Package ‘scattermore’

June 12, 2023

Title Scatterplots with More Points
Version 1.2
Description C-based conversion of large scatterplot data to rasters plus other operations such as data blurring or data alpha blending. Speeds up plotting of data with millions of points.
Imports ggplot2, scales, grid, grDevices, graphics
License GPL (>= 3)
Encoding UTF-8
RoxygenNote 7.2.3
Suggests covr, knitr, rmarkdown, testthat
VignetteBuilder knitr
NeedsCompilation yes
Author Tereza Kulichova [aut],
         Mirek Kratochvil [aut, cre] (<https://orcid.org/0000-0001-7356-4075>)
Maintainer Mirek Kratochvil <exa.exa@gmail.com>
Repository CRAN
Date/Publication 2023-06-12 08:30:02 UTC

R topics documented:

apply_kernel_histogram ........................................ 2
apply_kernel_rgbwt ........................................... 3
blend_rgba_float .............................................. 3
GeomScattermore .............................................. 4
GeomScattermost .............................................. 4
geom_scattermore ............................................ 5
geom_scattermost ............................................ 6
histogram_to_rgbwt .......................................... 7
merge_rgbwt .................................................. 7
rgba_float_to_rgba_int ...................................... 8
rgba_int_to_raster .......................................... 8
apply_kernel_histogram

Description

Apply a kernel to the given histogram.

Usage

apply_kernel_histogram(
  fhistogram,
  filter = "circle",
  mask = default_kernel(filter, radius, sigma),
  radius = 2,
  sigma = radius/2,
  threads = 0
)

Arguments

fhistogram  Matrix or array interpreted as histogram of floating-point values.
filter      Use the pre-defined filter, either circle, square, gauss. Defaults to circle.
mask        Custom kernel used for blurring, overrides filter. Must be a square matrix of odd size.
radius      Radius of the kernel (counted without the "middle" pixel"), defaults to 2. The generated kernel matrix will be a square with (2*radius+1) pixels on each side.
sigma       Radius of the Gaussian function selected by filter, defaults to radius/2.
threads     Number of parallel threads (default 0 chooses hardware concurrency).

Value

2D matrix with the histogram processed by the kernel application.
**apply_kernel_rgbwt**

---

**Description**

Apply a kernel to the given RGBWT raster.

**Usage**

```r
apply_kernel_rgbwt(
    fRGBWT,
    filter = "circle",
    mask = default_kernel(filter, radius, sigma),
    radius = 2,
    sigma = radius/2,
    threads = 0
)
```

**Arguments**

- **fRGBWT**
  RGBWT array with channels red, green, blue, weight and transparency. The dimension should be N times M times 5.
- **filter**
  Use the pre-defined filter, either circle, square, gauss. Defaults to circle.
- **mask**
  Custom kernel used for blurring, overrides filter. Must be a square matrix of odd size.
- **radius**
  Radius of the kernel (counted without the "middle" pixel"), defaults to 2. The generated kernel matrix will be a square with (2*radius+1) pixels on each side.
- **sigma**
  Radius of the Gaussian function selected by filter, defaults to radius/2.
- **threads**
  Number of parallel threads (default 0 chooses hardware concurrency).

**Value**

RGBWT matrix.

---

**blend_rgba_float**

---

**Description**

Blend RGBA matrices.

**Usage**

```r
blend_rgba_float(fRGBA_list)
```
Arguments

\text{fRGBA\_list} \quad \text{List of floating-point RGBA arrays with premultiplied alpha (each of the same size N-by-M-by-4). The "first" matrix in the list is the one that will be rendered on "top".}

Value

Blended RGBA matrix.

---

**GeomScattermore**

\textit{The actual geom for scattermore}

---

Description

The actual geom for scattermore

Usage

\text{GeomScattermore}

Format

An object of class \text{GeomScattermore} (inherits from \text{Geom, ggproto, gg}) of length 6.

---

**GeomScattermost**

\textit{The actual geom for scattermost}

---

Description

The actual geom for scattermost

Usage

\text{GeomScattermost}

Format

An object of class \text{GeomScattermost} (inherits from \text{Geom, ggproto, gg}) of length 4.
Description

ggplot2::ggplot() integration. This cooperates with the rest of ggplot (so you can use it to
e.g. add rasterized scatterplots to vector output in order to reduce PDF size). Note that the ggplot
processing overhead still dominates the plotting time. Use geom_scattermost() to tradeoff some
niceness and circumvent ggplot logic to gain speed.

Usage

geom_scattermore(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ..., 
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  interpolate = FALSE,
  pointsize = 0,
  pixels = c(512, 512)
)

Arguments

mapping, data, stat, position, inherit.aes, show.legend, ...
  passed to ggplot2::layer()
na.rm           Remove NA values, just as with ggplot2::geom_point().
interpolate     Default FALSE, passed to grid::rasterGrob().
pointsize       Radius of rasterized point. Use 0 for single pixels (fastest).
pixels          Vector with X and Y resolution of the raster, default c(512, 512).

Details

Accepts aesthetics x, y, colour and alpha. Point size is fixed for all points. Due to rasterization
properties it is often beneficial to try non-integer point sizes, e.g. 3.2 looks much better than 3.

Examples

library(ggplot2)
library(scattermore)
ggplot(data.frame(x = rnorm(1e6), y = rexp(1e6))) +
  geom_scattermore(aes(x, y, color = x),
                 pointsize = 3,
Description

Totally non-ggplotish version of `geom_scattermore()`, but faster. It avoids most of the ggplot processing by bypassing the largest portion of data around any ggplot functionality, leaving only enough data to set up axes and limits correctly. If you need to break speed records, use this.

Usage

```r
geom_scattermost(
  xy,
  color = "black",
  interpolate = FALSE,
  pointsize = 0,
  pixels = c(512, 512)
)
```

Arguments

- `xy` 2-column object with data, as in `scattermore()`.
- `color` Color vector (or a single color).
- `interpolate` Default FALSE, passed to `grid::rasterGrob()`.
- `pointsize` Radius of rasterized point. Use 0 for single pixels (fastest).
- `pixels` Vector with X and Y resolution of the raster, default `c(512, 512)`.

Examples

```r
library(ggplot2)
library(scattermore)
d <- data.frame(x = rnorm(1000000), y = rnorm(1000000))
x_rng <- range(d$x)
ggplot() +
  geom_scattermost(cbind(d$x, d$y),
                   color = heat.colors(100, alpha = .01)
                   [1 + 99 * (d$x - x_rng[1]) / diff(x_rng)],
                   pointsize = 2.5,
                   pixels = c(1000, 1000),
                   interpolate = TRUE)
```
histogram_to_rgbwt

Description
Colorize given histogram with input palette.

Usage

```r
histogram_to_rgbwt(
  fhistogram,
  RGBA = grDevices::col2rgb(col, alpha = T),
  col = grDevices::hcl.colors(10),
  zlim = c(min(fhistogram), max(fhistogram))
)
```

Arguments
- `fhistogram` Matrix or 2D array with the histogram of values.
- `col` Colors to use for coloring.
- `zlim` Values to use as extreme values of the histogram

Value
RGBWT matrix.

merge_rgbwt

Description
Merge RGBWT matrices.

Usage

```r
merge_rgbwt(fRGBWT_list)
```

Arguments
- `fRGBWT_list` List of RGBWT arrays. The order of the matrices does not matter (except for negligible floating-point rounding and other robustness errors).

Value
Merged RGBWT matrix.
rgba_float_to_rgba_int

Description
Convert a float RGBA bitmap with pre-multiplied alpha to integer RGBA bitmap.

Usage
rgba_float_to_rgba_int(fRGBA)

Arguments
fRGBA RGBA bitmap in N-by-M-by-4 array.

Value
RGBA matrix. The output is not premultiplied by alpha.

rgba_int_to_raster

Description
Create a raster from the given RGBA matrix.

Usage
rgba_int_to_raster(i32RGBA)

Arguments
i32RGBA Integer RGBA matrix (with all values between 0 and 255).

Value
The matrix converted to raster.
rgbwt_to_rgba_float

Description

Convert RGBWT matrix to floating-point RGBA matrix, suitable for alpha-blending.

Usage

rgbwt_to_rgba_float(fRGBWT)

Arguments

fRGBWT The RGBWT matrix.

Value

RGBA matrix, output is premultiplied by alpha.

rgbwt_to_rgba_int

Description

Convert a RGBWT matrix to an integer RGBA matrix.

Usage

rgbwt_to_rgba_int(fRGBWT)

Arguments

fRGBWT The RGBWT matrix.

Value

A RGBA matrix. The output is not premultiplied by alpha.
Description

Convert points to raster scatterplot rather quickly.

Usage

scattermore(xy, size = c(512, 512), xlim = c(min(xy[, 1]), max(xy[, 1])), ylim = c(min(xy[, 2]), max(xy[, 2])), rgba = c(0L, 0L, 0L, 255L), cex = 0, output.raster = TRUE)

Arguments

- **xy**: 2-column float matrix with point coordinates. As usual with rasters in R, X axis grows right, and Y axis grows DOWN. Flipping ylim causes the usual mathematical behavior.
- **size**: 2-element vector integer size of the result raster, defaults to c(512, 512).
- **xlim, ylim**: Float limits as usual (position of the first pixel on the left/top, and the last pixel on the right/bottom). You can easily flip the top/bottom to the "usual" mathematical system by flipping the ylim vector.
- **rgba**: 4-row matrix with color values of 0-255, or just a single 4-item vector for c(r,g,b,a). Best created with col2rgb(..., alpha=TRUE).
- **cex**: Additional point radius in pixels, 0=single-pixel dots (fastest)
- **output.raster**: Output R-style raster (as.raster)? Default TRUE. Raw array output can be used much faster, e.g. for use with png::writePNG.

Value

Raster with the result.

Examples

library(scattermore)
plot(scattermore(cbind(rnorm(1e6), rnorm(1e6)), rgba = c(64, 128, 192, 10)))
Description

Convenience base-graphics-like layer around `scattermore`. Currently only works with linear axes!

Usage

```r
scattermoreplot(
  x, y, xlim, ylim, size,
  col = grDevices::rgb(0, 0, 0, 1),
  cex = 0,
  pch = NULL,
  xlab, ylab,
  ...
)
```

Arguments

- `x, y, xlim, ylim, xlab, ylab, ...`
  used as in `graphics::plot()` or forwarded to `graphics::plot()`
- `size` forwarded to `scattermore()`, or auto-derived from device and plot size if missing (the estimate is not pixel-perfect on most devices, but gets pretty close)
- `col` point color(s)
- `cex` forwarded to `scattermore()`
- `pch` ignored (to improve compatibility with `graphics::plot()`

Examples

```r
# plot an actual rainbow
library(scattermore)
d <- data.frame(s = qlogis(1:1e6 / (1e6 + 1), 6, 0.5), t = rnorm(1e6, pi / 2, 0.5))
scattermoreplot(
  d$s * cos(d$t),
  d$s * sin(d$t),
  col = rainbow(1e6, alpha = .05)[c((9e5 + 1):le6, 1:9e5)],
  main = "scattermore demo"
)
```
**scatter_lines_histogram**

**Description**

Render lines into a histogram.

**Usage**

```r
scatter_lines_histogram(
  xy,
  xlim = c(min(xy[, c(1, 3)]), max(xy[, c(1, 3)])),
  ylim = c(min(xy[, c(2, 4)]), max(xy[, c(2, 4)])),
  out_size = c(512L, 512L),
  skip_start_pixel = FALSE,
  skip_end_pixel = TRUE
)
```

**Arguments**

- `xy` 4-column matrix with point coordinates. Each row contains X and Y coordinates of line start and X and Y coordinates of line end, in this order.
- `xlim`, `ylim` 2-element vector of rendered area limits (position of the first pixel on the left/top, and the last pixel on the right/bottom). You can flip the image coordinate system by flipping the `*lim` vectors.
- `out_size` 2-element vector size of the result raster, defaults to `c(512L, 512L)`.
- `skip_start_pixel` TRUE if the start pixel of the lines should be omitted, defaults to FALSE.
- `skip_end_pixel` TRUE if the end pixel of a line should be omitted, defaults to TRUE. (When plotting long ribbons of connected lines, this prevents counting the connecting pixels twice.)

**Value**

Histogram with the rendered lines.
Description

Render lines into a RGBWT bitmap.

Usage

scatter_lines_rgbwt(
  xy,
  xlim = c(min(xy[, c(1, 3)]), max(xy[, c(1, 3)])),
  ylim = c(min(xy[, c(2, 4)]), max(xy[, c(2, 4)])),
  out_size = c(512L, 512L),
  RGBA = c(0, 0, 0, 255),
  skip_start_pixel = FALSE,
  skip_end_pixel = TRUE
)

Arguments

xy 4-column matrix with point coordinates. Each row contains X and Y coordinates of line start and X and Y coordinates of line end, in this order.

xlim, ylim 2-element vector of rendered area limits (position of the first pixel on the left/top, and the last pixel on the right/bottom). You can flip the image coordinate system by flipping the *lim vectors.

out_size 2-element vector size of the result raster, defaults to c(512L, 512L).

RGBA Vector of 4 elements with integral RGBA color for the lines, defaults to c(0, 0, 0, 255).

skip_start_pixel TRUE if the start pixel of the lines should be omitted, defaults to FALSE.

skip_end_pixel TRUE if the end pixel of a line should be omitted, defaults to TRUE. (When plotting long ribbons of connected lines, this prevents counting the connecting pixels twice.)

Value

Lines plotted in RGBWT bitmap.
scatter_points_histogram

Description

Render a 2D histogram with given points

Usage

scatter_points_histogram(xy, xlim = c(min(xy[, 1]), max(xy[, 1])), ylim = c(min(xy[, 2]), max(xy[, 2])), out_size = c(512L, 512L))

Arguments

xy 2-column matrix with point coordinates (X and Y).
xlim, ylim 2-element vector of rendered area limits (position of the first pixel on the left/top, and the last pixel on the right/bottom). You can flip the image coordinate system by flipping the *lim vectors.
out_size 2-element vector size of the result raster, defaults to c(512L, 512L).

Value

2D histogram with the points "counted" in appropriate pixels.

scatter_points_rgbwt

Description

Render colored points into a RGBWT bitmap

Usage

scatter_points_rgbwt(xy, xlim = c(min(xy[, 1]), max(xy[, 1])), ylim = c(min(xy[, 2]), max(xy[, 2])), out_size = c(512, 512), RGBA = c(0, 0, 0, 255), map = NULL, palette = NULL)
**Arguments**

- **xy** 2-column matrix with N point coordinates (X and Y) in rows.
- **xlim, ylim** 2-element vector of rendered area limits (position of the first pixel on the left/top, and the last pixel on the right/bottom). You can flip the image coordinate system by flipping the `*lim` vectors.
- **out_size** 2-element vector size of the result raster, defaults to `c(512L, 512L)`.
- **RGBA** Point colors. Either a 4-element vector that specifies the same color for all points, or 4-by-N matrix that specifies color for each of the individual points. Color is specified using integer RGBA; i.e. the default black is `c(0, 0, 0, 255)`.
- **map** Vector with N integer indices to `palette`. Overrides RGBA-based coloring.
- **palette** Matrix 4-by-K matrix of RGBA colors used as a palette lookup for the map that gives the point colors. K is at least `max(map)`. Notably, using a palette may be faster than filling and processing the whole RGBA matrix.

**Value**

A RGBWT array with the rendered points.
Index

* datasets
  - GeomScattermore, 4
  - GeomScattermost, 4

apply_kernel_histogram, 2
apply_kernel_rgbwt, 3

blend_rgba_float, 3

geom_scattermore, 5
geom_scattermore(), 6
geom_scattermost, 6
geom_scattermost(), 5
GeomScattermore, 4
GeomScattermost, 4
ggplot2::geom_point(), 5
ggplot2::ggplot(), 5
ggplot2::layer(), 5
graphics::plot(), 11
grid::rasterGrob(), 5, 6

histogram_to_rgbwt, 7
merge_rgbwt, 7

rgba_float_to_rgba_int, 8
rgba_int_to_raster, 8
rgbwt_to_rgba_float, 9
rgbwt_to_rgba_int, 9

scatter_lines_histogram, 12
scatter_lines_rgbwt, 13
scatter_points_histogram, 14
scatter_points_rgbwt, 14
scattermore, 10
scattermore(), 6, 11
scattermoreplot, 11