Package: paradoxR (via r-universe)

November 20, 2026

Title Colour Palettes for 'ggplot2' and Faster 'Bioconductor' Analysis

Version 26.11.0

Description Provides drop-in replacements for the default colour scales of 'ggplot2', with discrete, continuous, binned, ordinal and diverging scales and one switch for every plot. Also makes 'clusterProfiler', 'GOSemSim', 'DESeq2' and 'sva' faster when loaded: GO and KEGG data are cached on disk, and slow internal steps are replaced by vectorised code. Stores the results of slow 'Seurat' and 'DiffBind' steps and reuses them when the data, input files, arguments and package versions are unchanged; unused cache entries expire after 7 days. Results are identical to the original functions.

License Apache License (>= 2)

URL https://github.com/catch-26/paradoxR

BugReports https://github.com/catch-26/paradoxR/issues

Depends R (>= 4.1.0)

Imports ggplot2 (>= 3.4.0), grDevices, rlang (>= 1.0.0), scales (>= 1.3.0), tools, utils

Suggests AnnotationDbi, clusterProfiler, colorspace, DBI, DESeq2, DiffBind, GO.db, GOSemSim, methods, org.Hs.eg.db, ps, qs2, Rsamtools, S4Vectors, Seurat, SeuratObject, SummarizedExperiment, sva, targets, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

Repository https://catch-26.r-universe.dev

Date/Publication 2026-11-20 20:49:49 UTC

RemoteUrl https://github.com/catch-26/paradoxR

RemoteRef HEAD

RemoteSha 93f30fe5acfa3be82c281c39034223356b64e1bd

Contents

clear_cache 2
reset_ggplot2_palette 3
scale_sienna 3
sienna_colours 7
sienna_gradient 7
sienna_harmonise 8
sienna_hcl 9
sienna_pal 9
use_sienna_palette 10
use_speedups 11

Index **13**

`clear_cache` *Clear paradoxR's cache*

Description

Deletes everything paradoxR keeps in memory and on disk: the GO annotation data, `enrichGO()` results, KEGG tables and the stored results of slow analysis steps. The cache lives in `tools::R_user_dir("paradoxR", "cache")`, or in the directory set with `options(paradoxR.cache_dir = )` (the older name `paradoxR.go_cache_dir` still works).

Usage

```
clear_cache()
```

Details

paradoxR also cleans up by itself: a cached item that has not been used for 7 days (`options(paradoxR.cache_max_age_days = )`) is deleted the next time paradoxR or a package it speeds up is loaded or uses the cache, also when the speed-ups are turned off. Writing to the cache always leaves 15% of the disk or 15 GB free, whichever is more (`options(paradoxR.cache_min_free_pct = , paradoxR.cache_min_free_gb = )`); the least recently used items are deleted first, and if there is still not enough room the result is returned without being stored.

Value

The number of files deleted, invisibly.

Examples

```
clear_cache()
```

reset_ggplot2_palette *Restore ggplot2's default colour scales*

Description

Clears the `ggplot2.discrete.*`, `ggplot2.continuous.*`, `ggplot2.binned.*` and `ggplot2.ordinal.*` options set by `use_<palette>_palette()`, such as `use_sienna_palette()`.

Usage

```
reset_ggplot2_palette()
```

Value

The previous option values, invisibly.

Examples

```
use_sienna_palette()
reset_ggplot2_palette()
```

scale_sienna *Sienna scales for ggplot2*

Description

Drop-in replacements for ggplot2's default colour and fill scales.

Usage

```
scale_colour_sienna_d(
  ...,
  h_start = 18,
  direction = 1,
  harmonise = TRUE,
  aesthetics = "colour"
)

scale_fill_sienna_d(
  ...,
  h_start = 18,
  direction = 1,
  harmonise = TRUE,
  aesthetics = "fill"
)
```

```
scale_colour_sienna_c(
  palette = "ocean",
  ...,
  reverse = FALSE,
  begin = 0.2,
  end = 1,
  aesthetics = "colour"
)

scale_fill_sienna_c(
  palette = "ocean",
  ...,
  reverse = FALSE,
  begin = 0,
  end = 1,
  aesthetics = "fill"
)

scale_colour_sienna_b(
  palette = "ocean",
  ...,
  reverse = FALSE,
  begin = 0.2,
  end = 1,
  aesthetics = "colour"
)

scale_fill_sienna_b(
  palette = "ocean",
  ...,
  reverse = FALSE,
  begin = 0,
  end = 1,
  aesthetics = "fill"
)

scale_colour_sienna_o(
  palette = "ocean",
  ...,
  reverse = FALSE,
  begin = 0.2,
  end = 1,
  aesthetics = "colour"
)

scale_fill_sienna_o(
  palette = "ocean",
  ...,
```

```
  reverse = FALSE,
  begin = 0.2,
  end = 1,
  aesthetics = "fill"
)

scale_colour_sienna_div(
  palette = "red_blue",
  ...,
  midpoint = 0,
  reverse = FALSE,
  aesthetics = "colour"
)

scale_fill_sienna_div(
  palette = "red_blue",
  ...,
  midpoint = 0,
  reverse = FALSE,
  aesthetics = "fill"
)

scale_color_sienna_d(
  ...,
  h_start = 18,
  direction = 1,
  harmonise = TRUE,
  aesthetics = "colour"
)

scale_color_sienna_c(
  palette = "ocean",
  ...,
  reverse = FALSE,
  begin = 0.2,
  end = 1,
  aesthetics = "colour"
)

scale_color_sienna_b(
  palette = "ocean",
  ...,
  reverse = FALSE,
  begin = 0.2,
  end = 1,
  aesthetics = "colour"
)
```

```
scale_color_sienna_div(
  palette = "red_blue",
  ...,
  midpoint = 0,
  reverse = FALSE,
  aesthetics = "colour"
)

scale_color_sienna_o(
  palette = "ocean",
  ...,
  reverse = FALSE,
  begin = 0.2,
  end = 1,
  aesthetics = "colour"
)
```

Arguments

`...`	Passed to the underlying ggplot2 scale, such as `name`, `limits`, `breaks` or `na.value`.
`h_start`	Starting hue in degrees. 18 is the palette's red; `scales::pal_hue()` starts at 15.
`direction`	`1` for increasing hue, `-1` for decreasing.
`harmonise`	If `TRUE`, return harmonised colours; if `FALSE`, the original swatches for the same selection.
`aesthetics`	Aesthetics the scale applies to.
`palette`	A name from `sienna_gradients`, `sienna_diverging` or `sienna_colours`.
`reverse`	If `TRUE`, reverse the order of the stops.
`begin, end`	Part of the gradient to use, from 0 (low end) to 1 (high end), applied before `reverse`. Colour scales and ordinal scales start at `0.2`, so the palest colours, which barely show as points and lines on a white or grey panel, are left out; fill scales use the whole gradient.
`midpoint`	Data value mapped to the neutral centre.

Details

ggplot2 default	sienna replacement
`scale_colour_discrete()`	`scale_colour_sienna_d()`
`scale_fill_continuous()`	`scale_fill_sienna_c("ocean")`
`scale_fill_binned()`	`scale_fill_sienna_b("ocean")`
`scale_fill_ordinal()`	`scale_fill_sienna_o("ocean")`
`scale_fill_gradient2(midpoint = m)`	`scale_fill_sienna_div("red_blue", midpoint = m)`

Value

A ggplot2 scale.

Examples

```
p <- ggplot2::ggplot(ggplot2::mpg, ggplot2::aes(displ, hwy, colour = class)) +
  ggplot2::geom_point()
p + scale_colour_sienna_d()

ggplot2::ggplot(ggplot2::faithfuld, ggplot2::aes(waiting, eruptions, fill = density)) +
  ggplot2::geom_raster() +
  scale_fill_sienna_c("teal", reverse = TRUE)
```

---

sienna_colours *Sienna colours*

Description

The 24 sienna swatches. `sienna_colours` holds the original swatches for logos, tables and one-off accents. `sienna_colours_harmonised` holds the same hues with lightness and chroma pulled towards L 63, C 60; plots use these. `sienna_gradients`, `sienna_diverging` and `sienna_ramps` hold the 9-stop sequential, diverging and single-hue palettes.

Format

Named character vectors of hex codes (`sienna_colours`, `sienna_colours_harmonised`) and named lists of 9 hex codes each (`sienna_gradients`, `sienna_diverging`, `sienna_ramps`).

Examples

```
sienna_colours[["teal"]]
sienna_colours_harmonised[["teal"]]
names(sienna_gradients)
```

---

sienna_gradient *Sienna gradients*

Description

`sienna_gradient()` returns the 9 stops of a sequential, diverging or single-hue palette. `sienna_ramp()` returns the single-hue ramp of one swatch. `sienna_grad_pal()` returns a palette function for values in `[0, 1]`, like `scales::pal_gradient_n()`.

Usage

```
sienna_gradient(palette = "ocean", reverse = FALSE)

sienna_ramp(colour)

sienna_grad_pal(palette = "ocean", reverse = FALSE)
```

Arguments

palette	A name from sienna_gradients, sienna_diverging or sienna_colours.
reverse	If TRUE, reverse the order of the stops.
colour	A name from sienna_colours.

Details

Sequential gradients run light (low) to dark (high), the opposite of ggplot2's default; reverse = TRUE matches ggplot2.

Value

sienna_gradient() and sienna_ramp() return 9 hex codes; sienna_grad_pal() returns a function.

Examples

```
sienna_gradient("ocean")
sienna_ramp("purple")
sienna_grad_pal("sunset")(seq(0, 1, length.out = 10))
```

---

sienna_harmonise *Harmonise colours to the sienna palette*

Description

Keeps each colour's hue and pulls its lightness and chroma towards the palette's level, keeping strength of the original deviation. This is the pull that produced sienna_colours_harmonised. Needs the colorspace package.

Usage

```
sienna_harmonise(x, l = sienna_l, c = sienna_c, strength = sienna_harmony)
```

Arguments

x	Hex codes, optionally named.
l, c	Target luminance and chroma.
strength	Share of the original deviation to keep: 0 flattens fully, 1 leaves the colours untouched.

Value

Hex codes with the names of `x`.

Examples

```
if (rlang::is_installed("colorspace")) sienna_harmonise(c(accent = "#FF0066"))
```

---

sienna_hcl *Sienna colours from HCL coordinates*

Description

Converts hue, chroma and luminance (polar CIELUV, as in `grDevices::hcl()`) to hex. Out-of-gamut colours have their chroma lowered in steps of 0.5 until they fit, instead of returning `NA`.

Usage

```
sienna_hcl(h, c, l)
```

Arguments

`h`	Hue in degrees.
`c`	Chroma.
`l`	Luminance, 0 to 100.

Value

A character vector of hex codes.

Examples

```
sienna_hcl(h = 200, c = 40, l = 70)
```

---

sienna_pal *Sienna discrete palette*

Description

Drop-in replacement for `scales::pal_hue()`. Spaces `n` target hues evenly from `h_start` and snaps each to the nearest unused hue of the 13-hue sienna wheel, so group `k` lands on roughly the same hue as in ggplot2's default. Above 13 groups it falls back to evenly spaced hues at sienna lightness and chroma.

Usage

```
sienna_pal(h_start = 18, direction = 1, harmonise = TRUE)
```

Arguments

h_start	Starting hue in degrees. 18 is the palette's red; scales::pal_hue() starts at 15.
direction	1 for increasing hue, -1 for decreasing.
harmonise	If TRUE, return harmonised colours; if FALSE, the original swatches for the same selection.

Value

A function of n returning n hex codes.

Examples

```
sienna_pal()(5)
sienna_pal(h_start = 145, direction = -1)(3)
```

---

use_sienna_palette *Make sienna the default ggplot2 palette*

Description

Sets ggplot2's ggplot2.discrete.*, ggplot2.continuous.*, ggplot2.binned.* and ggplot2.ordinal.* options, so every plot uses sienna without a scale_*() call. reset_ggplot2_palette() switches back.

Usage

```
use_sienna_palette(h_start = 18, harmonise = TRUE, continuous = "ocean")
```

Arguments

h_start	Starting hue in degrees. 18 is the palette's red; scales::pal_hue() starts at 15.
harmonise	If TRUE, return harmonised colours; if FALSE, the original swatches for the same selection.
continuous	Palette for continuous, binned and ordinal scales.

Value

The previous option values, invisibly.

Examples

```
use_sienna_palette(continuous = "ocean")
ggplot2::ggplot(ggplot2::mpg, ggplot2::aes(displ, hwy, colour = class)) +
  ggplot2::geom_point()
reset_ggplot2_palette()
```

use_speedups *Apply or remove paradoxR's speed-ups*

Description

use_speedups() replaces functions inside clusterProfiler, DOSE, GOSemSim, DESeq2, sva, Seurat and DiffBind so that clusterProfiler::enrichGO(), clusterProfiler::simplify() and the functions that call them, such as clusterProfiler::compareCluster(), reuse earlier work and run faster:

Usage

```
use_speedups(packages = NULL)

reset_speedups(packages = NULL)
```

Arguments

packages	Names of the packages to speed up or restore, any of "clusterProfiler", "DOSE", "GOSemSim", "DESeq2", "sva", "Seurat" and "DiffBind". NULL, the default, means all of them.

Details

- the GO annotation for an OrgDb, ontology and key type is built once and kept on disk, so a new R session reads it in under a second instead of querying the OrgDb again;
- the result of an enrichGO() call is kept, so the same gene list with the same settings returns at once;
- Wang semantic similarity between GO terms is computed for all term pairs at once instead of pair by pair, which speeds up simplify() and enrichplot::pairwise_termsim(method = "Wang"); results are unchanged;
- the first GO annotation for an OrgDb is read with one SQL query per ontology table instead of AnnotationDbi::mapIds(), and information content in GOSemSim::godata() is summed over all terms at once; the results are identical;
- DESeq2::DESeq() computes the trimmed means behind Cook's distances with one sort per matrix instead of one mean(trim =) call per gene; a gene's value is used only where it provably equals the original;
- sva::ComBat_seq() moves counts to their adjusted quantiles in one vectorised stats::pnbinom() and stats::qnbinom() call instead of one call per gene and sample;
- every KEGG table that enrichKEGG(), gseKEGG(), enrichMKEGG() and gseMKEGG() download from the KEGG REST API is kept on disk per species, KEGG database and key type, so switching species or key type, or starting a new R session, no longer downloads it again. The cache is checked against KEGG's release dates once an hour and downloads again when KEGG has been updated; when KEGG is unreachable, the cached tables are used;

- the results of `Seurat::FindAllMarkers()` and `DiffBind::dba.count()` are stored on disk, so running a script or rendering a document again returns them at once when nothing they depend on has changed. A stored result is used only when the argument values, the content of every input file (such as the BAM files and their indexes that a DiffBind object points to), the installed versions of the package and everything it depends on, the R build and the relevant options all match; messages, warnings and the random number state of the original call are replayed. Only calls that took at least 10 seconds are stored (`options(paradoxR.result_cache_min_seconds = )`), and `options(paradoxR.result_cache = FALSE)` turns this off. Nothing is stored when the cache directory is on a network drive or when the `ps` package, which measures free disk space, is not installed. `qs2`, when installed, makes storing and reading faster.

Called automatically when paradoxR is loaded. Packages loaded later are sped up as they load. The cache is keyed by the clusterProfiler and GO.db versions and the OrgDb database file, so updating any of them starts a fresh cache. Every cached item that has not been used for 7 days is deleted automatically; see `clear_cache()`. The `simplify()`, similarity, DESeq2 and sva replacements apply only to the package versions they were checked against; other versions keep their own code.

`reset_speedups()` restores the original functions, for example to compare results against the original code. With `packages`, only the functions of those packages are restored and the others stay fast. A restored package stays original for the rest of the session, also if it is loaded later, until `use_speedups()` is called for it. `clear_cache()` deletes the cache.

To keep the original functions from the start of a session, set `options(paradoxR.speedups = FALSE)` (all packages), `options(paradoxR.deseq2 = FALSE)` or `options(paradoxR.sva = FALSE)` before loading paradoxR.

Value

`use_speedups()` returns `TRUE` invisibly when at least one function of `packages` is replaced, `FALSE` when none is, for example because the packages are not loaded yet; they are then sped up when they load. `reset_speedups()` returns `TRUE` invisibly when it restored at least one function.

Examples

```
use_speedups()
reset_speedups("DESeq2")
use_speedups("DESeq2")
reset_speedups()
use_speedups()
```

Index

clear_cache, 2
clear_cache(), *12*

grDevices::hcl(), *9*

reset_ggplot2_palette, 3
reset_ggplot2_palette(), *10*
reset_speedups *(*use_speedups*)*, 11

scale_color_sienna_b *(*scale_sienna*)*, 3
scale_color_sienna_c *(*scale_sienna*)*, 3
scale_color_sienna_d *(*scale_sienna*)*, 3
scale_color_sienna_div *(*scale_sienna*)*, 3
scale_color_sienna_o *(*scale_sienna*)*, 3
scale_colour_sienna_b *(*scale_sienna*)*, 3
scale_colour_sienna_c *(*scale_sienna*)*, 3
scale_colour_sienna_d *(*scale_sienna*)*, 3
scale_colour_sienna_div *(*scale_sienna*)*, 3
scale_colour_sienna_o *(*scale_sienna*)*, 3
scale_fill_sienna_b *(*scale_sienna*)*, 3
scale_fill_sienna_c *(*scale_sienna*)*, 3
scale_fill_sienna_d *(*scale_sienna*)*, 3
scale_fill_sienna_div *(*scale_sienna*)*, 3
scale_fill_sienna_o *(*scale_sienna*)*, 3
scale_sienna, 3
scales::pal_gradient_n(), *7*
scales::pal_hue(), *9*
sienna_colours, 7
sienna_colours_harmonised, *8*
sienna_colours_harmonised *(*sienna_colours*)*, 7
sienna_diverging *(*sienna_colours*)*, 7
sienna_grad_pal *(*sienna_gradient*)*, 7
sienna_gradient, 7
sienna_gradients *(*sienna_colours*)*, 7
sienna_harmonise, 8
sienna_hcl, 9
sienna_pal, 9
sienna_ramp *(*sienna_gradient*)*, 7
sienna_ramps *(*sienna_colours*)*, 7

use_sienna_palette, 10
use_sienna_palette(), *3*
use_speedups, 11