed loop, so the test cannot be computed, major concerns. Both tests have low power.

CINeMA's authors stress that these rules are a starting point: the reasons say what each rule saw, so a judgment can be revised by giving it in judgments. CINeMA also leaves any overall rating to the reviewers; this function gives none, though a rating you supply is kept and shown.

Sources

Nikolakopoulou A, Higgins JPT, Papakonstantinou T, et al. CINeMA: an approach for assessing confidence in the results of a network meta-analysis. PLoS Medicine 2020;17(4):e1003082. doi:10.1371/journal.pmed.1003082

Papakonstantinou T, Nikolakopoulou A, Higgins JPT, Egger M, Salanti G. CINeMA: software for semiautomated assessment of the confidence in the results of network meta-analysis. Campbell Systematic Reviews 2020;16:e1080. doi:10.1002/cl2.1080

Papakonstantinou T, Nikolakopoulou A, Rucker G, et al. Estimating the contribution of studies in network meta-analysis: paths, flows and streams. F1000Research 2018;7:610.

Examples

```
if (requireNamespace("netmeta", quietly = TRUE) &&
    requireNamespace("meta", quietly = TRUE)) {
  pw <- meta::pairwise(treat = treatment, event = pasi75_r,
                      n = pasi75_n, studlab = study,
                      data = psoriasis_nma, sm = "OR")
  nma <- netmeta::netmeta(pw, common = FALSE)
  # Illustrative study judgments, invented for this example: they are
  # not published assessments of these trials.
  rob <- data.frame(
    study = c("CLEAR", "ERASURE", "FEATURE", "FIXTURE", "JUNCTURE"),
    judgment = c("high", "low", "some concerns", "low", "some concerns")
  )
  j <- cinema_judge(nma, rob = rob, threshold = 1.25,
                   small_values = "undesirable",
                   reporting = data.frame(judgment = "undetected"))
  j
  head(j$judgments)
}
```

cinema_league

Draw a league table with a CINeMA confidence profile in every cell

Description

Draws every pairwise estimate of a network meta-analysis as a grid laid out like `ggleague()` and `netmeta::netleague()`: each cell compares the treatment that comes first on the diagonal with the one that comes later, network estimates sit below the diagonal and direct estimates above it. Under each network estimate six small marks give its judgment in each CINeMA domain, left to right within-study bias, reporting bias, indirectness, imprecision, heterogeneity and incoherence, colored by no, some or major concerns. The marks are never added into a score.

Usage

```
cinema_league(x, ..., title = NULL, caption = NULL, family = "Lato")
```

Arguments

<code>x</code>	Judgments from <code>cinema_judge()</code> , or a network meta-analysis from <code>netmeta::netmeta()</code> , which is then judged with the arguments in
<code>...</code>	When <code>x</code> is a <code>netmeta</code> fit, arguments for <code>cinema_judge()</code> : the study judgments <code>rob</code> and <code>indirectness</code> , <code>reporting</code> , <code>threshold</code> , <code>rule</code> , your own judgments, <code>small_values</code> and <code>order</code> . Giving only judgments, such as a report exported from the CINeMA web application, draws your judgments as they are.
<code>title, caption</code>	Title above the table and note below it. The default caption says which estimates sit on each side of the diagonal and, domain by domain, which judgments were computed by which rule and which are yours. A caption you give is added above it.
<code>family</code>	Font family. The package ships Lato and registers it on load.

Details

Hovering over a cell lists its judgments; clicking it opens a panel under the table with every domain's judgment, the reason for it and whether it was computed by a rule or given by you, the estimate against the range of little difference, and a bar of the contribution of each study to the estimate, colored by its risk of bias. The cells can also be reached with the keyboard: Tab moves between them and Enter or Space opens one. Hovering over a treatment on the diagonal lights its row and column.

A mark drawn hollow is a domain not judged, such as reporting bias when you gave no judgment for it. A round mark in the incoherence place means there was no local test for that comparison, only the global test of the whole network, as CINeMA prescribes.

Value

An object of class `cinema_league`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `judgments` holds the result of `cinema_judge()` the table was drawn from.

Rules

Every rule below is the one CINeMA implements, as published; where the papers leave a detail open the choice made here is stated.

- **Within-study bias** and **indirectness** combine the study judgments with the percentage contribution of each study to each estimate (Papakonstantinou et al. 2018), from `netmeta::netcontrib()`. The *majority* rule takes the level with the largest total contribution, the more serious level on a tie; the *average* rule scores low 1, moderate 2 and high 3, averages the scores weighted by contribution and rounds, halves up; the *highest* rule takes the most serious level among the studies that contribute more than 0.0001 percent. Low, moderate and high become no, some and major concerns.
- **Reporting bias** is your judgment; CINeMA suggests suspected or undetected, and suspected is shown as some concerns.
- **Imprecision** compares the confidence interval with the range of little difference. There are no concerns when the interval lies wholly within the range, or wholly on the side of no effect that

the point estimate is on; some concerns when it crosses no effect but not the limit on the other side; and major concerns when it passes that limit, so that it holds important effects in both directions.

- **Heterogeneity** judges the prediction interval by the same rule. There are no concerns when it reaches the same step as the confidence interval, some concerns when it reaches one step further and major concerns when it reaches two (Table 4 of Papakonstantinou et al. 2020; this reproduces every scenario in Figure 3 of Nikolakopoulou et al. 2020). A common effect model has no prediction interval, so heterogeneity is then not judged.
- **Incoherence**, for a comparison with direct and indirect evidence, uses the test of the difference between them from `netmeta::netsplit()` (SIDE). With p above 0.10 there are no concerns. Otherwise the areas below, within and above the range of little difference are compared: when both confidence intervals reach the same areas there are no concerns, when they differ in one area some concerns, and when they differ in two or three major concerns. A comparison with only direct or only indirect evidence cannot be tested locally, and takes its judgment from the global design by treatment interaction test of `netmeta::decomp.design()`: major concerns below 0.05, some from 0.05 to 0.10 and no concerns above; when the network has no closed loop, so the test cannot be computed, major concerns. Both tests have low power.

CINeMA's authors stress that these rules are a starting point: the reasons say what each rule saw, so a judgment can be revised by giving it in judgments. CINeMA also leaves any overall rating to the reviewers; this function gives none, though a rating you supply is kept and shown.

Sources

Nikolakopoulou A, Higgins JPT, Papakonstantinou T, et al. CINeMA: an approach for assessing confidence in the results of a network meta-analysis. *PLoS Medicine* 2020;17(4):e1003082. doi:10.1371/journal.pmed.1003082

Papakonstantinou T, Nikolakopoulou A, Higgins JPT, Egger M, Salanti G. CINeMA: software for semiautomated assessment of the confidence in the results of network meta-analysis. *Campbell Systematic Reviews* 2020;16:e1080. doi:10.1002/cl2.1080

Papakonstantinou T, Nikolakopoulou A, Rucker G, et al. Estimating the contribution of studies in network meta-analysis: paths, flows and streams. *F1000Research* 2018;7:610.

Examples

```
if (requireNamespace("netmeta", quietly = TRUE) &&
    requireNamespace("meta", quietly = TRUE)) {
  pw <- meta::pairwise(treat = treatment, event = pasi75_r,
                      n = pasi75_n, studlab = study,
                      data = psoriasis_nma, sm = "OR")
  nma <- netmeta::netmeta(pw, common = FALSE)
  # Illustrative judgments, invented for this example.
  rob <- data.frame(
    study = c("CLEAR", "ERASURE", "FEATURE", "FIXTURE", "JUNCTURE"),
    judgment = c("high", "low", "some concerns", "low", "some concerns")
  )
  cinema_league(nma, rob = rob, threshold = 1.25,
                small_values = "undesirable",
                caption = "Illustrative judgments, not published assessments.")
}
```

```
}
```

cinema_network	<i>Draw a network plot with a strand for every study, colored by its judgment</i>
----------------	---

Description

Draws the network of a network meta-analysis from arm level data, with every line drawn as one strand per study, so a comparison's width counts its studies and its colors show each study's risk of bias or indirectness instead of an average that would hide a mix of judgments. Node area follows the participants randomized to each treatment when n is given.

Usage

```
cinema_network(  
  data,  
  study,  
  treatment,  
  n = NULL,  
  rob = NULL,  
  indirectness = NULL,  
  color = NULL,  
  positions = NULL,  
  title = NULL,  
  caption = NULL,  
  family = "Lato"  
)
```

Arguments

data	A data frame with one row per study arm.
study, treatment	Bare column names identifying the study and the treatment of each arm.
n	Optional bare column with the number of participants in each arm; it sets node area and the counts in the panels.
rob, indirectness	Study level judgments of risk of bias and of indirectness, at least one of them: data frames with columns study, judgment and optionally reason, as in cinema_judge() , or the studies field of its result.
color	Which judgment colors the strands at first, "rob" or "indirectness". Defaults to risk of bias when it was given.
positions	Optional data frame placing the treatments by hand, with columns treatment, x and y and y pointing up, as in ggmma() . Defaults to a circle.
title, caption	Title above the plot and note below it. A caption you give is added above the default notes.
family	Font family. The package ships Lato and registers it on load.

Details

In the widget, a switch colors the strands by risk of bias or by indirectness, and a sentence under the plot counts the comparisons that rest only on studies with the most serious judgment, those that mix judgments and those that rest only on studies without concerns. Hovering over a strand names its study and judgments. Clicking a line or a treatment, or pressing Enter on it, opens a panel that lists the studies behind it with their judgments and the reasons for them.

Value

An object of class `cinema_network`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `edges` lists each comparison with its studies and how many of them have each judgment.

Sources

Nikolakopoulou A, Higgins JPT, Papakonstantinou T, et al. CINeMA: an approach for assessing confidence in the results of a network meta-analysis. *PLoS Medicine* 2020;17(4):e1003082. [doi:10.1371/journal.pmed.1003082](https://doi.org/10.1371/journal.pmed.1003082)

Papakonstantinou T, Nikolakopoulou A, Higgins JPT, Egger M, Salanti G. CINeMA: software for semiautomated assessment of the confidence in the results of network meta-analysis. *Campbell Systematic Reviews* 2020;16:e1080. [doi:10.1002/cl2.1080](https://doi.org/10.1002/cl2.1080)

Papakonstantinou T, Nikolakopoulou A, Rucker G, et al. Estimating the contribution of studies in network meta-analysis: paths, flows and streams. *F1000Research* 2018;7:610.

Examples

```
# Illustrative judgments, invented for this example.
rob <- data.frame(
  study = c("CLEAR", "ERASURE", "FEATURE", "FIXTURE", "JUNCTURE"),
  judgment = c("high", "low", "some concerns", "low", "some concerns"),
  reason = "Illustrative only."
)
cinema_network(psoriasis_nma, study, treatment, n = n, rob = rob,
  caption = "Illustrative judgments, not published assessments.")
```

clefts_qci

Quality of Care Index for orofacial clefts, 1990 to 2019

Description

Yearly Quality of Care Index (QCI) scores for orofacial clefts in fifteen countries. The QCI is a composite of four secondary indices derived from Global Burden of Disease estimates and summarized by principal component analysis, rescaled to run from 0 to 100, where higher is better care.

Usage

```
clefts_qci
```

Format

A data frame with 450 rows and 5 columns:

- country** Country name.
- iso** ISO 3166-1 alpha-2 country code, lower case.
- region** World region, a factor with five levels.
- year** Year, 1990 to 2019.
- qci** Quality of Care Index, 0 to 100.

Details

The published analysis covers every country; the fifteen here were chosen to span continents, to cover a wide range of scores and to include several changes of rank over the period, which makes the set a useful example for `ggrace()`. Country names are shortened for plotting, and `iso` matches the codes `race_flags()` uses.

Source

Sofi-Mahmudi A, Shamsoddin E, Khademioore S, Khazaei Y, Vahdati A, Tovani-Palone MR (2025). Global, regional, and national survey on burden and Quality of Care Index (QCI) of orofacial clefts: Global burden of disease systematic analysis 1990-2019. PLOS ONE 20(1): e0317267. [doi:10.1371/journal.pone.0317267](https://doi.org/10.1371/journal.pone.0317267)

Examples

```
head(clefts_qci)
subset(clefts_qci, year == 2019)[order(-subset(clefts_qci, year == 2019)$qci), ]
```

clefts_qci_world	<i>Quality of Care Index for orofacial clefts in every country, 1990 to 2019</i>
------------------	--

Description

Yearly Quality of Care Index (QCI) scores for orofacial clefts in 195 countries and territories, the full country panel of the analysis that `clefts_qci` samples. The QCI is a composite of four secondary indices derived from Global Burden of Disease estimates and summarized by principal component analysis, rescaled to run from 0 to 100, where higher is better care. World regions, WHO regions, World Bank income groups and SDI groups in the published panel are left out.

Usage

```
clefts_qci_world
```

Format

A data frame with 5,850 rows and 4 columns:

country Country or territory, as named by the Global Burden of Disease study.

iso3 ISO 3166-1 alpha-3 code.

year Year, 1990 to 2019.

qci Quality of Care Index, 0 to 100.

Details

Countries keep the names the Global Burden of Disease study gives them, such as "Iran (Islamic Republic of)", and carry the ISO 3166-1 alpha-3 code of the map `ggchoropleth()` draws, so either column can name the region.

Source

Sofi-Mahmudi A, Shamsoddin E, Khademioore S, Khazaei Y, Vahdati A, Tovani-Palone MR (2025). Global, regional, and national survey on burden and Quality of Care Index (QCI) of orofacial clefts: Global burden of disease systematic analysis 1990-2019. PLOS ONE 20(1): e0317267. [doi:10.1371/journal.pone.0317267](https://doi.org/10.1371/journal.pone.0317267)

Examples

```
head(clefts_qci_world)
ggchoropleth(clefts_qci_world, iso3, year, values = c(QCI = "qci"))
```

cleft_dag	<i>An illustrative causal diagram for maternal smoking and orofacial clefts</i>
-----------	---

Description

A small directed acyclic graph for a study of maternal smoking and orofacial clefts in the child, with a rationale for every node and arrow and published references for several of them. It shows each kind of role `ggcausal()` colors: an exposure, an outcome, a confounder, an unobserved cause of the outcome, a collider created by counting only live births, and a free text role. The structure is a teaching example rather than the result of a formal review.

Usage

```
cleft_dag
```

Format

A list of two data frames.

nodes Six nodes with columns name, label, role, rationale, references and timing. timing is not a reserved column, so it appears as a field in the hover card and panel.

edges Eight arrows with columns from, to, rationale and references.

Source

Rationales written for this package. References were checked against PubMed and are listed in full in the references columns.

Examples

```
cleft_dag$nodes[c("name", "role")]
ggcausal(cleft_dag$edges, cleft_dag$nodes)
```

ggcausal

Draw an interactive causal diagram

Description

Draws a directed acyclic graph (DAG) in which every node and every arrow carries its own justification. Hovering over a node shows its role and the reason it is in the diagram; hovering over an arrow shows the assumed direction of effect and the reason for it. Clicking either opens a panel under the diagram with the full rationale and clickable references, which also makes the diagram usable on touch screens.

Usage

```
ggcausal(
  edges,
  nodes = NULL,
  direction = c("right", "down"),
  palette = NULL,
  legend = TRUE,
  legend_title = NULL,
  title = NULL,
  caption = NULL,
  family = "Lato",
  paths = FALSE,
  exposure = NULL,
  outcome = NULL,
  adjust = character(0)
)
```

Arguments

edges	A data frame with one row per arrow and columns from and to naming its ends. Optional columns: rationale, the reason for the arrow and its direction, and references. Any other column is shown as a labeled field.
nodes	A data frame with one row per node and a name column matching from and to. Optional columns: label, the text in the box, which may contain line breaks and defaults to name; role; rationale; references; and x and y to place the box by hand, in grid units with y pointing up. Any other column is shown as a labeled field. When NULL, the nodes are taken from edges.

<code>direction</code>	Which way the arrows point: "right" or "down".
<code>palette</code>	Colors for the roles, as a vector named by role. Roles left out keep their default color.
<code>legend</code>	Draw a legend of the roles above the diagram.
<code>legend_title</code>	Text in front of the legend, such as "Role".
<code>title, caption</code>	Title above the diagram and note below it.
<code>family</code>	Font family. The package ships Lato and registers it on load.
<code>paths</code>	Show which paths between the exposure and the outcome are open or blocked for an adjustment set. The widget then lets the reader choose the exposure and outcome, click variables to adjust for them, and read why each path is open or blocked, whether the set is sufficient and which minimal sets would be.
<code>exposure, outcome</code>	Names of the exposure and the outcome. Default to the nodes whose role is exposure and outcome.
<code>adjust</code>	Names of the nodes adjusted for at the start.

Details

The layout is layered, so every arrow points the same way, left to right or top down. An arrow that skips a layer is routed around the boxes in between on a smooth curve. Give x and y columns in nodes to place the boxes yourself; arrows are then drawn straight.

Nodes are colored by role. The roles exposure, outcome, confounder, mediator, collider, instrument, unobserved and latent have fixed colors, and the last two are drawn with a dashed border. Any other role is allowed and takes the next color from `race_palette()`.

references may be a list column with one character vector per row, or text with several references separated by | or a line break. A reference that contains a URL is linked to it; otherwise one that contains a DOI is linked to <https://doi.org/>.

Value

An object of class `ggcausal`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The fields `width` and `height` give the natural size in inches.

Adjustment paths

A path between the exposure and the outcome is causal when every arrow on it points from the exposure toward the outcome, and biasing otherwise. A path is blocked when it passes through a variable that is adjusted for, unless that variable is a collider, where two arrows meet head to head; a collider blocks a path until it, or one of its descendants, is adjusted for, which opens it. An adjustment set is sufficient by the backdoor criterion when it blocks every biasing path and contains no descendant of the exposure (Pearl 2009). Unobserved and latent variables cannot be adjusted for. Adjustment is conditioning, not intervention, and a diagram gives no size of effect.

Examples

```
dag <- ggcausal(cleft_dag$edges, cleft_dag$nodes, legend_title = "Role")
dag

# Only the arrows are required.
ggcausal(data.frame(from = c("A", "A", "B"), to = c("B", "C", "C")))
```

ggchoropleth

*Draw an interactive choropleth map over time***Description**

Colors every region of a map by a measure, with one map per measure side by side, and a slider and play button under them that step through the years. All the maps show the same year, so a region's measures read together: hovering over a country outlines it on every map and lists its value and rank on each measure that year, and clicking it opens its whole series under the maps. The year can be dragged to or played through, and the shades change in place. Ranks count from the highest value that year, so for a measure where lower is better, such as mortality, rank 1 is the worst.

Usage

```
ggchoropleth(
  data,
  region,
  time,
  values,
  map = "world",
  map_id = NULL,
  at = NULL,
  ncol = NULL,
  palette = NULL,
  interval = 0.6,
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

data	A data frame with a row for each region and time.
region	Column of data naming the region: a code or an English name for the world map, or a value of map_id for an 'sf' map.
time	Column of data giving the time, usually the year.
values	Columns of data to map, as a character vector, one map each. Names label the maps; without them, the columns' label attributes or names are used.
map	"world", or an 'sf' object of polygons.

<code>map_id</code>	For an 'sf' map, the column holding the keys region matches. Defaults to the first character or factor column.
<code>at</code>	The time shown first, and in static copies. Defaults to the last.
<code>ncol</code>	Maps per row. Defaults to two, or one for a single measure.
<code>palette</code>	Colors for the maps, one entry per measure, in order or named by map label. One color gives a sequential scale; two give a diverging scale, the first for values below zero and the second for values above.
<code>interval</code>	Seconds each time is shown while playing.
<code>title, caption</code>	Title above the maps and note below them.
<code>family</code>	Font family. The package ships Lato and registers it on load.

Details

The default map is the world: Natural Earth's 1:50m countries in the Equal Earth projection, bundled with the package. Regions are matched by ISO 3166-1 alpha-3 or alpha-2 code, or by English name, allowing for case, accents and punctuation and for the forms the WHO and the Global Burden of Disease study use, such as "Iran (Islamic Republic of)". Rows for places that are not on the map, such as world regions and income groups, are left out with a message naming them.

Any other map can be given as an 'sf' object of polygons, projected as it should be drawn, with `map_id` naming the column that region matches. A map in longitude and latitude is scaled so a degree of longitude has its true length at the middle of the map. Detailed boundaries make a heavy page, so simplify them first, for example with `sf::st_simplify()`.

Each measure has its own color scale, fixed across the years, so a change of shade is a change of value. A measure with values on both sides of zero, such as a change since the first year, gets a diverging scale centered on zero. Shades are a color at increasing strength, drawn with transparency, so a map suits a light or a dark page.

Value

An object of class `ggchoropleth`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot of the year at or a file, and `animate_choropleth()` to play the years as an animation.

Examples

```
qci <- clefts_qci_world
first <- qci$qci[qci$year == 1990][match(qci$iso3, qci$iso3[qci$year == 1990])]
qci$change <- qci$qci - first
ggchoropleth(qci, iso3, year,
  values = c("Quality of Care Index" = "qci",
    "Change since 1990" = "change"),
  title = "Quality of care for orofacial clefts")
```

ggdiagnostic

*Draw an interactive diagnostic threshold explorer***Description**

Shows what a cutoff on a continuous test means for the people tested. The marker's distribution in those with and without the condition sits beside the ROC curve and the predictive values across prevalence, above a grid of 1,000 people who are found, missed, falsely alarmed or correctly cleared, and a table of sensitivity, specificity, predictive values and likelihood ratios with their confidence intervals.

Usage

```
ggdiagnostic(
  formula,
  data,
  cutoff = NULL,
  prevalence = NULL,
  direction = c("auto", "higher", "lower"),
  labels = NULL,
  marker_label = NULL,
  level = 0.95,
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

formula	A formula outcome ~ marker. The outcome is logical, 0 and 1, or a factor or text with two values, whose second level (or TRUE, or 1) means the condition is present. The marker is numeric.
data	A data frame holding both.
cutoff	The prespecified cutoff. A result at or beyond it, in the direction of direction, is positive. Defaults to the cutoff that maximizes Youden's index in these data.
prevalence	The prevalence of the condition where the test will be used, between 0 and 1. Defaults to the prevalence in data.
direction	Whether "higher" or "lower" values point to the condition. "auto" picks the direction with an area under the curve of at least one half.
labels	Names for those without and with the condition, in that order.
marker_label	Axis label for the marker, with its unit.
level	Confidence level for the intervals.
title, caption	Title above the plot and note below it.
family	Font family. The package ships Lato and registers it on load.

Details

In the widget, dragging the cutoff on the distributions or moving its slider updates everything at once, and a second slider sets the prevalence of the population the test will be used in, which changes the predictive values and the grid but not sensitivity or specificity. A sentence under the plot says what the current cutoff does in words.

A cutoff chosen in the same data it is judged on looks better than it will in new patients, so a prespecified cutoff is marked as such, and the default, the cutoff that maximizes Youden's index, is labeled as chosen in these data. A test cutoff is not a treatment threshold, and in a case control sample the prevalence in the data is not the prevalence in practice; set prevalence to the one that applies.

Value

An object of class `ggdiagnostic`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static `ggplot` or a file. The field `accuracy` holds the measures at the cutoff, `roc` the curve and `auc` the area under it.

Intervals

Sensitivity and specificity have Wilson score intervals. Predictive values at a set prevalence use the logit intervals of Mercaldo, Lau and Zhou (2007), likelihood ratios the log method, and the area under the curve the method of DeLong, DeLong and Clarke-Pearson (1988). Intervals that need a count of zero are not shown.

Examples

```
if (requireNamespace("MASS", quietly = TRUE)) {
  pima <- rbind(MASS::Pima.tr, MASS::Pima.te)
  ggdiagnostic(type ~ glu, pima, cutoff = 126, prevalence = 0.1,
    labels = c("No diabetes", "Diabetes"),
    marker_label = "Plasma glucose (mg/dL)")
}
```

ggfunnel

Draw an interactive funnel plot for a meta-analysis

Description

Draws each study's effect against its standard error, with the most precise studies at the top, to show small-study effects. The shaded contours mark where a study would be statistically significant against no effect (Peters et al. 2008), so a gap in the unshaded area, where studies would not be significant, points to publication bias rather than heterogeneity alone. The solid line is the pooled estimate and the dashed lines around it the region where 95% of studies would fall without heterogeneity or bias.

Usage

```
ggfunnel(
  x,
  data = NULL,
  rob = NULL,
  hover = NULL,
  contours = c(0.1, 0.05, 0.01),
  trim_fill = FALSE,
  tests = TRUE,
  exponentiate = NULL,
  xlim = NULL,
  xlab = NULL,
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

x	A fitted meta-analysis: an <code>rma.uni</code> object from <code>metafor::rma()</code> , or a meta object from the 'meta' package, such as the result of <code>meta::metabin()</code> or <code>meta::metagen()</code> . Models with moderators are not supported.
data	Optional data frame with one row per study, in the order of the model, holding the columns to show. Defaults to the data stored in the fit, which is there when the model was fitted with a data argument.
rob	Name of the column of data holding each study's overall risk of bias judgment, which colors its point. Judgments are matched by their wording, as in <code>ggmeta()</code> .
hover	Names of columns of data shown in each study's hover card. Defaults to columns.
contours	Significance levels for the shaded contours, against no effect. NULL draws none.
trim_fill	Add the studies imputed by trim and fill and the adjusted estimate.
tests	Add the tests for small-study effects, in a collapsed section under the plot.
exponentiate	Show effects on the ratio scale. Defaults to TRUE for ratio measures (RR, OR, HR, IRR, ROM and Peto odds ratios), which are modeled on the log scale.
xlim	Optional limits of the effect axis, on the scale shown.
xlab	Axis label. Defaults to the name of the effect measure.
title, caption	Title above the plot and note below it.
family	Font family. The package ships Lato and registers it on load.

Details

Hovering over a study shows its effect, weight, risk of bias, the columns named in `hover` and the significance zone it falls in. Clicking it opens the pooled estimate with that study left out, beside its full record.

With `tests = TRUE`, a section under the plot, collapsed until the reader opens it, gives Egger's regression test (Egger et al. 1997), Begg's rank correlation test (Begg and Mazumdar 1994) and, with `trim_fill = TRUE`, the trim and fill estimate (Duval and Tweedie 2000), as computed by 'metafor' or 'meta', with a note on what each asks. Egger's test is the classical one, `metafor::regtest(model = "lm")`, which 'meta' also computes; metafor's own default, `regtest()` with `model = "rma"`, gives a different p value. The two packages' trim and fill estimators can also impute different numbers of studies from the same data. The table is also returned as the field `tests`. These tests have little power with fewer than ten studies, and asymmetry can come from heterogeneity, chance or the quality of small studies as well as from publication bias.

With `trim_fill = TRUE`, the studies trim and fill imputes are drawn as hollow circles and the adjusted estimate as a dashed line, and the widget gets a switch that hides them.

Value

An object of class `ggfunnel`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `tests` holds the tests as a data frame.

Examples

```
if (requireNamespace("metafor", quietly = TRUE)) {
  dat <- metafor::escalc(measure = "RR", ai = tpos, bi = tneg,
                        ci = cpos, di = cneg, data = metadat::dat.bcg,
                        slab = paste(author, year))
  fit <- metafor::rma(yi, vi, data = dat)
  f <- ggfunnel(fit, hover = "alloc", trim_fill = TRUE)
  f
  f$tests
}
```

ggkm

Draw an interactive Kaplan-Meier plot

Description

Draws Kaplan-Meier curves by group with confidence bands, censoring marks and a table of the numbers at risk, followed by the hazard ratios and the proportional hazards tests. Hovering anywhere along the time axis reads every group at that time: survival with its confidence interval, the number at risk, the events so far and the hazard ratio against the reference group at that time, while the matching column of the risk table lights up. Clicking opens the full readout for that time under the plot. Hovering over a curve shows that group's summary, and clicking it opens the group's survival at each break.

Usage

```
ggkm(
  formula,
  data,
```

```

type = c("survival", "risk"),
hr_time = c("schoenfeld", "log", "linear", "constant", "none"),
ph_tests = FALSE,
rmst = NULL,
risk_table = TRUE,
conf_int = TRUE,
breaks = NULL,
reference = NULL,
palette = NULL,
xlab = "Time",
ylab = NULL,
legend = TRUE,
legend_title = NULL,
title = NULL,
caption = NULL,
family = "Lato"
)

```

Arguments

formula	A formula of the form <code>Surv(time, status) ~ group</code> , with right censored survival times and a single grouping variable.
data	A data frame holding the variables in formula.
type	"survival" for the survival probability, or "risk" for the cumulative probability of the event.
hr_time	How the hazard ratio at each time is estimated: "schoenfeld", "log", "linear", "constant" or "none".
ph_tests	Add the hazard ratios and proportional hazards tests, in a collapsed section under the plot. The time interaction models grow with the number of events, so they are skipped with a message for very large data.
rmst	Restricted mean survival time: NULL for none, the prespecified horizon τ on the time scale, or TRUE for the end of follow-up in the group followed least.
risk_table	Show the numbers at risk.
conf_int	Shade the confidence bands.
breaks	Times for the axis ticks and the risk table. Defaults to about eight evenly spaced times.
reference	The group the hazard ratios compare against. Defaults to the first level.
palette	Colors for the groups, unnamed in level order or named by group. The reference group defaults to a neutral gray.
xlab, ylab	Axis labels. xlab also names the time in the hover card, so include its unit, such as "Years since randomization".
legend	Draw a legend of the groups above the plot.
legend_title	Text in front of the legend.
title, caption	Title above the plot and note below it.
family	Font family. The package ships Lato and registers it on load.

Details

The hazard ratio at a time comes from `hr_time`. The default, "schoenfeld", smooths the scaled Schoenfeld residuals of the Cox model against time, which estimates the log hazard ratio as a function of time (Grambsch and Therneau 1994), the curve `survival::plot.cox.zph()` draws. "log" and "linear" take it from a Cox model with an interaction between the group and log time or time, and "constant" shows the Cox estimate at every time.

With `rmst`, the plot shades the area under each curve up to a horizon τ and tabulates the restricted mean survival time, the mean time alive (or free of the event) within τ , for each group, with its difference from the reference group. RMST and its standard error are those `survival::survfit()` reports for each group; the difference assumes the groups are independent, as in a randomized comparison. The widget adds a slider to explore other horizons, up to the end of follow-up in the group followed least, while the prespecified horizon stays marked, so the horizon reported is the one planned rather than the most favorable one.

With `ph_tests = TRUE`, a section under the plot, collapsed until the reader opens it, gives the Cox hazard ratios, the log-rank test, the Grambsch and Therneau test for each comparison and overall, and the group by time and group by log time interactions with their joint Wald tests, with a note on what each test asks. The table is also returned as the field `ph`. It belongs to the widget, so static copies from `graph_plot()` and `graph_save()` leave it out.

Value

An object of class `ggkm`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file, and `animate_km()` to draw the curves over follow-up as an animation. With `ph_tests = TRUE`, the field `ph` holds the proportional hazards table as a data frame.

Examples

```
if (requireNamespace("survival", quietly = TRUE)) {
  colon <- subset(survival::colon, etype == 2)
  colon$years <- colon$time / 365.25
  km <- ggkm(survival::Surv(years, status) ~ rx, data = colon,
    xlab = "Years since randomization")
  km
}

# The proportional hazards tests fit interaction models, which takes a
# few seconds.
if (requireNamespace("survival", quietly = TRUE)) {
  km <- ggkm(survival::Surv(years, status) ~ rx, data = colon,
    ph_tests = TRUE, xlab = "Years since randomization")
  km$ph
}
```

ggleague

Draw an interactive league table for a network meta-analysis

Description

Draws every pairwise estimate of a network meta-analysis as a grid, with the treatments on the diagonal and a ranking beside it. Following `netmeta::netleague()`, each cell compares the treatment that comes first in the table with the one that comes second: network estimates sit below the diagonal and direct estimates, from the trials that compare the pair head to head, above it.

Usage

```
ggleague(
  x,
  data = NULL,
  study = NULL,
  treatment = NULL,
  pooled = NULL,
  small_values = NULL,
  order = NULL,
  ranking = TRUE,
  contributions = NULL,
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

<code>x</code>	A network meta-analysis from <code>netmeta::netmeta()</code> .
<code>data</code>	Optional arm level data, one row per study arm, for the click panels, such as the data given to <code>ggnma()</code> . Every column other than <code>study</code> and <code>treatment</code> becomes a row of the arm table.
<code>study, treatment</code>	Bare column names in <code>data</code> identifying the study and treatment of each arm. Treatment names must match those in <code>x</code> .
<code>pooled</code>	Which model to show, "random" or "common". Defaults to the random effects model when <code>x</code> has one.
<code>small_values</code>	Whether small values of the effect are "desirable", as for mortality, or "undesirable", as for a response. Sets which treatment a cell favors and the direction of the ranking. Defaults to the setting stored in <code>x</code> , which is worth checking.
<code>order</code>	Order of the treatments along the diagonal. Defaults to the ranking, best first.
<code>ranking</code>	Draw the P-score ranking beside the table.

contributions	Show where each network estimate comes from: TRUE to compute the share of it that flows through each direct comparison with <code>netmeta::netcontrib()</code> , or an object that function returned, which saves recomputing it for a large network. Hovering or tapping a network estimate then marks the direct comparisons it draws on with their shares, and its panel lists them.
title,caption	Title above the table and note below it. The default caption explains which estimates sit on each side of the diagonal.
family	Font family. The package ships Lato and registers it on load.

Details

Cells are shaded by the size of the effect, in one color when it favors the first treatment and another when it favors the second, and faded when the confidence interval includes the null. Hovering over a cell shows its network, direct and indirect estimates and the share of the network estimate that comes from direct trials. Clicking a cell opens the direct trials under the table: their arm level data when data is given, and otherwise each trial's own estimate. Hovering over a treatment on the diagonal or in the ranking lights its row and column.

Value

An object of class `ggleague`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static `ggplot` or a file. The fields `treatments` and `pscore` give the order used and the P-scores.

Examples

```
if (requireNamespace("netmeta", quietly = TRUE) &&
    requireNamespace("meta", quietly = TRUE)) {
  pw <- meta::pairwise(treat = treatment, event = pasi75_r,
                      n = pasi75_n, studlab = study,
                      data = psoriasis_nma, sm = "OR")
  nma <- netmeta::netmeta(pw, common = FALSE)
  ggleague(nma, psoriasis_nma, study, treatment,
           small_values = "undesirable")
}
```

ggmeta

Draw an interactive forest plot for a meta-analysis

Description

Draws the forest plot of a fitted meta-analysis, with the data behind every study a hover and a click away. Hovering over a study shows its effect, weight and the columns named in hover; clicking it opens its full record, every column of data, under the plot. Hovering over the pooled diamond shows the heterogeneity statistics and the prediction interval. Risk of bias judgments, given in `rob`, are drawn as traffic lights beside each study.

Usage

```
ggmeta(
  x,
  data = NULL,
  columns = NULL,
  rob = NULL,
  hover = NULL,
  cumulative = FALSE,
  exponentiate = NULL,
  xlim = NULL,
  favors = NULL,
  xlab = NULL,
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

x	A fitted meta-analysis: an <code>rma.uni</code> object from <code>metafor::rma()</code> , or a meta object from the 'meta' package, such as the result of <code>meta::metabin()</code> or <code>meta::metagen()</code> . Models with moderators are not supported.
data	Optional data frame with one row per study, in the order of the model, holding the columns to show. Defaults to the data stored in the fit, which is there when the model was fitted with a data argument.
columns	Names of columns of data to print as text columns beside the study labels, such as event counts. Name the vector to set the column headers.
rob	Names of columns of data holding risk of bias judgments, one per domain, drawn as traffic lights. Name the vector to set the column headers, such as <code>c(D1 = "rob.R", Overall = "rob.overall")</code> . Judgments are matched by their wording: "low", "some concerns" or "moderate", "unclear", "high" or "serious", "critical", and "no information", in any case.
hover	Names of columns of data shown in each study's hover card. Defaults to columns.
cumulative	Show the cumulative meta-analysis rather than the individual studies.
exponentiate	Show effects on the ratio scale. Defaults to TRUE for ratio measures (RR, OR, HR, IRR, ROM and Peto odds ratios), which are modeled on the log scale.
xlim	Optional limits of the effect axis, on the scale shown.
favors	Optional labels for the two sides of the null line, such as <code>c("Favors treatment", "Favors control")</code> .
xlab	Axis label. Defaults to the name of the effect measure.
title, caption	Title above the plot and note below it.
family	Font family. The package ships Lato and registers it on load.

Details

Square area follows the study's weight in the model. The diamond is the pooled estimate with its confidence interval, and the line through it the prediction interval for a random effects model. Ratio measures such as risk, odds and hazard ratios are shown on a log axis. A confidence interval that runs past the axis ends in an arrow.

With `cumulative = TRUE`, each row shows the pooled estimate from the studies up to and including it, in the order of the data, so sort the data by year before fitting the model for a cumulative meta-analysis over time. `animate_meta()` replays that sequence as an animation.

Value

An object of class `ggmeta`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static `ggplot` or a file, and `animate_meta()` for the cumulative replay.

Examples

```
if (requireNamespace("metafor", quietly = TRUE)) {
  dat <- metafor::escalc(
    measure = "OR", ai = p2y12.mi, n1i = p2y12.total,
    ci = aspirin.mi, n2i = aspirin.total,
    data = metadat::dat.chiarito2020, slab = paste(study, year)
  )
  dat <- dat[!is.na(dat$yi), ]
  dat$p2y12 <- paste0(dat$p2y12.mi, "/", dat$p2y12.total)
  dat$aspirin <- paste0(dat$aspirin.mi, "/", dat$aspirin.total)
  fit <- metafor::rma(yi, vi, data = dat)

  ggmeta(
    fit,
    columns = c("P2Y12 inhibitor" = "p2y12", Aspirin = "aspirin"),
    rob = c(R = "rob.R", D = "rob.D", Mi = "rob.Mi", Me = "rob.Me",
            S = "rob.S", Overall = "rob.overall"),
    favors = c("Favors P2Y12 inhibitor", "Favors aspirin")
  )
}
```

Description

Draws a specification curve: every reasonable analysis of the same question, one per row of data, sorted by its estimate, with its confidence interval, above a grid that marks the choices each analysis made. The primary analysis stays marked, the reference value is drawn, and beside each choice sits the median estimate of the analyses that made it, so the choices that move the result stand out.

Usage

```
ggmultiverse(
  data,
  estimate,
  lower,
  upper,
  decisions,
  primary = NULL,
  ratio = NULL,
  ylab = "Estimate",
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

<code>data</code>	A data frame with one row per analysis.
<code>estimate, lower, upper</code>	Bare columns of data with each estimate and its confidence interval.
<code>decisions</code>	Names of the columns of data that hold the choices, such as the outcome definition, the adjustment set and the model.
<code>primary</code>	Optional logical expression of the columns of data, or a row number, marking the primary analysis.
<code>ratio</code>	Whether the estimates are ratios, drawn on a log scale around 1. Defaults to TRUE when every lower limit is positive.
<code>ylab</code>	Label of the estimate axis, such as "Odds ratio".
<code>title, caption</code>	Title above the plot and note below it.
<code>family</code>	Font family. The package ships Lato and registers it on load.

Details

In the widget, hovering over an analysis shows its estimate and every choice behind it. Dragging across the curve selects a run of analyses and says which choices they share. Clicking a choice in the grid keeps only the analyses that made it, and several choices can be combined; the primary analysis is never hidden.

The share of analyses with intervals that exclude the reference value is a description of this set of analyses, not a probability that the effect is real, and the analyses are not independent.

Value

An object of class `ggmultiverse`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `influence` holds the median estimate for every choice.

Examples

```
specs <- expand.grid(outcome = c("Primary", "Broad"),
                    adjustment = c("Minimal", "Standard", "Extended"),
                    model = c("Logistic", "Log-binomial"),
                    stringsAsFactors = FALSE)

set.seed(1)
log_or <- -0.3 + 0.12 * (specs$outcome == "Broad") +
  0.1 * match(specs$adjustment, c("Minimal", "Standard", "Extended")) +
  rnorm(nrow(specs), 0, 0.03)
specs$or <- exp(log_or)
specs$lo <- exp(log_or - 1.96 * 0.09)
specs$hi <- exp(log_or + 1.96 * 0.09)
ggmultiverse(specs, or, lo, hi, decisions = c("outcome", "adjustment", "model"),
             primary = outcome == "Primary" & adjustment == "Standard" & model == "Logistic",
             ylab = "Odds ratio")
```

ggnma

Draw an interactive network plot for a network meta-analysis

Description

Draws the network of treatment comparisons from arm level data. Each node is a treatment and each line joins two treatments compared directly in at least one study. Hovering over a node shows a compact table of every arm on that treatment; hovering over a line shows the arms of each study that makes that comparison. The card holds the columns named in hover. Clicking either opens a panel under the plot with the full arm level data side by side, one column per arm, in the manner of a trial's baseline table: every column of data other than study, treatment and group becomes a row, so baseline characteristics and outcomes are shown as they were given.

Usage

```
ggnma(
  data,
  study,
  treatment,
  n = NULL,
  group = NULL,
  hover = NULL,
  multiarm = TRUE,
  positions = NULL,
  palette = NULL,
  legend = TRUE,
  legend_title = NULL,
  title = NULL,
  caption = NULL,
  family = "Lato",
  contributions = NULL
)
```

Arguments

<code>data</code>	A data frame with one row per study arm.
<code>study, treatment</code>	Bare column names identifying the study and the treatment of each arm. Each treatment may appear once per study.
<code>n</code>	Optional bare column giving the number of participants in each arm. Sets node area and the participant counts in the hover cards.
<code>group</code>	Optional bare column naming a class for each treatment, such as a drug class. Nodes are then colored by class and a legend is drawn. Each treatment must belong to exactly one class.
<code>hover</code>	Names of the columns shown in the hover card, one row each, such as the sample size, an outcome and a key baseline characteristic. Defaults to the first four columns of the arm table. Use <code>character(0)</code> for a card that only lists the studies. The click panel always shows every column.
<code>multiarm</code>	Shade a polygon for each set of treatments compared in a study with more than two arms.
<code>positions</code>	Optional data frame placing the nodes by hand, with columns <code>treatment</code> , <code>x</code> and <code>y</code> , one row per treatment and <code>y</code> pointing up. Any units will do: the layout is scaled to fit the plot, keeping its shape. Labels point away from the middle of the layout.
<code>palette</code>	Node colors. With <code>group</code> , a vector named by class, or an unnamed vector recycled over the classes; without it, a single color. Defaults to <code>race_palette()</code> .
<code>legend</code>	Draw the legend when <code>group</code> is given.
<code>legend_title</code>	Text in front of the legend, such as "Class".
<code>title, caption</code>	Title above the plot and note below it.
<code>family</code>	Font family. The package ships Lato and registers it on load.
<code>contributions</code>	Show where the evidence for each comparison comes from: a fit from <code>netmeta::netmeta()</code> on the same network, whose contributions are then computed with <code>netmeta::netcontrib()</code> , or an object that function returned. The widget gains a menu of every comparison; picking one widens and colors each line by the share of that network estimate flowing through it, labels the shares and says them in words under the plot.

Details

Treatments sit on a circle, starting at the top and running clockwise in the order of the levels of treatment when it is a factor, or in order of first appearance otherwise; `positions` places them by hand instead. Line width follows the number of studies that make the comparison. When `n` is given, node area follows the total number of participants on that treatment. A study with more than two arms adds a line for every pair of its treatments and, with `multiarm`, a shaded polygon joining them. Studies that compare the same set of treatments share one polygon, which has its own hover card and panel.

Row labels come from each column's `label` attribute when it has one, as set by the 'labelled', 'Hmisc' or 'haven' packages, and otherwise from its name. Text that contains a URL or a DOI, such as a reference column, is linked in the panel.

Value

An object of class `ggnma`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static `ggplot` or a file. The fields `nodes` and `edges` hold the treatments and comparisons with their study counts, `multiarm` the sets of treatments compared in multi-arm studies, and `width` and `height` the natural size in inches. With contributions, the field `contributions` holds the shares.

Examples

```
net <- ggnma(psoriasis_nma, study, treatment, n = n, group = class,
            legend_title = "Class")
net
net$edges

if (requireNamespace("netmeta", quietly = TRUE) &&
    requireNamespace("meta", quietly = TRUE)) {
  pw <- meta::pairwise(treat = treatment, event = pasi75_r, n = pasi75_n,
                      studlab = study, data = psoriasis_nma, sm = "OR")
  fit <- netmeta::netmeta(pw, common = FALSE)
  ggnma(psoriasis_nma, study, treatment, n = n, contributions = fit)
}
```

ggnomogram

Draw an interactive nomogram for a regression model

Description

Draws the nomogram of a fitted regression model: one axis per predictor, scaled in points, a total points axis and one or more axes that turn the total into a prediction. In the widget every predictor has a handle. Dragging it, clicking a category or using the arrow keys sets a patient's values, and the points, the total and the prediction with its 95% confidence interval follow, computed in the page from the model's coefficients and their covariance.

Usage

```
ggnomogram(
  fit,
  data = NULL,
  values = NULL,
  labels = NULL,
  ranges = NULL,
  outcome = NULL,
  times = NULL,
  level = NULL,
  title = NULL,
  caption = NULL,
```

```

    family = "Lato"
  )

```

Arguments

<code>fit</code>	A fitted regression model. See Details for the classes read.
<code>data</code>	The data the model was fitted to. Defaults to the data named in the model's call, which is enough when that data is still around.
<code>values</code>	Starting values for the predictors, as a named list. The others start at the median, or at the most common category.
<code>labels</code>	Axis labels for the predictors, as a named character vector. The others use the column's <code>label</code> attribute or its name.
<code>ranges</code>	Ranges for numeric axes, as a named list of pairs.
<code>outcome</code>	Label of the prediction axis. For a model with several, a character vector with one label each.
<code>times</code>	For a survival model, the times to predict survival at, named to label the axes, such as <code>c("1 year" = 365, "5 years" = 1826)</code> . Defaults to the median follow-up time.
<code>level</code>	For a multinomial model, the category whose nomogram the static copy draws. Defaults to the first after the reference.
<code>title, caption</code>	Title above the nomogram and note below it.
<code>family</code>	Font family. The package ships Lato and registers it on load.

Details

Contributions are computed through the model's design matrix, so transformed and nonlinear terms, such as `log(x)`, `poly()`, `ns()`, `rcs()` or a smooth from 'mgcv', are drawn as they were fitted. An axis whose effect rises and then falls is folded onto more than one line, as rms does. When predictors interact, the axis of a later predictor in the interaction is drawn for the current values of the earlier ones and redraws as they change. Numeric axes run from the 2.5th to the 97.5th percentile of the data unless `ranges` says otherwise.

The prediction depends on the model:

Linear models `lm()`, `nlme::gls()`, `quantreg::rq()`, `rms::ols()` and Gaussian models with an identity link: the predicted mean, with a prediction interval for `lm()`.

Generalized linear models `glm()`, `MASS::glm.nb()`, `geepack::geeglm()`, `logistf::logistf()`, `rms::lrm()` and `mgcv::gam()`: the mean on the response scale, such as a probability or a rate, through the model's link.

Mixed models `lme4::lmer()`, `lme4::glmer()`, `nlme::lme()` and `glmmTMB::glmmTMB()`: the prediction for a typical cluster, with the random effects at zero. On a nonlinear link this is the prediction for that cluster, not the average over the population. The interval reflects the fixed effects only. For `glmmTMB`, zero inflation and dispersion models are left out.

Cox models `survival::coxph()` and `rms::cph()`: survival at each of times, with the interval `survival::survfit()` gives for the same patient. A stratified model gets a choice of stratum.

Parametric survival models `survival::survreg()` and `rms::psm()`: the median survival time and survival at times.

Ordinal models `MASS::polr()`, `ordinal::clm()`, `rms::orm()` and `ordinal rms::lrm()`: the probability of each category or above, with the probability of every category beside the plot.

Multinomial models `nnet::multinom()`: one nomogram per category, scaled on its odds against the reference category, with the probability of every category beside the plot.

Value

An object of class `ggnomogram`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static `ggplot` or a file. `nomogram_predict()` gives the prediction for any set of values, as the widget computes it.

Examples

```
bw <- MASS::birthwt
bw$race <- factor(bw$race, labels = c("White", "Black", "Other"))
bw$smoke <- factor(bw$smoke, labels = c("No", "Yes"))
fit <- glm(low ~ age + lwt + race + smoke, family = binomial, data = bw)
n <- ggnomogram(fit, outcome = "Risk of low birth weight",
               labels = c(age = "Age (years)", lwt = "Weight (lb)"))
n
nomogram_predict(n, list(age = 30, lwt = 110, race = "Black", smoke = "Yes"))
```

`ggrace`

Build a bar chart race

Description

Turns a long panel of name, time, value observations into a bar chart race and returns one row per entity per frame. Draw a single frame with `race_frame()` or write the whole animation with `animate_race()`.

Usage

```
ggrace(
  data,
  value,
  name,
  time,
  group = NULL,
  top_n = 10,
  duration = 25,
  fps = 60,
  end_pause = 2,
  swap = 0.2,
  palette = NULL,
```

```

    legend = TRUE,
    legend_title = NULL,
    title = NULL,
    caption = NULL,
    breaks = scales::breaks_extended(4),
    label_value = scales::label_comma(accuracy = 1),
    label_time = NULL,
    images = NULL,
    image_size = 0.87,
    timeline = TRUE,
    play_button = TRUE,
    card = TRUE,
    width = 736,
    res = 100,
    family = "Lato"
  )

```

Arguments

<code>data</code>	A data frame in long format.
<code>value, name, time</code>	Bare column names holding the bar length, the bar label, and the time point. <code>time</code> may be numeric or a Date. Each name and time pair must be unique.
<code>group</code>	Optional bare column naming a category for each entity, such as a continent. Bars are then colored by category rather than individually, and a legend is drawn above the axis. Each entity must belong to exactly one category. A factor keeps the legend in the order of its levels.
<code>top_n</code>	Number of bars visible at once.
<code>duration</code>	Length of the animation in seconds, excluding <code>end_pause</code> .
<code>fps</code>	Frames per second.
<code>end_pause</code>	Seconds to hold the final frame.
<code>swap</code>	Seconds a bar takes to move to a new rank. Bars hold their position and change places in one quick eased move rather than drifting the whole way between one time point and the next.
<code>palette</code>	Colors for the bars. Either an unnamed vector, recycled in alphabetical order, or a named vector. Names are entities, or categories when <code>group</code> is given. Defaults to race_palette() .
<code>legend</code>	Draw the legend when <code>group</code> is given. The card grows to make room for it, wrapping onto more rows if the categories do not fit across the card.
<code>legend_title</code>	Text in front of the legend, such as "Continent".
<code>title, caption</code>	Card title and the source note under the timeline.
<code>breaks</code>	A function taking the axis range and returning the gridline positions, or a numeric vector of fixed positions. Breaks past the current maximum are dropped, so the axis fills in as the field grows.
<code>label_value</code>	A function formatting the number printed after each bar.

label_time	A function formatting the large time label in the corner. Defaults to the floor of the interpolated time for numeric input and the year for dates.
images	Pictures to sit at the end of each bar, as a character vector of image file paths named by entity. Entities with no image get none. race_flags() returns paths to the bundled country flags. Needs the magick package.
image_size	Image diameter as a fraction of the bar height. Images are cropped to a circle and right aligned just inside the end of the bar.
timeline	Draw the timeline strip with the moving marker.
play_button	Draw the round pause button next to the timeline.
card	Draw the chart on a white card with a drop shadow over a light page. Set to FALSE for a plain white background.
width	Output width in pixels. Frames are laid out for this width, so it also fixes every font size.
res	Output resolution in pixels per inch.
family	Font family. The package ships Lato and registers it on load.

Details

Values move linearly and ranks do not. Every frame is ranked on its own interpolated values, and a bar whose rank changes eases into its new slot over swap seconds. Bars therefore rest in place and trade positions in one short move, rather than drifting for a whole time step, which is what keeps a crowded field readable while it reorders.

Ranks are clamped to $\text{top_n} + 1$. An entity far down the field therefore waits just below the visible window and slides in from the bottom edge instead of flying up from off screen, which is what makes entries and exits read cleanly.

The x axis carries no headroom. The longest bar always reaches the right edge of the plotting area and the axis maximum is the largest value in the current frame, so gridlines drift as the field grows.

Value

An object of class `ggrace`. It prints as an interactive widget, like the package's other graphs: the card's play button and timeline work, and hovering over a bar shows its value and rank. Use [graph_widget\(\)](#) or [graph_save\(\)](#) with a .html file for the widget, [race_frame\(\)](#) or [graph_plot\(\)](#) for a static frame, and [animate_race\(\)](#) for a GIF or video.

Examples

```
phones <- as.data.frame.table(datasets::WorldPhones, responseName = "phones")
names(phones)[1:2] <- c("year", "region")
phones$year <- as.numeric(as.character(phones$year))

race <- ggrace(phones, phones, region, year, top_n = 7, duration = 5)
race

# Frames are set in Lato, which only devices that understand registered
# fonts can use, so draw them with ragg rather than the default device.
file <- tempfile(fileext = ".png")
```

```

ragg::agg_png(file, width = race$width, height = 500, units = "px",
              res = race$res)
print(race_frame(race, 60))
dev.off()

animate_race(race, tempfile(fileext = ".gif"), cores = 1)

```

ggresponder

Draw an interactive responder threshold plot

Description

Shows what a responder definition keeps and what it throws away. For two arms, the curves give the share of patients in each arm who improved by at least each amount, so the whole distribution of change stays in view, with the prespecified threshold marked. Beside them, the difference in responders is drawn across every possible threshold with its confidence band, which shows how much the conclusion depends on where the line is drawn. Under both, a table gives the responders in each arm, their difference, the number needed to treat, and the mean difference, which uses every patient.

Usage

```

ggresponder(
  formula,
  data,
  threshold,
  higher_is_better = TRUE,
  reference = NULL,
  xlab = "Improvement from baseline",
  level = 0.95,
  title = NULL,
  caption = NULL,
  family = "Lato"
)

```

Arguments

formula	A formula <code>change ~ arm</code> , where <code>change</code> is each patient's change from baseline and <code>arm</code> has two values.
data	A data frame holding both.
threshold	The prespecified improvement that makes a responder, in the units of change, such as a minimal important difference.
higher_is_better	Whether a rise in change is an improvement. When <code>FALSE</code> , as for pain, the change is turned around so that improvement is positive.
reference	The control arm. Defaults to the first level of arm.

xlab	Label of the improvement axis, with its unit.
level	Confidence level for the intervals.
title, caption	Title above the plot and note below it.
family	Font family. The package ships Lato and registers it on load.

Details

In the widget, a slider or a drag on either panel moves the threshold, and every number and a sentence follow; a button returns to the prespecified threshold.

Responders have Wilson score intervals and their difference the hybrid score interval of Newcombe (1998). The number needed to treat is the reciprocal of the difference; when the interval of the difference includes zero, its interval runs from benefit through infinity to harm, as Altman (1998) describes. The mean difference has a Welch interval.

Value

An object of class `ggresponder`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `responders` holds the measures at the threshold and curve the difference at every threshold.

Examples

```
set.seed(3)
pain <- data.frame(arm = rep(c("Placebo", "Active"), each = 120),
  change = c(rnorm(120, -1.3, 2), rnorm(120, -2.2, 2)))
ggresponder(change ~ arm, pain, threshold = 2, higher_is_better = FALSE,
  xlab = "Improvement in pain (points on a 0 to 10 scale)")
```

`ggsensitivity`

Draw an interactive bias and tipping point explorer

Description

Asks how strong unmeasured confounding would have to be to change a conclusion. The surface covers every pair of strengths an unmeasured confounder could have, as a risk ratio with the exposure and a risk ratio with the outcome, and is shaded by what would remain of the result under it: an effect still clinically important with an interval clear of the null, an interval clear of the null, an estimate on the same side of the null, or nothing. Curves mark where each of these is lost, the E-values for the estimate and for the confidence limit sit on the diagonal, and measured covariates given as benchmarks show how strong confounding of a known kind was.

Usage

```
ggsensitivity(
  estimate,
  lower,
  upper,
  measure = c("RR", "OR", "HR"),
  rare = FALSE,
  important = NULL,
  benchmarks = NULL,
  max_strength = NULL,
  xlab = "Confounder with the exposure (risk ratio)",
  ylab = "Confounder with the outcome (risk ratio)",
  title = NULL,
  caption = NULL,
  family = "Lato"
)
```

Arguments

<code>estimate, lower, upper</code>	The estimate and its confidence interval, on the ratio scale.
<code>measure</code>	"RR", "OR" or "HR".
<code>rare</code>	Whether the outcome is rare, below about 15 percent, so that an odds ratio or hazard ratio can stand in for a risk ratio.
<code>important</code>	The smallest effect that would matter clinically, on the same scale and on the same side of 1 as the estimate, such as 1.25 or 0.8. Optional.
<code>benchmarks</code>	Optional data frame of measured covariates to compare with, with columns <code>label</code> , <code>exposure</code> and <code>outcome</code> : each covariate's risk ratio with the exposure and with the outcome.
<code>max_strength</code>	The largest strength on the axes. Defaults to a little past the E-value.
<code>xlab, ylab</code>	Axis labels of the surface.
<code>title, caption</code>	Title above the plot and note below it.
<code>family</code>	Font family. The package ships Lato and registers it on load.

Details

Beside the surface, the estimate as analyzed is drawn above the estimate adjusted for a chosen confounder, one for each benchmark, and the ones at the two E-values. In the widget, clicking the surface, or moving the two sliders, chooses the confounder, and a sentence says what it would do.

The adjustment divides the estimate by the bounding factor of Ding and VanderWeele (2016), which is the most bias a confounder of those strengths could cause, so the adjusted values are the worst case for each pair. An odds ratio or a hazard ratio for a common outcome is first converted to an approximate risk ratio, as VanderWeele and Ding (2017) propose; set `rare = TRUE` when the outcome is rare, to use it as it is.

Value

An object of class `ggsensitivity`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static ggplot or a file. The field `evaluates` holds the E-values and benchmarks the adjusted estimates for each benchmark.

Examples

```
ggsensitivity(1.8, 1.4, 2.31, important = 1.25,
             benchmarks = data.frame(label = c("Age", "Smoking"),
                                     exposure = c(1.6, 2.3), outcome = c(1.9, 1.5)))
```

ggswimmer

Draw an interactive swimmer plot

Description

Draws one lane per patient: a bar for the time on treatment or on study, marks for the events along it, such as responses, progression and death, and an arrow for patients still ongoing. Hovering over a lane shows the patient's group, the columns named in `hover`, the first time of each event and whether they are ongoing, and fades the other lanes. Clicking it opens the patient's events in order under the plot, beside their full record. Buttons under the widget reorder the lanes, which slide into place.

Usage

```
ggswimmer(
  data,
  id,
  end,
  events = NULL,
  group = NULL,
  ongoing = NULL,
  start = NULL,
  hover = NULL,
  sort = c("duration", "group", "response", "change", "data"),
  palette = NULL,
  ongoing_label = "Ongoing",
  xlab = "Time",
  legend = TRUE,
  title = NULL,
  caption = NULL,
  family = "Lato",
  waterfall = NULL,
  trajectories = NULL,
  thresholds = c(-30, 20),
  change_label = "Best change from baseline (%)"
)
```

Arguments

<code>data</code>	A data frame with one row per patient.
<code>id</code>	Column of data identifying the patient.
<code>end</code>	Column of data, or an expression of its columns, giving the time each bar ends.
<code>events</code>	Optional data frame with one row per event: the patient's id in a column named as <code>id</code> is in <code>data</code> , the time in <code>time</code> and what happened in <code>event</code> .
<code>group</code>	Optional column of data, such as the arm, that colors the bars.
<code>ongoing</code>	Optional logical column of data, or an expression of its columns, marking patients still ongoing at <code>end</code> , drawn with an arrow.
<code>start</code>	Optional column of data giving the time each bar starts. Defaults to zero.
<code>hover</code>	Names of columns of data shown in each lane's hover card.
<code>sort</code>	Order of the lanes in a static copy and when the widget opens: "duration", the longest first; "group"; "response", those with a complete response first, then a partial one; "change", the largest decrease in waterfall first; or "data", the order of data.
<code>palette</code>	Colors for the groups, unnamed in level order or named by group.
<code>ongoing_label</code>	What the arrow means, for the legend and the hover card, such as "On treatment" or "Alive at last follow-up".
<code>xlab</code>	Axis label. Include the unit, such as "Months since first dose".
<code>legend</code>	Draw a legend of the groups and events above the plot.
<code>title, caption</code>	Title above the plot and note below it.
<code>family</code>	Font family. The package ships Lato and registers it on load.
<code>waterfall</code>	Optional column of data, or an expression of its columns, giving each patient's best percent change from baseline, such as in the sum of target lesion diameters. A missing value is marked not evaluable.
<code>trajectories</code>	Optional data frame with one row per assessment: the patient's id in a column named as <code>id</code> is in <code>data</code> , the time in <code>time</code> on the same scale as <code>end</code> , and the percent change from baseline in <code>change</code> . A patient's line starts from no change at the start of the lane when no assessment comes first.
<code>thresholds</code>	The percent decrease that counts as a response and the percent increase that counts as progression, marked on the change panels. The default, <code>c(-30, 20)</code> , is the rule for target lesions in RECIST 1.1, which also counts new lesions and other progression; the panels show only the measured change.
<code>change_label</code>	Axis label for the percent change.

Details

With `waterfall`, each lane also gets a bar for the patient's best change from baseline, beside the lanes, so the swimmer plot and the waterfall plot share one row per patient; an order by best change sorts both. With `trajectories`, each patient's course of change over time is drawn under the lanes on the same time axis. Every mark of a patient in the three panels shares one id, so hovering over any of them lights the patient in all three, and the response and progression thresholds are marked on both change panels, with a count of the patients past each.

Events are drawn by their wording: a complete response as a star, a partial response or other response as a triangle, progression, relapse or recurrence as a diamond, and death as a cross, with any other event as a circle or square in its own color.

Value

An object of class `ggswimmer`, which prints as an interactive widget. Use `graph_widget()`, `graph_plot()` or `graph_save()` for the widget, a static `ggplot` or a file.

Examples

```
if (requireNamespace("survival", quietly = TRUE)) {
  aml <- subset(survival::myeloid, id <= 30)
  month <- 30.44
  events <- rbind(
    data.frame(id = aml$id, time = aml$crttime / month, event = "Complete response"),
    data.frame(id = aml$id, time = aml$txtime / month, event = "Transplant"),
    data.frame(id = aml$id, time = aml$rltime / month, event = "Relapse"),
    data.frame(id = aml$id, time = ifelse(aml$death == 1, aml$futime / month, NA),
      event = "Death")
  )
  events <- events[!is.na(events$time), ]
  ggswimmer(aml, id, futime / month, events = events, group = trt,
    ongoing = death == 0, hover = c("sex", "flt3"),
    ongoing_label = "Alive at last follow-up",
    xlab = "Months since randomization")
}
```

graph_widget

Use an interactive graph as a widget, a ggplot or a file

Description

A bar chart race from `ggrace()`, or a graph built by `ggcausal()`, `ggnma()`, `ggmeta()`, `ggfunnel()`, `ggleague()`, `ggkm()`, `ggswimmer()`, `ggnomogram()`, `ggchoropleth()`, `ggdiagnostic()`, `ggsensitivity()`, `ggmultiverse()`, `ggresponder()`, `cinema_league()`, `cinema_contribution()`, `cinema_clinical()`, `cinema_incoherence()` or `cinema_network()` prints as an interactive widget. These functions give the other forms it can take. `graph_widget()` returns the 'htmlwidgets' object, for use in 'shiny' or to save with `htmlwidgets::saveWidget()`. `graph_plot()` returns the underlying `ggplot`, which draws as an ordinary static plot. `graph_save()` writes either form to a file chosen by its extension.

Usage

```
graph_widget(x, theme = c("auto", "light", "dark"))
```

```
graph_plot(x, theme = c("light", "dark"))
```

```
graph_save(x, file, res = 300, theme = NULL)
```

```
## S3 method for class 'ggx_graph'
knit_print(x, ...)

## S3 method for class 'ggrace'
knit_print(x, ...)
```

Arguments

<code>x</code>	A race from <code>ggrace()</code> , whose static copy is its last frame, or a graph from <code>ggcausal()</code> , <code>ggnma()</code> , <code>ggmeta()</code> , <code>ggfunnel()</code> , <code>ggleague()</code> , <code>ggkm()</code> , <code>ggswimmer()</code> , <code>ggnomogram()</code> , <code>ggchoropleth()</code> , <code>ggdiagnostic()</code> , <code>ggsensitivity()</code> , <code>ggmultiverse()</code> , <code>ggresponder()</code> , <code>cinema_league()</code> , <code>cinema_contribution()</code> , <code>cinema_clinical()</code> , <code>cinema_incoherence()</code> or <code>cinema_network()</code> .
<code>theme</code>	For the widget, "auto" follows the page it sits on: a dark 'pkgdown' or 'bslib' page (data-bs-theme="dark"), a dark Quarto theme, a page marked data-theme="dark", or, for a widget saved as its own page, the viewer's system setting. "light" and "dark" fix it. For <code>graph_plot()</code> and a PNG from <code>graph_save()</code> , "light" or "dark" picks the colors of the static copy; a saved .html defaults to "auto" and a .png to "light".
<code>file</code>	Output path. .html writes the widget as a single file, which needs 'pandoc'; .png writes a static image with 'ragg'.
<code>res</code>	Resolution of a PNG in pixels per inch.
<code>...</code>	Passed to <code>knitr::knit_print()</code> .

Details

The graph is laid out at a fixed size, so its boxes always fit their labels. The widget scales to the width of the page it sits in; a static copy should be drawn at `x$width` by `x$height` inches, which is what `graph_save()` does. The widget embeds a web copy of Lato, so it looks the same on machines without the font.

On a dark page the widget switches to a dark palette of its own: the background, text, lines and neutral fills take their dark counterparts, colors that carry meaning keep their hue, and the hover cards and panels follow. It also follows a page that switches theme while it is open.

Value

`graph_widget()` returns an `htmlwidget` and `graph_plot()` a `ggplot`. `graph_save()` returns `file`, invisibly.

Examples

```
dag <- ggcausal(cleft_dag$edges, cleft_dag$nodes)
w <- graph_widget(dag)
p <- graph_plot(dag)

graph_save(dag, tempfile(fileext = ".png"))
```

nomogram_predict	<i>Predict from a nomogram</i>
------------------	--------------------------------

Description

Gives the linear predictor, its standard error and every prediction the nomogram shows, for any set of predictor values, computed the way the widget computes them. Use it to check a nomogram against the model's own `predict()` method, or to read predictions for a list of patients.

Usage

```
nomogram_predict(x, values = list())
```

Arguments

<code>x</code>	A nomogram from <code>ggnomogram()</code> .
<code>values</code>	A named list of predictor values. Predictors left out take their starting values.

Value

A list with the total points, the linear predictor `lp` and its standard error `se`, and outputs, a data frame with one row per prediction and its confidence interval.

psoriasis_nma	<i>Arm level data from five trials in plaque psoriasis</i>
---------------	--

Description

Baseline characteristics and PASI 75 response for the 15 arms of five randomized trials in moderate to severe plaque psoriasis: CLEAR, ERASURE, FEATURE, FIXTURE and JUNCTURE. The trials compare secukinumab at two doses with placebo, etanercept and ustekinumab, and most have more than two arms, which makes a small but well connected network for `ggnma()`.

Usage

```
psoriasis_nma
```

Format

A data frame with 15 rows, one per arm, and 15 columns:

study Trial name.
treatment Treatment, a factor with placebo first.
class Drug class, a factor with four levels.
n Participants randomized.

pasi75_r, pasi75_n Participants with a PASI 75 response, and the number analyzed.

age Mean age in years.

male Percentage of men.

weight Mean weight in kilograms.

bmi Mean body mass index.

pasi_w0 Mean PASI score at baseline.

duration Mean duration of psoriasis in years.

prior_systemic Percentage with previous systemic treatment.

psa Percentage with psoriatic arthritis.

reference The trial's main publication, with its DOI.

Details

Every column except study, treatment and class carries a label attribute, which `ggmma()` uses to name the rows of its arm tables.

Source

Aggregate data compiled by Phillippo (2019) and distributed as `plaque_psoriasis_agd` in the 'multinma' package. The analysis that uses them is Phillippo DM, Dias S, Ades AE, et al. (2020). Multilevel network meta-regression for population-adjusted treatment comparisons. Journal of the Royal Statistical Society Series A 183(3): 1189-1210. doi:10.1111/rssa.12579. Trial references were checked against PubMed.

Examples

```
psoriasis_nma[c("study", "treatment", "n", "pasi75_r")]
attr(psoriasis_nma$age, "label")
```

race_flags

Bundled circular country flags

Description

Returns file paths to circular flag images, ready to hand to the `images` argument of `ggrace()`. The package ships one flag per ISO 3166-1 country, plus Kurdistan, mostly drawn from the same artwork `gt::fmt_flag()` uses. See `inst/extdata/flags/SOURCE.txt` for the provenance.

Usage

```
race_flags(country)
```

Arguments

`country` Country names or ISO codes, in any mix.

Details

Lookup tries, in order: the two letter code, the three letter code, the full country name, and finally a unique partial match on the name. Names follow the World Bank style used by 'gt', so a few common spellings do not match: pass the code for those, and use `race_flag_codes()` to find it.

Value

A character vector of file paths, named by the input.

Examples

```
race_flags(c("br", "Iran", "Kenya"))

# "Turkey" and "South Korea" are spelled differently in the table, so pass
# their codes instead.
race_flags(c("tr", "kr"))
```

race_flag_codes	<i>Countries the bundled flags cover</i>
-----------------	--

Description

Countries the bundled flags cover

Usage

```
race_flag_codes()
```

Value

A data frame with the two letter code, the three letter code and the country name of every bundled flag.

Examples

```
head(race_flag_codes())
subset(race_flag_codes(), grepl("Korea", country))
```

race_frame

Draw one frame of a bar chart race

Description

The frame is a single `ggplot2::ggplot` laid out in card units, so the title, axis, bars, timeline and footer all live in one coordinate system and land on fixed positions rather than wherever a layout engine puts them. That is what keeps the panel from shifting sideways between frames.

Usage

```
race_frame(x, frame = 1L)
```

Arguments

x	A <code>ggrace</code> object from <code>ggrace()</code> .
frame	Frame index, between 1 and the number of frames.

Details

Frames are set in Lato, which the package registers with 'systemfonts'. Draw them on a device that understands registered fonts, such as `ragg::agg_png()`, which is what `animate_race()` uses. On other devices pass `family = ""` to `ggrace()` to fall back to the device default.

Value

A `ggplot2::ggplot` object.

Examples

```
phones <- as.data.frame.table(datasets::WorldPhones, responseName = "phones")
names(phones)[1:2] <- c("year", "region")
phones$year <- as.numeric(as.character(phones$year))

race <- ggrace(phones, phones, region, year, top_n = 7, family = "")
race_frame(race, 1)
```

race_palette

Colors used by the bar chart race

Description

A qualitative palette of muted teals, greens, blues, purples and warm earth tones, chosen to stay readable side by side in a dense stack of bars. Consecutive colors are far apart in hue, so the first few remain easy to tell apart when only a handful are used, as with group in `ggrace()`.

Usage

```
race_palette(n = 26)
```

Arguments

n Number of colors to return. The palette is recycled when **n** is larger than the number of base colors.

Value

A character vector of **n** hex colors.

Examples

```
race_palette(5)
```

race_size	<i>The pixel size of a race frame</i>
-----------	---------------------------------------

Description

A frame has a fixed aspect ratio, set by the width and by how many bars are visible. [animate_race\(\)](#) uses this to size its device; use it when drawing a single frame yourself, so the layout is not stretched.

Usage

```
race_size(x, width = x$width)
```

Arguments

x A ggrace object from [ggrace\(\)](#).

width Output width in pixels. Defaults to the width the race was built for.

Value

A named numeric vector with the width and height in pixels.

Examples

```
race <- ggrace(clefts_qci, qci, country, year, top_n = 10)
race_size(race)
```

theme_race	<i>The bare theme a race frame is drawn on</i>
------------	--

Description

A race frame paints its own title, axis, timeline and footer as layers in one canvas coordinate system, so the theme only has to supply the page color and get everything else out of the way.

Usage

```
theme_race(page = race_ink$page)
```

Arguments

page	Background color of the page behind the card.
------	---

Value

A `ggplot2::theme` object.

Examples

```
library(ggplot2)
ggplot(mtcars, aes(wt, mpg)) + geom_point() + theme_race()
```

Index

* datasets

- cleft_dag, [22](#)
- clefts_qci, [20](#)
- clefts_qci_world, [21](#)
- psoriasis_nma, [52](#)

animate_choropleth, [3](#)
animate_choropleth(), [26](#)
animate_km, [4](#)
animate_km(), [32](#)
animate_meta, [6](#)
animate_meta(), [36](#)
animate_race, [7](#)
animate_race(), [42, 44, 55, 56](#)

cinema_clinical, [8](#)
cinema_clinical(), [14, 50, 51](#)
cinema_contribution, [9](#)
cinema_contribution(), [14, 50, 51](#)
cinema_incoherence, [11](#)
cinema_incoherence(), [14, 50, 51](#)
cinema_judge, [13](#)
cinema_judge(), [8–10, 12, 17, 19](#)
cinema_league, [16](#)
cinema_league(), [14, 50, 51](#)
cinema_network, [19](#)
cinema_network(), [50, 51](#)
cleft_dag, [22](#)
clefts_qci, [20, 21](#)
clefts_qci_world, [21](#)

ggcausal, [23](#)
ggcausal(), [22, 50, 51](#)
ggchoropleth, [25](#)
ggchoropleth(), [3, 4, 22, 50, 51](#)
ggdiagnostic, [27](#)
ggdiagnostic(), [50, 51](#)
ggfunnel, [28](#)
ggfunnel(), [50, 51](#)
ggkm, [30](#)
ggkm(), [4, 5, 50, 51](#)
ggleague, [33](#)
ggleague(), [16, 50, 51](#)
ggmeta, [34](#)
ggmeta(), [6, 29, 50, 51](#)
ggmultiverse, [36](#)
ggmultiverse(), [50, 51](#)
ggnma, [38](#)
ggnma(), [10, 19, 33, 50–53](#)
ggnomogram, [40](#)
ggnomogram(), [50–52](#)
ggplot2::ggplot, [55](#)
ggplot2::theme, [57](#)
ggrace, [42](#)
ggrace(), [6, 7, 21, 50, 51, 53, 55, 56](#)
ggresponder, [45](#)
ggresponder(), [50, 51](#)
ggsensitivity, [46](#)
ggsensitivity(), [50, 51](#)
ggswimmer, [48](#)
ggswimmer(), [50, 51](#)
graph_plot (graph_widget), [50](#)
graph_plot(), [9, 10, 12, 17, 20, 24, 26, 28, 30, 32, 34, 36, 37, 40, 42, 44, 46, 48, 50, 51](#)
graph_save (graph_widget), [50](#)
graph_save(), [9, 10, 12, 17, 20, 24, 26, 28, 30, 32, 34, 36, 37, 40, 42, 44, 46, 48, 50, 51](#)
graph_widget, [50](#)
graph_widget(), [9, 10, 12, 17, 20, 24, 26, 28, 30, 32, 34, 36, 37, 40, 42, 44, 46, 48, 50](#)

htmlwidgets::saveWidget(), [50](#)

knit_print.ggrace (graph_widget), [50](#)
knit_print.ggx_graph (graph_widget), [50](#)
knitr::knit_print(), [51](#)

`meta::metabin()`, [29](#), [35](#)
`meta::metagen()`, [29](#), [35](#)
`metafor::rma()`, [29](#), [35](#)

`netmeta::decomp.design()`, [15](#), [18](#)
`netmeta::netcontrib()`, [15](#), [17](#), [34](#), [39](#)
`netmeta::netleague()`, [16](#), [33](#)
`netmeta::netmeta()`, [8](#), [10](#), [12](#), [13](#), [17](#), [33](#), [39](#)
`netmeta::netsplit()`, [11](#), [14](#), [15](#), [18](#)
`nomogram_predict`, [52](#)
`nomogram_predict()`, [42](#)

`parallel::mclapply()`, [7](#)
`psoriasis_nma`, [52](#)

`race_flag_codes`, [54](#)
`race_flag_codes()`, [54](#)
`race_flags`, [53](#)
`race_flags()`, [21](#), [44](#)
`race_frame`, [55](#)
`race_frame()`, [42](#), [44](#)
`race_palette`, [55](#)
`race_palette()`, [24](#), [39](#), [43](#)
`race_size`, [56](#)
`ragg::agg_png()`, [55](#)

`sf::st_simplify()`, [26](#)
`survival::plot.cox.zph()`, [32](#)
`survival::survfit()`, [32](#)

`theme_race`, [57](#)