

Package: gghalftone (via r-universe)

September 29, 2026

Title Halftone, Dither and Line-Screen Fills for 'ggplot2'

Version 0.1.0

Date 2026-09-29

Description Halftone screens as ggplot2 layers. Dots are placed at draw time on a lattice with a physical pitch in millimetres, so a figure gets the same screen at any output size. Provides `geom_halftone()` for gridded fields, `geom_spot()` for per-point tone discs, `with_halftone()` to screen the fill of any layer (ribbons, areas, bars, polygons, sf), `with_halo()` and `with_relief()` for lines over a screen, a screen aesthetic for colour-free encodings, line screens, Bayer, blue-noise and Floyd-Steinberg dithering, overprinting of overlapping groups, `with_press()` for press artefacts, and a theme modifier with export helpers.

URL <https://github.com/kinelhu/gghalftone>,
<https://kinelhu.github.io/gghalftone/>

BugReports <https://github.com/kinelhu/gghalftone/issues>

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.1)

Imports ggplot2 (>= 3.4.0), grid, scales, Rcpp, stats, grDevices,
utils

LinkingTo Rcpp

Suggests testthat (>= 3.0.0), ragg, polyclip, magick, maps, mapproj,
patchwork, survival, sf, png, mgcv, MASS, knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

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

Date/Publication 2026-09-29 16:09:30 UTC
RemoteUrl <https://github.com/kinelhu/gghalftone>
RemoteRef HEAD
RemoteSha dec86a926905e2beb9d435b36e8b364db9ab0c17

Contents

cmyk	2
dither	3
draw_key_halftone	4
geom_halftone	5
geom_spot	8
halftone_fields	11
halftone_inks	12
halftone_plot	13
halftone_raster	15
km_steps	15
scale_halftone	16
scale_screen_discrete	17
scale_tone_continuous	18
theme_halftone	19
with_halftone	21
with_halo	23
with_press	24
with_relief	26
Index	29

cmyk	<i>Four-colour process screens</i>
------	------------------------------------

Description

halftone_cmyk() separates an RGB field (from halftone_raster()) into cyan, magenta, yellow and black tone fields; geom_halftone_cmyk() returns the four geom_halftone() layers at the classic screen angles (C 15, M 75, Y 0, K 45) in process inks.

Usage

```
halftone_cmyk(field)

geom_halftone_cmyk(field, pitch = 1, levels = 6, alpha = 0.85, ...)
```

Arguments

field A field with r, g, b columns in $[0, 1]$.
pitch, levels, alpha, ...
 Passed to `geom_halftone()`.

Value

A list of four fields, or a list of four layers to add to a plot.

dither	<i>Threshold matrices and dithering</i>
--------	---

Description

The threshold matrices behind `levels = k` quantisation. `bayer_matrix()` is the ordered-dither matrix of size `n` (a power of two); `blue_noise_matrix()` is a void-and-cluster matrix (cached, seeded) whose thresholds are evenly spread with no periodic structure. `dither_bayer()` and `dither_blue_noise()` quantise a tone matrix `z` in $[0, 1]$ to `levels` steps and dither the remainder with the tiled matrix.

Usage

```
bayer_matrix(n = 4)

blue_noise_matrix(n = 32, sigma = 1.5)

dither_blue_noise(z, levels = 1, n = 32)

dither_bayer(z, levels = 1, n = 4)
```

Arguments

`n` Matrix size (Bayer: 2, 4, 8, ...; blue noise: 32 by default).
`sigma` Gaussian width, in cells, of the energy filter used to build the blue-noise matrix.
`z` Numeric matrix of tone in $[0, 1]$.
`levels` Number of quantisation steps.

Details

Use Bayer for graded tone and blue noise for a binary stipple (`levels = 1`). You rarely need to call these functions directly. Pass `levels` and `algorithm` to `geom_halftone()` or `with_halftone()` instead. By default, tone is continuous and no dithering takes place.

Value

A numeric matrix of thresholds in $(0, 1)$ (matrices), or a quantised tone matrix (dither functions).

References

Bayer, B. E. (1973). An optimum method for two-level rendition of continuous-tone pictures. IEEE International Conference on Communications, 26, 11-15. Floyd, R. W., and Steinberg, L. (1976). An adaptive algorithm for spatial greyscale. Proceedings of the Society for Information Display, 17(2), 75-77. Ulichney, R. (1993). The void-and-cluster method for dither array generation. Proceedings of SPIE, 1913, 332-343. doi:10.1117/12.152707

Examples

```
bayer_matrix(2)
range(blue_noise_matrix(32))
```

draw_key_halftone	<i>Legend keys</i>
-------------------	--------------------

Description

draw_key_halftone() draws a mid-tone swatch at the layer's screen angle and shape: dots, or strips for a line screen. The key is a sample of the screen itself, run over a key-sized area by the code that draws the panel, so it reads at the density of the fill it stands for. draw_key_spot() draws a disc at the break's tone or, for a size legend, at the break's radius. Both are the default keys of the corresponding geoms. They are exported for use with key_glyph.

Usage

```
draw_key_halftone(data, params, size)
```

```
draw_key_spot(data, params, size)
```

Arguments

data	A single row data frame containing the scaled aesthetics to display in this key
params	A list of additional parameters supplied to the geom.
size	Width and height of key in mm.

Value

A grob.

geom_halftone

*Halftone screen of a gridded field***Description**

Draws a gridded field (x, y and a value) as a halftone. Dots or hatch lines are placed at draw time on a lattice with a physical pitch in millimetres. The area of each dot follows the tone at that point. The screen is the same whether the figure is saved at 89 mm or 183 mm. Colour and fill scales apply to the field as usual, and every dot inherits the colour of the cell it samples.

Usage

```
geom_halftone(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  pitch = NULL,
  angle = NULL,
  grid = "hex",
  levels = NULL,
  algorithm = "bayer",
  bayer_n = 4,
  dot_max = 0.9,
  range = NULL,
  shape = "circle",
  gamma = 1,
  overlap = "overprint",
  blend = "alternate",
  tone_max = NULL,
  min_feature = 0.09,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)

## S3 method for class 'halftone'
makeContent(x)
```

Arguments

mapping	Set of aesthetic mappings created by <code>aes()</code> . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options:

If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.

A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.

A function will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(.x, 10)`).

<code>stat</code>	The statistical transformation to use on the data for this layer. When using a <code>geom_*()</code> function to construct a layer, the <code>stat</code> argument can be used to override the default coupling between geoms and stats. The <code>stat</code> argument accepts the following: • A Stat ggproto subclass, for example <code>StatCount</code>. • A string naming the stat. To give the stat as a string, strip the function name of the <code>stat_</code> prefix. For example, to use <code>stat_count()</code>, give the stat as <code>"count"</code>. • For more information and other ways to specify the stat, see the layer stat documentation.
<code>position</code>	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following: • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the <code>position_</code> prefix. For example, to use <code>position_jitter()</code>, give the position as <code>"jitter"</code>. • For more information and other ways to specify the position, see the layer position documentation.
<code>...</code>	Other arguments passed to <code>ggplot2::layer()</code> , such as fixed aesthetics (<code>colour = "black"</code>).
<code>pitch</code>	Lattice spacing in mm; <code>NULL</code> means 0.35. Journal figures want 0.3 to 0.45. Coarser pitches suit posters.
<code>angle</code>	Rotation of the lattice in degrees. <code>NULL</code> picks the default: 45 on a square lattice and for line screens, 15 for hex dots, so that no lattice axis is horizontal or vertical. When screen is mapped, the screen specs are absolute and angle (if given) is added to them.
<code>grid</code>	<code>"hex"</code> (default) or <code>"square"</code> . A 45-degree square lattice is the classic map and photo screen.
<code>levels</code>	<code>NULL</code> for continuous tone (dot area follows tone exactly). An integer quantises tone to that many steps and dithers the remainder with <code>algorithm</code> ; <code>levels = 1</code> is a binary stipple.
<code>algorithm</code>	Dither used when <code>levels</code> is set: <code>"bayer"</code> (graded tone), <code>"blue_noise"</code> (stipple) or <code>"floyd_steinberg"</code> (photographs).
<code>bayer_n</code>	Size of the Bayer matrix when <code>algorithm = "bayer"</code> .

dot_max	Diameter of a full-tone dot as a fraction of pitch (0.9). Above 1 dots merge; it is the ink weight of the screen at 100% tone. For line screens it is the full-tone strip width, as a fraction of pitch.
range	Value range mapped to tone 0..1 when z is used; NULL uses the data range. A wider range lightens the screen.
shape	"circle", "square", "diamond" (area-matched, so a mixed-shape screen stays in one register) or "line" for a line screen whose strip width follows tone.
gamma	Tone curve: tone is raised to this power before printing. Below 1 lifts mid-tones, above 1 deepens them.
overlap	How groups sharing the panel combine: "overprint" (woven on one lattice, default), "interleave" (each group on its own phase-shifted lattice) or "stack" (last group drawn wins; hides overlaps).
blend	Colour of a cell carrying several inks under "overprint". "alternate" (default) weaves them, so a reader can still see which inks are present. "multiply" is the subtractive physics of real ink, the product of the reflectances, so cyan over magenta over yellow goes black. "mix" is a darkened average, which is decorative rather than physical, and it collapses every overlap to one colour.
tone_max	Tone ceiling in [0, 1]. NULL means 1, or 0.55 for a binary stipple.
min_feature	Smallest printable feature in mm (0.09, i.e. 0.25 pt, the minimum line weight in journal artwork guidelines; see References). A cell whose dot would be smaller prints at the floor with probability tone/floor, so coverage is preserved and light tone becomes sparse minimum dots or broken hairlines. Hatch strips are never thinner. Set to 0 to disable.
na.rm	Remove missing values silently.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. annotation_borders() .
x	A half-tone grob (internal; makeContent method).

Details

You can give the value in two ways. `aes(z = )` is normalised to [0, 1] inside the geom, over range or the data range. `aes(tone = )` goes through [scale_tone_continuous\(\)](#), which adds a legend. Groups that share the panel (through colour, fill or group) are printed on one lattice by default, so every ink stays visible where groups overlap.

Value

A `ggplot2` layer.

Defaults

A 0.35 mm hex lattice (73 lines per inch) rotated 15 degrees, so that no lattice axis is horizontal or vertical (a square lattice, and any line screen, goes to 45). Continuous tone and circular dots. Coarser pitches print as a dot pattern rather than as tone; 0.6 mm suits a poster. A binary stipple (levels = 1) is capped at 55% tone so that the densest region remains a stipple. Set tone_max to override.

References

Nature Portfolio. Formatting guide: figures. <https://www.nature.com/nature/for-authors/formatting-guide> Elsevier. Artwork and media instructions. <https://www.elsevier.com/about/policies-and-standards/author/artwork-and-media-instructions>

See Also

[with_halftone\(\)](#) to screen the fill of an existing layer, [geom_spot\(\)](#) for per-point discs, [scale_screen_discrete\(\)](#) for colour-free encodings, [with_halo\(\)](#) for lines drawn over a screen.

Examples

```
vol <- data.frame(expand.grid(x = seq_len(ncol(volcano)), y = seq_len(nrow(volcano))),
  z = as.vector(t(volcano)))
ggplot2::ggplot(vol, ggplot2::aes(x, y, z = z)) + geom_halftone() +
  ggplot2::theme_bw() + theme_halftone()
# engraving: a line screen, plus contours with a paper halo
ggplot2::ggplot(vol, ggplot2::aes(x, y, z = z)) + geom_halftone(shape = "line", angle = 30) +
  with_halo(ggplot2::geom_contour(colour = "black", linewidth = 0.2), width = 0.15) +
  ggplot2::theme_bw() + theme_halftone()
```

geom_spot

Tone discs

Description

Each point becomes a disc of radius *r* mm (or `aes(size = )`, in mm) filled with a halftone whose tone is the point's value. Use it for a dot plot in which a second quantity is shown by ink density instead of a colour ramp. Each disc has its own centred hex lattice, clipped to the disc, and a ring in the disc's colour.

Usage

```
geom_spot(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  r = 3,
```

```

    pitch = 0.35,
    levels = NULL,
    bayer_n = 4,
    dot_max = 0.9,
    range = NULL,
    ring = TRUE,
    ring_lwd = 0.3,
    min_feature = 0.09,
    angle = NULL,
    shape = "circle",
    na.rm = FALSE,
    show.legend = NA,
    inherit.aes = TRUE
  )

  ## S3 method for class 'spot'
  makeContent(x)

```

Arguments

mapping	Set of aesthetic mappings created by aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to ggplot(). A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
stat	The statistical transformation to use on the data for this layer. When using a <code>geom_*()</code> function to construct a layer, the <code>stat</code> argument can be used to override the default coupling between geoms and stats. The <code>stat</code> argument accepts the following: • A <code>Stat</code> ggproto subclass, for example <code>StatCount</code>. • A string naming the stat. To give the stat as a string, strip the function name of the <code>stat_</code> prefix. For example, to use <code>stat_count()</code>, give the stat as <code>"count"</code>. • For more information and other ways to specify the stat, see the layer stat documentation.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following: • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position.

	• A string naming the position adjustment. To give the position as a string, strip the function name of the position_ prefix. For example, to use <code>position_jitter()</code>, give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
...	Other arguments passed to <code>ggplot2::layer()</code> , such as fixed aesthetics (<code>colour = "black"</code>).
r	Disc radius in mm when size is not mapped.
pitch	Lattice spacing inside the discs, in mm.
levels	NULL for continuous tone (dot area follows tone exactly). An integer quantises tone to that many steps and dithers the remainder with <code>algorithm</code> ; <code>levels = 1</code> is a binary stipple.
bayer_n	Size of the Bayer matrix when <code>algorithm = "bayer"</code> .
dot_max	Diameter of a full-tone dot as a fraction of pitch (0.9). Above 1 dots merge; it is the ink weight of the screen at 100% tone. For line screens it is the full-tone strip width, as a fraction of pitch.
range	Value range mapped to tone 0..1 when z is used; NULL uses the data range. A wider range lightens the screen.
ring	Draw the disc outline.
ring_lwd	Line width of the ring.
min_feature	Smallest printable feature in mm (0.09, i.e. 0.25 pt, the minimum line weight in journal artwork guidelines; see References). A cell whose dot would be smaller prints at the floor with probability tone/floor, so coverage is preserved and light tone becomes sparse minimum dots or broken hairlines. Hatch strips are never thinner. Set to 0 to disable.
angle	Rotation of the rosette in degrees; NULL means 15 for dots and 45 for hatching. Added to a mapped screen spec's angle only when given.
shape	"circle", "square", "diamond" or "line" (hatched discs, for one-ink dot plots). <code>aes(screen =)</code> with <code>scale_screen_discrete()</code> varies angle and shape per group.
na.rm	Remove missing values silently.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. <code>annotation_borders()</code> .
x	A spot grob (internal; <code>makeContent</code> method).

Details

Tone comes from `aes(tone = )` through `scale_tone_continuous()`, which gives it a legend of discs at the breaks, or from `aes(z = )` normalised inside the geom.

Value

A ggplot2 layer.

Examples

```
d <- expand.grid(gene = c("A", "B", "C"), cluster = 1:4); d$expr <- runif(12); d$pct <- runif(12)
ggplot2::ggplot(d, ggplot2::aes(cluster, gene, tone = expr, size = pct)) + geom_spot() +
  scale_tone_continuous() + ggplot2::scale_radius(range = c(1, 2.2)) +
  ggplot2::theme_minimal() + theme_halftone()
```

halftone_fields

Fields from statistical objects

Description

Helpers that turn common objects into gridded fields (x, y, z) for `geom_halftone()`. Most figures do not need them, because `with_halftone()` screens the fill of a ribbon, area, bar or polygon layer directly. Use them when the tone field is computed rather than drawn: an estimate with a tent or gaussian profile (`halftone_band()`), bars that fade towards the axis (`halftoneBars()`), map regions rasterised by point-in-region lookup (`halftone_regions()`, requires the maps package), or ridgelines (`halftone_ridges()`).

Usage

```
halftone_band(
  x,
  est,
  lo,
  hi,
  ny = 250,
  profile = c("tent", "gauss", "flat"),
  ylim = NULL,
  keep = NULL
)
```

```
halftoneBars(
  cat,
  height,
  width = 0.8,
  nx = 25,
  ny = 120,
  fade = 0.35,
  extra = NULL
)
```

```
halftone_regions(db = "state", values, res = 0.25, ...)
```

```
halftone_ridges(values, groups, scale = 1.6, n = 256, fade = 0.35, bw = "nrd0")
```

Arguments

- x, est, lo, hi Positions, estimate and interval limits.
- ny, nx Field resolution.
- profile Tone profile across the band.
- ylim, keep Vertical extent, and extra columns (one row per x) carried into the field.
- cat, height, width, fade, extra
 Bar categories, heights, width, tone at the base, extra columns.
- db, values, res, ...
 Map database, named values per region, cell size in degrees, arguments to maps::map().
- groups, scale, n, bw
 Ridgeline groups, height scale, resolution, bandwidth for stats::density().

Value

A data frame (halftone_ridges()): a list of fields, lines, levels).

halftone_inks	<i>Inks, paper, ramp and column widths</i>
---------------	--

Description

Named constants shared by the geoms and [theme_halftone\(\)](#).

Usage

- halftone_inks
- halftone_paper
- halftone_ink
- halftone_widths
- halftone_process
- halftone_ramp

Format

Character vectors of hex colours, or a named numeric vector of widths.

Details

- `halftone_inks`: the six-ink palette (red, blue, ochre, green, violet, grey). On ggplot2 4.0 and later the theme sets it as the default palette for mapped colour and fill. It stops at six, which is already more inks than a press would use. Beyond six groups ggplot2 warns and the extra groups get no fill, so use `scale_screen_discrete()`, facets, or your own palette instead.
- `halftone_process`: press colours, each made of one or two process plates at 100% (K, M+Y, C+M, C+Y, C, M).
- `halftone_ramp`: the default continuous ramp (paper, ochre, red, near-black).
- `halftone_paper`, `halftone_ink`: a cream paper colour and a near-black ink colour, for plates.
- `halftone_widths`: journal column widths in mm.

halftone_plot

Screen a whole plot

Description

Takes a finished plot and returns it printed: every filled layer screened, a paper hairline under every line that crosses a screen, and the halftone theme modifier on top. One call, where `with_halftone()` and `with_halo()` would otherwise be composed layer by layer.

Usage

```
halftone_plot(plot, halo = 0.09, theme = TRUE, ...)
```

Arguments

<code>plot</code>	A ggplot, or a patchwork of them. The object handed in is left alone.
<code>halo</code>	Width in mm of the paper hairline drawn under lines that cross a screen, the printer's knockout channel. 0 turns it off.
<code>theme</code>	Add <code>theme_halftone()</code> : paper ground, no gridlines under the screen, screen-sized legend keys and the ink palette. FALSE keeps the plot's own theme.
<code>...</code>	Passed to <code>with_halftone()</code> for every layer it screens, for example pitch, shape, grid, levels or dot_max.

Value

The plot, with its layers wrapped.

What it does to each layer

The treatment follows the geom.

- **Screened:** ribbons, areas, densities, bars, columns, histograms, tiles, rectangles, polygons, sf geometries, violins, boxplots, crossbars and smooths. Each gets the tone profile `with_halftone()` picks for it, so bars and maps come out flat, intervals follow the likelihood of the estimate, and densities get a soft vignette.
- **Haloed:** lines, paths, steps, contours, segments and points, but only where a screened layer sits underneath them. Anything drawn before the first screen has nothing to stay legible against and is left alone.
- **Left alone:** text, labels, error bars, rugs, reference lines, rasters, and any geom not listed above.

A layer you wrapped yourself is left as you wrapped it, so `halftone_plot()` can be applied to a plot that is already part screened, and applying it twice changes nothing the second time.

What it does not do

It screens the fill and chooses no colours. Line weights, point shapes and any colour you set or mapped stay as the plot set them, so a figure built for the screen from the start still looks better: see the package README for the pair.

A colour you did not choose is a different matter, and it belongs to the theme rather than here. On ggplot2 4.0 an unmapped geom colour comes from `theme(geom = )`, whose stock accent is the blue `geom_smooth()` draws its line in. `theme_halftone()` sets that accent to the package's ink, so the one colour in a screened figure that would sit outside the register comes into it. `theme = FALSE` keeps your own theme, and that with it.

`geom_raster()` draws an image rather than polygons, so there is nothing to clip a screen to and the layer prints as it was. Use `geom_tile()` instead, or `geom_halftone()` on the field.

See Also

`with_halftone()` to screen one layer, `with_press()` to print the result.

Examples

```
m <- loess(dist ~ speed, cars, span = 0.9)
nd <- data.frame(speed = seq(4, 25, length.out = 60))
pr <- predict(m, nd, se = TRUE)
nd$fit <- pr$fit; nd$lo <- pr$fit - 1.96 * pr$se.fit; nd$hi <- pr$fit + 1.96 * pr$se.fit
p <- ggplot2::ggplot(nd, ggplot2::aes(speed)) +
  ggplot2::geom_ribbon(ggplot2::aes(ymin = lo, ymax = hi), fill = "steelblue") +
  ggplot2::geom_line(ggplot2::aes(y = fit))
halftone_plot(p)
halftone_plot(p, pitch = 0.6, shape = "line")
```

halftone_raster	<i>Image to field</i>
-----------------	-----------------------

Description

Reads an image from a path or a magick image object and returns a field data frame (x, y, z and r, g, b) for `geom_halftone()` or `geom_halftone_cmyk()`. When `invert = TRUE`, dark pixels get high tone. Requires the magick package.

Usage

```
halftone_raster(
  img,
  max_px = 160,
  channel = c("luminance", "red", "green", "blue"),
  invert = TRUE
)
```

Arguments

<code>img</code>	A file path or a magick-image.
<code>max_px</code>	Longest side of the resampled image in cells.
<code>channel</code>	Which channel becomes z.
<code>invert</code>	Map dark to high tone.

Value

A data frame.

km_steps	<i>Kaplan-Meier frames for halftone survival plots</i>
----------	--

Description

Convert a `survival::survfit` object into data frames for `with_halftone(geom_ribbon())` and `with_halo(geom_step())`. `km_steps()` returns the step outline with one row per corner (time, surv, lo, hi, strata), starting at (0, 1), so that a ribbon between lo and hi follows the steps exactly. `km_censor()` returns the censor marks (time, surv, strata). `km_risk()` returns the number at risk at times. Strata labels are the level values without the variable name; "ph.ecog=1" becomes "1".

Usage

```
km_steps(fit, conf = TRUE)
```

```
km_censor(fit)
```

```
km_risk(fit, times)
```

Arguments

<code>fit</code>	A survfit object.
<code>conf</code>	Include the confidence limits (lo, hi); otherwise both equal surv.
<code>times</code>	Times at which to report the number at risk.

Value

A data frame.

References

Loprinzi, C. L., Laurie, J. A., Wieand, H. S., et al. (1994). Prospective evaluation of prognostic variables from patient-completed questionnaires. North Central Cancer Treatment Group. *Journal of Clinical Oncology*, 12(3), 601-607. doi:10.1200/JCO.1994.12.3.601. Distributed as `survival::lung`.

Examples

```
if (requireNamespace("survival", quietly = TRUE)) {
  fit <- survival::survfit(survival::Surv(time, status) ~ sex, data = survival::lung)
  s <- km_steps(fit)
  ggplot2::ggplot(s, ggplot2::aes(time, group = strata)) +
    with_halftone(ggplot2::geom_ribbon(ggplot2::aes(ymin = lo, ymax = hi, fill = strata))) +
    with_halo(ggplot2::geom_step(ggplot2::aes(y = surv, colour = strata))) +
    ggplot2::theme_classic() + theme_halftone()
}
```

scale_halftone	<i>Ink palette scales</i>
----------------	---------------------------

Description

Manual colour and fill scales over [halftone_inks](#). On ggplot2 4.0 and later, `theme_halftone()` sets the inks as the default palette. Use these scales with older ggplot2 versions or without the modifier.

Usage

```
scale_colour_halftone(...)
```

```
scale_fill_halftone(...)
```

Arguments

... Passed to `ggplot2::scale_colour_manual()` / `ggplot2::scale_fill_manual()`.

Value

A ggplot2 scale.

scale_screen_discrete *Screen scales: colour-free encodings*

Description

Map a discrete variable to the screen aesthetic of `geom_halftone()` and of any layer wrapped in `with_halftone()`. A screen specification is a string "angle|shape|tone|line_angle": the lattice angle in degrees, the dot shape (circle, square, diamond, or line to hatch that group), a tone multiplier (0.5 prints half the ink, which is how an ordinal scale is drawn in one ink), and the hatch angle used when the layer is a line screen. A bare number is an angle. Specifications are absolute. If the layer has an angle, it is added.

Usage

```
scale_screen_discrete(..., grid = c("hex", "square"), name = waiver())
```

```
scale_screen_manual(values, ..., name = waiver())
```

Arguments

... Passed to `ggplot2::discrete_scale()`.

grid Lattice of the layer, which sets the angle period.

name The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If NULL, the legend title will be omitted.

values Character or numeric vector of screen specs, one per level.

Details

`scale_screen_discrete()` uses a recipe in which consecutive screens differ in both angle and shape: three angles spaced over the lattice period (60 degrees for hex), and hatch angles 45, 135, 0, 90 and so on for line screens. `scale_screen_manual()` takes your own specifications, for example `c("45|line", "135|line", "15|circle")` to mix hatching and dots in one layer. Angle alone distinguishes three screens. For more groups, vary shape and tone, or add a second ink.

Value

A ggplot2 scale.

Examples

```
d <- data.frame(g = c("a", "b", "c"), n = c(3, 2, 1))
ggplot2::ggplot(d, ggplot2::aes(g, n, screen = g)) +
  with_halftone(ggplot2::geom_col(fill = "black", colour = "black"), shape = "line") +
  scale_screen_discrete(name = NULL) + ggplot2::theme_classic() + theme_halftone()
```

scale_tone_continuous *Tone scale*

Description

Maps a continuous variable to the tone aesthetic of `geom_spot()` and `geom_halftone()`: ink density in $[0, 1]$. The legend shows discs (or swatches) at the scale breaks. `scale_tone()` is an alias.

Usage

```
scale_tone_continuous(name = waiver(), ..., range = c(0, 1), guide = "legend")
```

```
scale_tone(name = waiver(), ..., range = c(0, 1), guide = "legend")
```

Arguments

name	The name of the scale. Used as the axis or legend title. If <code>waiver()</code> , the default, the name of the scale is taken from the first mapping used for that aesthetic. If <code>NULL</code> , the legend title will be omitted.
...	Passed to <code>ggplot2::continuous_scale()</code> (limits, breaks, labels, trans, ...).
range	Output tone range; narrow it (e.g. <code>c(0.1, 0.9)</code>) to keep the lightest value visible.
guide	A function used to create a guide or its name. See <code>guides()</code> for more information.

Value

A `ggplot2` scale.

theme_halftone	<i>Theme modifier and export helpers</i>
----------------	--

Description

theme_halftone() is an incomplete theme to add on top of your own theme, for example theme_classic() + theme_halftone(). It changes only what a halftone needs and leaves fonts, sizes and axes to the theme it is added to:

Usage

```
theme_halftone(paper = "white", palette = c("inks", "process", "none"))
```

```
ggsave_journal(
  filename,
  plot,
  width = c("double", "single", "onehalf"),
  height = 100,
  dpi = 600,
  format = NULL,
  bg = "white",
  ...
)
```

```
halftone_proof(
  plot,
  width = "single",
  height = 62,
  dir = tempdir(),
  centre = c(0.5, 0.5),
  size = 20,
  zoom = 4,
  dpi = 600,
  bg = "white"
)
```

Arguments

paper	Background colour of the plot and panel. Use halftone_paper for cream stock.
palette	Default discrete palette on ggplot2 4.0 and later: "inks" (halftone_inks), "process" (halftone_process , press colours of one or two plates) or "none" to leave the palettes alone.
filename, plot, dpi, ...	Passed to ggplot2::ggsave() .
width	"single" (89 mm), "onehalf" (120 mm), "double" (183 mm) or a width in mm.

height	Height in mm.
format	"png", "tiff" or "pdf"; NULL takes it from the extension of filename.
bg	Device background. Set it to the paper colour when the plot has a fixed aspect ratio, because the device shows beside the panel.
dir, centre, size, zoom	For <code>halftone_proof()</code> : output directory, the crop centre as a fraction of width and height, the crop size in mm, and the magnification.

Details

- paper ground: plot and panel backgrounds in paper, no gridlines, no panel border fill;
- legend keys large enough to show a screen (6 by 4 mm), no key background;
- on ggplot2 4.0 and later, `halftone_inks` (or `halftone_process`) as the default discrete palette and `halftone_ramp` as the default continuous palette. It also sets the geom accent, which on ggplot2 4.0 is where an unmapped geom colour comes from: the stock accent is the blue `geom_smooth()` draws its line in, and left alone it is the one colour in a screened figure that sits outside the register. `halftone_inks` holds six inks and `halftone_ramp` four stops; past six groups, map `scale_screen_discrete()` instead of colour, or pass `palette = "none"` and set your own.

`ggsave_journal()` saves at a journal column width in mm. The file extension selects the format: PNG or TIFF at 600 dpi through `ragg`, or vector PDF through `cairo_pdf()`. Most journals prefer vector files for line art, and in a vector file every dot is a path at physical size. `halftone_proof()` renders a plot at final size and a magnified crop of one region (by default the panel centre at 4x) and returns the two file paths.

Value

A ggplot2 theme; `ggsave_journal()` is called for its side effect.

References

Nature Portfolio. Formatting guide: figures. <https://www.nature.com/nature/for-authors/formatting-guide> Elsevier. Artwork and media instructions. <https://www.elsevier.com/about/policies-and-standards/author/artwork-and-media-instructions>

Examples

```
ggplot2::ggplot(mtcars, ggplot2::aes(wt, mpg)) + ggplot2::geom_point() +
  ggplot2::theme_classic() + theme_halftone()
```

with_halftone

*Screen the fill of any layer***Description**

Wraps a ggplot2 layer so that its fill is drawn as a halftone. The wrapped geom draws as usual. At draw time, every filled polygon, rectangle or path in its grob tree is rasterised onto a millimetre lattice, given a tone field derived from its geometry, and drawn as dots or hatch in its fill colour, clipped to the shape. Works with ribbons, areas, bars and columns, densities and violins, polygons, tiles and geom_sf().

Usage

```
with_halftone(
  layer,
  pitch = 0.35,
  angle = NULL,
  grid = "hex",
  tone = NULL,
  profile = c("vertical", "radial"),
  levels = NULL,
  bayer_n = 4,
  dot_max = 0.9,
  gamma = 1,
  tone_max = NULL,
  outline = TRUE,
  shape = "circle",
  algorithm = "bayer",
  clip = TRUE,
  overlap = c("overprint", "stack"),
  level = 0.95,
  min_feature = 0.09,
  redundant = NULL
)

## S3 method for class 'halftone_fill'
makeContent(x)
```

Arguments

layer	A ggplot2 layer, for example <code>geom_ribbon(aes(ymin = lo, ymax = hi, fill = g))</code> ; a list holding one, which is what <code>geom_sf()</code> returns; or a whole plot or patchwork, in which case every layer in it is wrapped and the object handed in is left alone. Screening a whole plot gives a usable figure in one call, but it screens the fill and nothing else: add <code>with_halo()</code> to the lines that cross a screen and <code>theme_halftone()</code> for the paper ground and the ink palette.
-------	---

pitch	Lattice spacing in mm (0.35, 73 lines per inch). Coarsen deliberately for a poster, or where several hatched groups overlap.
angle	Lattice angle in degrees. NULL picks 45 on a square lattice and for hatching, 15 for hex dots.
grid	"hex" (default) or "square". A 45-degree square lattice is the classic map and photo screen.
tone	Tone profile: NULL (from the geometry, see below), "likelihood", "flat", "vignette", "centre" (the older gaussian, within a hair of likelihood), "edge", "tent" or "centre-soft".
profile	Distance used by the non-flat profiles. "vertical" measures to the top and bottom edge along each column and is normalised per column, so every column reaches full tone at its own middle; this is right for ribbons and densities. "radial" measures to the nearest point of the outline and is normalised over the whole shape, so a narrow arm stays lighter than the body.
levels	NULL for continuous tone (dot area follows tone exactly). An integer quantises tone to that many steps and dithers the remainder with algorithm; levels = 1 is a binary stipple.
bayer_n	Size of the Bayer matrix when algorithm = "bayer".
dot_max	Diameter of a full-tone dot as a fraction of pitch (0.9). Above 1 dots merge; it is the ink weight of the screen at 100% tone. For line screens it is the full-tone strip width, as a fraction of pitch.
gamma	Tone curve: tone is raised to this power before printing. Below 1 lifts mid-tones, above 1 deepens them.
tone_max	Tone ceiling; NULL picks the register above.
outline	Keep the layer's own outline (with its fill removed) on top of the screen.
shape	"circle", "square", "diamond" (area-matched, so a mixed-shape screen stays in one register) or "line" for a line screen whose strip width follows tone.
algorithm	Dither used when levels is set: "bayer" (graded tone), "blue_noise" (stipple) or "floyd_steinberg" (photographs).
clip	Clip the screen to the exact fill region (grid clipping path; ragg, cairo and pdf honour it).
overlap	"overprint" weaves all inks present in a cell (default); "stack" lets the last-drawn shape win, for nested intervals and ridgelines.
level	Confidence level the ribbon represents, for the "likelihood" profile.
min_feature	Smallest printable feature in mm (0.09, i.e. 0.25 pt, the minimum line weight in journal artwork guidelines; see References). A cell whose dot would be smaller prints at the floor with probability tone/floor, so coverage is preserved and light tone becomes sparse minimum dots or broken hairlines. Hatch strips are never thinner. Set to 0 to disable.
redundant	Also give each fill group its own screen (angle and shape), so colour is never the only encoding. NULL means yes for geoms whose groups tile the plane (bars, areas, polygons, tiles, sf) and no for intervals and densities, whose overlapping groups are woven on one lattice; separate lattices there moire.
x	A halftone_fill grob (internal; makeContent method).

Value

The layer, with its geom replaced by a halftone-drawing subclass.

Tone profile

With tone = NULL, the profile depends on the geometry:

- bars, columns, tiles, areas, polygons and sf: "flat", because for those the interior carries the value;
- ribbons (intervals): "likelihood", the normal density of the estimate across the interval: 1 on the estimate, 0.146 at a 95% limit (level);
- densities and violins: "vignette", a soft fade towards the outline, so that overlapping groups stay legible;
- line screens: always "flat", because a tapered hatch looks like fringe;
- a mapped screen: always "flat", because it is a categorical pattern.

Register

tone_max = NULL sets one tone ceiling across figure types: flat 0.45 (polygons and sf 0.6), centre 0.6, vignette 0.7, hatching 0.4, hatched intervals a hairline (the strip width that equals min_feature). Alpha on the fill is folded into tone: a fill with 30% alpha prints as a 30% screen, and the output holds no partial transparency.

Examples

```
x <- seq(0, 10, length.out = 60)
d <- data.frame(x, y = sin(x), lo = sin(x) - 0.5, hi = sin(x) + 0.5)
ggplot2::ggplot(d, ggplot2::aes(x)) +
  with_halftone(ggplot2::geom_ribbon(ggplot2::aes(ymin = lo, ymax = hi),
                                     fill = halftone_inks[["blue"]])) +
  with_halo(ggplot2::geom_line(ggplot2::aes(y = y), colour = halftone_inks[["blue"]])) +
  ggplot2::theme_classic() + theme_halftone()
```

with_halo

Paper halo under a line or point layer

Description

Draws the layer twice: first in paper colour with width mm added to each side of every stroke, then as is. Use it for a line that crosses a dot field, such as a step curve over a screened confidence band or contours over an elevation field. Keep the width small: 0.09 mm (0.25 pt, the printable minimum) against dots, 0.15 mm against a line screen. A wider halo looks like a second line.

Usage

```
with_halo(layer, width = 0.09, colour = "white")
```

Arguments

layer	A ggplot2 layer drawing lines, paths, steps, contours or points; a list holding one; or a whole plot or patchwork, in which case every layer in it is wrapped. The object handed in is left alone.
width	Halo width in mm on each side of the stroke.
colour	Halo colour. Use the paper colour of the plot; the default is white.

Value

The layer, with its geom replaced by a halo-drawing subclass.

Examples

```
ggplot2::ggplot(mtcars, ggplot2::aes(wt, mpg)) + with_halo(ggplot2::geom_line())
```

with_press	<i>Press artefacts</i>
------------	------------------------

Description

Wraps a halftone layer so that its screen is drawn the way a press would put it on paper rather than the way the plate describes it. Compose it around [with_halftone\(\)](#) or a halftone geom: `with_press(with_halftone(geom_col()), gain = 0.3)`.

Usage

```
with_press(
  layer,
  gain = 0.2,
  slur = 0,
  slur_angle = 90,
  fillet = 0,
  mottle = 0,
  mottle_scale = 4,
  registration = 0,
  seed = NULL
)
```

Arguments

layer	A halftone layer, a list holding one, or a whole plot. Given a plot, every layer in it is wrapped, each with its own seed, so the inks miss each other and mottle independently the way separate plates do.
gain	Tone value increase at a 50% screen. 0 leaves the screen alone.

slur	Length of the smear, as a fraction of the lattice pitch, in the direction the sheet travelled. 0.1 is a press running a little fast, 0.3 a visible fault. 0 leaves the dots round.
slur_angle	Direction of that smear in degrees, measured anticlockwise from the x axis.
fillet	Radius of the ink bridge where two dots meet, as a fraction of the lattice pitch. Useful values are small, and they act with gain: at gain = 0.25, 0.02 rounds the cusp, 0.04 draws a clear bridge and blots the last eighth of the tone range, and 0.08 pulls the fill-in down into the midtones. Much above that the closing swallows the gaps and the shadows go flat. 0 leaves the cusp where the circles cross. Needs the polyclip package.
mottle	Relative standard deviation of the slow variation in ink density across the sheet. 0.1 is a visible but unremarkable impression, 0.25 a poor one.
mottle_scale	Distance in mm over which that variation changes. Real mottle runs at a few millimetres.
registration	Standard deviation in mm of this layer's plate offset. 0 is perfect registration.
seed	Seed for the plate offset, so a figure rebuilds identically. NULL draws a new one.

Value

The layer, with its geom replaced by one that carries the press settings.

Dot gain

Ink spreads into paper, so a printed dot is larger than its plate. The trade measures this as tone value increase, the extra coverage at a 50% screen: roughly 0.10 to 0.20 for offset on coated stock and 0.25 to 0.35 on newsprint. gain is that number.

It applies to coverage, the fraction of paper the screen inks, which is what a densitometer reads and is not the same as the tone the screen was asked for. At the default ink weight a full-tone cell covers about three quarters of its lattice cell, not all of it. The increase follows $\sin(\pi * \text{coverage})$, so it vanishes at bare paper and at a covered sheet and peaks where a press gains most. Coverage is capped at the sheet, and that cap is what fills a shadow in. Above about gain = 0.2 the dark dots grow past the pitch and meet, printing as solid with pinholes.

Mottle

mottle adds a slow random variation in ink density across the sheet, smooth at the scale of mottle_scale millimetres, which is what separates a real impression from a clean digital screen. It varies tone, so it survives resizing like everything else here.

Slur

A sheet moving under the plate smears each dot along its direction of travel, so the dot prints as a capsule rather than a circle. slur is the length of that smear as a fraction of the pitch and slur_angle its direction. It is drawn as the true swept shape, not as a second faint impression: the output holds no partial transparency.

Ink bridges

Where two dots overlap, the circles cross in a sharp concave cusp. Wet ink does not: surface tension pulls a fillet across the notch, so the boundary stays convex. `fillet` is the radius of that bridge as a fraction of the pitch, applied as a morphological closing of the union of the overlapping dots, which rounds concave corners and leaves convex boundaries alone. Only dots close enough to reach a neighbour are processed, so a figure pays for it in its shadows. A bridge is ink, so it darkens the shadows on top of gain.

Ink cannot bridge dots that do not meet, and at the default ink weight (`dot_max = 0.9`) a full-tone dot still stands a tenth of a pitch clear of its neighbour. So `fillet` needs something to work with: either gain above about 0.2, which closes that gap in the shadows, or a heavier plate. On its own it changes nothing below the top of the tone range.

What the bridge changes is the shape of the white interstices between dots, which are features of the pitch. At 0.35 mm they are too small to read as shapes and the fillet arrives as ink weight alone. Use it where the screen already reads as dots: an editorial plate at 0.6 mm and up.

It needs the `polycip` package. It is also the one setting here with a measurable cost: about a second per twenty thousand touching dots.

Registration

Each ink is a separate plate and the plates never align perfectly. `registration` is the standard deviation, in mm, of a random offset applied to this layer's lattice. It only shows when a figure is built from several halftone layers, one per ink, as `geom_halftone_cmyk()` does. Values around 0.05 mm read as a good press, 0.2 mm as a cheap one.

Examples

```
d <- data.frame(x = seq(0, 10, length.out = 60))
d$y <- sin(d$x); d$lo <- d$y - 0.5; d$hi <- d$y + 0.5
ggplot2::ggplot(d, ggplot2::aes(x)) +
  with_press(with_halftone(ggplot2::geom_ribbon(ggplot2::aes(ymin = lo, ymax = hi),
                                                fill = "black")),
            gain = 0.3) +
  ggplot2::theme_classic() + theme_halftone()
```

with_relief

Illuminated contours

Description

Wraps a contour layer, or any path layer, so that each segment is lit or shaded according to the direction of its slope relative to a light source. This is Kitiro Tanaka's illuminated-contour method. Lit segments are drawn in paper colour and shaded segments in ink. Both widen as the slope faces the light more directly. A hairline base contour is drawn under both. Over a `geom_halftone()` field or a hatched engraving, the paper segments cut through the screen and the surface appears in relief.

Usage

```
with_relief(
  layer,
  light = 315,
  width = c(0.05, 0.35),
  colours = c(lit = "white", shade = "black", base = "#8A8A8A"),
  uphill = c("auto", "left", "right")
)

## S3 method for class 'relief'
makeContent(x)
```

Arguments

layer	A <code>geom_contour()</code> , <code>geom_path()</code> or <code>geom_line()</code> layer; a list holding one; or a whole plot or patchwork, in which case every layer in it is wrapped. The object handed in is left alone.
light	Azimuth of the light in degrees, clockwise from north; 315 is upper left.
width	Line width in mm at grazing and at full illumination, <code>c(min, max)</code> .
colours	Named vector: <code>lit</code> (the paper colour of the plot), <code>shade</code> (ink), <code>base</code> (the hairline contour under both; NA for none).
uphill	"auto", "left" or "right" of the path direction.
x	A relief grob (internal; <code>makeContent</code> method).

Details

With `uphill = "auto"`, the uphill side of each contour is inferred from the nearest neighbouring contour at another level. A closed ring with no such neighbour is treated as a summit. For a path that is not a contour, set `uphill` explicitly.

Draw this over a screened field. Lit segments are paper-coloured, so on bare paper they are invisible and only the shaded half of each contour appears.

Value

The layer, with its `geom` replaced by a relief-drawing subclass.

References

Tanaka, K. (1950). The relief contour method of representing topography on maps. *Geographical Review*, 40(3), 444-456. doi:[10.2307/211219](https://doi.org/10.2307/211219)

Examples

```
vol <- data.frame(expand.grid(x = seq_len(ncol(volcano)), y = seq_len(nrow(volcano))),
  z = as.vector(t(volcano)))
ggplot2::ggplot(vol, ggplot2::aes(x, y, z = z)) +
  geom_halftone(shape = "line", colour = "black", angle = 30) +
  with_relief(ggplot2::geom_contour(bins = 10)) + ggplot2::coord_equal(expand = FALSE) +
```

```
ggplot2::theme_bw() + theme_halftone()
```

Index

`aes()`, [5](#), [9](#)
`annotation_borders()`, [7](#), [10](#)

`bayer_matrix` (`dither`), [3](#)
`blue_noise_matrix` (`dither`), [3](#)

`cmk`, [2](#)

`dither`, [3](#)
`dither_bayer` (`dither`), [3](#)
`dither_blue_noise` (`dither`), [3](#)
`draw_key_halfnote`, [4](#)
`draw_key_spot` (`draw_key_halfnote`), [4](#)

`fortify()`, [6](#), [9](#)

`geom_halfnote`, [5](#)
`geom_halfnote()`, [2](#), [3](#), [11](#), [14](#), [15](#), [17](#), [18](#), [26](#)
`geom_halfnote_cmyk` (`cmyk`), [2](#)
`geom_halfnote_cmyk()`, [15](#), [26](#)
`geom_spot`, [8](#)
`geom_spot()`, [8](#), [18](#)
`ggplot()`, [6](#), [9](#)
`ggplot2::continuous_scale()`, [18](#)
`ggplot2::discrete_scale()`, [17](#)
`ggplot2::ggsave()`, [19](#)
`ggplot2::layer()`, [6](#), [10](#)
`ggplot2::scale_colour_manual()`, [17](#)
`ggplot2::scale_fill_manual()`, [17](#)
`ggsave_journal` (`theme_halfnote`), [19](#)
`guides()`, [18](#)

`halfnote_band` (`halfnote_fields`), [11](#)
`halfnote_bars` (`halfnote_fields`), [11](#)
`halfnote_cmyk` (`cmyk`), [2](#)
`halfnote_fields`, [11](#)
`halfnote_ink` (`halfnote_inks`), [12](#)
`halfnote_inks`, [12](#), [16](#), [19](#), [20](#)
`halfnote_paper`, [19](#)
`halfnote_paper` (`halfnote_inks`), [12](#)
`halfnote_plot`, [13](#)

`halfnote_process`, [19](#), [20](#)
`halfnote_process` (`halfnote_inks`), [12](#)
`halfnote_proof` (`theme_halfnote`), [19](#)
`halfnote_ramp`, [20](#)
`halfnote_ramp` (`halfnote_inks`), [12](#)
`halfnote_raster`, [15](#)
`halfnote_raster()`, [2](#)
`halfnote_regions` (`halfnote_fields`), [11](#)
`halfnote_ridges` (`halfnote_fields`), [11](#)
`halfnote_widths` (`halfnote_inks`), [12](#)

`km_censor` (`km_steps`), [15](#)
`km_risk` (`km_steps`), [15](#)
`km_steps`, [15](#)

`layer` `position`, [6](#), [10](#)
`layer` `stat`, [6](#), [9](#)

`makeContent_halfnote` (`geom_halfnote`), [5](#)
`makeContent_halfnote_fill`
 (`with_halfnote`), [21](#)
`makeContent_relief` (`with_relief`), [26](#)
`makeContent_spot` (`geom_spot`), [8](#)

`scale_colour_halfnote` (`scale_halfnote`),
 [16](#)
`scale_fill_halfnote` (`scale_halfnote`), [16](#)
`scale_halfnote`, [16](#)
`scale_screen_discrete`, [17](#)
`scale_screen_discrete()`, [8](#), [10](#), [13](#), [20](#)
`scale_screen_manual`
 (`scale_screen_discrete`), [17](#)
`scale_tone` (`scale_tone_continuous`), [18](#)
`scale_tone_continuous`, [18](#)
`scale_tone_continuous()`, [7](#), [10](#)

`theme_halfnote`, [19](#)
`theme_halfnote()`, [12–14](#), [16](#), [21](#)

`with_halfnote`, [21](#)
`with_halfnote()`, [3](#), [8](#), [11](#), [13](#), [14](#), [17](#), [24](#)

with_halo, [23](#)
with_halo(), [8](#), [13](#), [21](#)
with_press, [24](#)
with_press(), [14](#)
with_relief, [26](#)