Package 'rampage'

June 19, 2026

Type Package

Title Calibrated Color Ramps

Version 0.2.1

Maintainer Adam T. Kocsis <adam.t.kocsis@gmail.com>

Description Value-calibrated color ramps can be useful to emphasize patterns in data from complex distributions. Colors can be tied to specific values, and the association can be expanded into full color ramps that also include the relationship between colors and values. Such ramps can be used in a variety of cases when heatmap-type plots are necessary, including the visualization of vector and raster spatial data, such as topographies.

License CC BY 4.0

Date 2026-06-19

URL https://adamtkocsis.com/rampage/

BugReports https://github.com/adamkocsis/rampage/issues

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports grDevices

NeedsCompilation no

VignetteBuilder knitr

Suggests knitr, rmarkdown, terra

Config/roxygen2/version 8.0.0

Author Adam T. Kocsis [cre, aut] (ORCID:
<https://orcid.org/0000-0002-9028-665X>),
Deutsche Forschungsgemeinschaft [fnd],
FAU GeoZentrum Nordbayern [fnd]

Repository CRAN

Date/Publication 2026-06-19 10:20:02 UTC

Contents

bathymetry . . . 2
colorpoints . . . 3
expand . . . 4
limit . . . 5
paleomap . . . 6
plot.calibramp . . . 7
rampage . . . 8
ramplegend . . . 8
ramps . . . 10
topos . . . 11
trimramp . . . 12

Index **13**

`bathymetry` *Bathymetric color palettes with tiepoints*

Description

The object contains `data.frame`-class objects to be used with the `expand` function to produce full calibrated color ramps.

Usage

```
data(bathymetry)
```

Format

A `list` with 1 `data.frame` elements:

`sandy` : A color ramp resembling a sandy beach.

Examples

```
data(bathymetry)
sandyExp <- expand(bathymetry$sandy, n=200)
plot(sandyExp)
```

`colorpoints` *Heatmap with colored points*

Description

The function is a wrapper around the `points` function, controlling the color of the points similar to `ggplot`, but using S-style plotting. If neither `ramp`, nor `col` or `breaks` are given, the function will default to using the internal `gradinv` palette with 256 levels evenly distributed from the minimum to the maximum of `z`.

Usage

```
colorpoints(
  x,
  y = NULL,
  z,
  ramp = NULL,
  col = NULL,
  breaks = NULL,
  legend = list(x = "topleft"),
  ...
)
```

Arguments

`x`	The `x` argument of `points`.
`y`	The `y` argument of `points`.
`z`	`numeric`, the variable to visualize using the colors.
`ramp`	A `calibramp`-class object (including both `breaks` and `colors`).
`col`	A vector of colors. Used only if `ramp` is not given.
`breaks`	A vector of breaks. If given, this has to be one element longer than the length of `col`.
`legend`	A list of arguments passed to the `ramplegend` function. Set to `NULL` if you do not want to plot a legend.
`...`	Arguments passed to the `points` function.

Value

The function has no return value.

Examples

```
# random points
set.seed(1)
x <- rnorm(5000) # x coord
```

```
y <- rnorm(5000) # y coord
dist <- sqrt(x^2+y^2) # distance from origin

# default plotting
plot(x,y, col=NA)
colorpoints(x=x, y=y, z=dist)

# custom color scheme
levs<- data.frame(color=rainbow(5), z=c(0, 0.5,1, 3, 4.5))
ramp <-expand(levs, n=256)

# very customized (experiment with difference device sizes!)
plot(x,y, col=NA, main="Distance to origin")
colorpoints(x=x, y=y, z=dist,
col=paste0(ramp$col, "BB"),
breaks=ramp$breaks,
pch=16,
legend=list(x=3,y=0,cex=0.7, box.args=list(border=NA)))
```

---

expand *Create a calibrated color ramp object from color-tiepoint data.frames*

Description

Create ramp palettes from fixed color positions.

Usage

```
expand(x = NULL, n, color = "color", z = "z", ...)
```

Arguments

x	A `data.frame` object with two columns: `color` for hexadecimal color values, `z` for their position.
n	A single `integer` number.
color	A `chraracter` value, the column name of the colors in `x`, defaults to `"color"`. Alternatively, a `character` vector of hexadecimal color values, with the same length as `z`.
z	A `character` value, the column name of the values in `x`, defaults to `"z"`. Alternatively, a `numeric` vector of color values (must have the same length as `color`.
...	Arguments passed to the `colorRampPalette` function.

Details

The function creates objects of the S3 class `calibramp`. The `calibramp`-class lists have three elements: `col` hexadecimal color values, `mid`: z-values of midpoints (one for every color), and `breaks`: separator borders between color values. Color interpolation will be executed linearly, using `colorRampPalette`, the order of the values will be forced to ascending, the values in `mid` will be halfway between breaks.

Value

A calibramp-class object (see description above).

Examples

```
library(rampage)
data(topos)
ramp <- expand(topos$havanna2, n=200)
plot(ramp)
```

---

limit *Limit range of values in a* SpatRaster *object*

Description

Shorthand for limiting maximum and minimum values in a SpatRaster-class object.

Usage

```
limit(x, y = NULL, min = NULL, max = NULL)
```

Arguments

x	Object to limit, a SpatRaster-class object.
y	Either a range (i.e. a numeric vector with two values), or data.frame with positioned color values (column z indicates values), or a calibrated ramp (e.g. produced with expand).
min	If y is not given, the minimum value to have in the data.frame.
max	If y is not given, the maximum value to have in the data.frame.

Value

A SpatRaster object.

Examples

```
# This function relies on the terra extension
if(requireNamespace("terra", quietly=TRUE)){
 library(terra)
# Example 1. Using specific values
# a SpatRaster object
r<- terra::rast()
# populate with a Gaussian distribution
terra::values(r) <- rnorm(terra::ncell(r), 0.5,1 )
# and limit
rLimit <- limit(r, min=-0.2, max=0.2)
plot(rLimit)
```

```
# Example 2. Using an expanded color ramp
# Create a data.frame
df <- data.frame(
z=c(-1, -0.2, 0, 0.2, 1),
color=rev(gradinv(5))
)
ramp <- expand(df, n=200)
rLimited <- limit(r, y=ramp)
# default
plot(rLimited)

# manual ramping.
plot(rLimited, breaks=ramp$breaks, col=ramp$col,
legend=FALSE)

# temporary solution for manual legend
# Marginal ramps will be implemented later
ramplegend(x=140, y=90, ramp=ramp, cex=0.5,
at=c(-1, 0, 1), label=c("< -1", 0, "> +1"))


}
```

`paleomap` *Topographic gradient map color map of the PALEOMAP project*

Description

The elevation to color bindings are the work of C. Scotese. Data from Scotese, C. R., Vérard, C., Burgener, L., Elling, R. P., & Kocsis, A. T. (2025). The Cretaceous World: Plate Tectonics, Paleogeography, and Paleoclimate. Geological Society, London, Special Publications, 544(1), SP544–2024..

Usage

```
data(paleomap)
```

Format

A `calibramp`-class `list` with 3 `numerics`:

`col` : The color levels as hexadecimal RGB values.

`breaks` : The boundaries for the individual levels.

`mid` : The mid values of the color levels.

Source

https://zenodo.org/records/10659112

`plot.calibramp` *Visualize a calibrated color ramp*

Description

The method can be used to inspect and visualize calbirated color ramp object.

Usage

```
## S3 method for class 'calibramp'
plot(x, ...)

rampplot(
  x,
  breaks = FALSE,
  breaklabs = TRUE,
  axis.args = list(side = 2),
  ylab = "z",
  xlab = ""
)
```

Arguments

`x`	The calibrated color ramp object (`calibramp`-class object).
`...`	Arguments passed to the `rampplot` function.
`breaks`	Should the distribution of breaks be visualized?
`breaklabs`	Should the minimum and maximum break labels be visualized?
`axis.args`	Arguments passed to the axis function.
`ylab`	y-axis label.
`xlab`	x-axis label.

Value

The functions have no return values.

Examples

```
# the paleomap ramp
data(paleomap)
plot(paleomap)
# 0-calibrated, expanded ramp
tiepoints <- data.frame(z=c(c(-1, -0.1, 0, 0.1, +1)), color=gradinv(5))
ramp <- expand(tiepoints, n=255)
plot(ramp)
```

rampage *Strechable Color Ramps*

Description

Value-calibrated color ramps can be useful to emphasize patterns in weirdly distributed data. Tie-points can be used to fix colors to specific values, and the resulting association can then be expanded into full color ramps. Such ramps can be used a variety of cases when heatmap type plotting is necessary, including the visualization of raster and polygon spatial data.

Details

This is still a Beta version. As is R, this is free software and comes with ABSOLUTELY NO WARRANTY. Nevertheless, notes about found bugs and suggestions are more than welcome.

Author(s)

Adam T. Kocsis (adam.t.kocsis@gmail.com)

See Also

Useful links:

- https://adamtkocsis.com/rampage/
- Report bugs at https://github.com/adamkocsis/rampage/issues

ramplegend *Create a heatmap legend based on calibrated color ramp values*

Description

Optional legend for cases where calibramp objects are used.

Usage

```
ramplegend(
  x = "topleft",
  y = NULL,
  shift = c(0, 0),
  ramp = NULL,
  col = NULL,
  breaks = NULL,
  zlim = NULL,
  height = 3,
  width = 0.3,
  tick.length = 0.15,
```

```
  cex = 1,
  box.args = list(col = "#ffffffbb"),
  horizontal = FALSE,
  at = NULL,
  label = NULL
)
```

Arguments

x	Position of the legend or the left coordinate of legend box. Either a numeric coordinate value or a the "topleft", "topright", "bottomleft" or "bottomright".
y	Coordinate of the upper edge of the legend (if needed).
shift	Used instead of the inset argument of the default legend. If plotted within the inner plotting area, the x and y user coordinates with which the position of the legend will be shifted.
ramp	A calibrated color ramp object. Either ramp or both col and breaks are required.
col	Vector of colors.
breaks	Breaks between the colors.
zlim	A numeric vector with two values, passed to trimramp. The low and high extreme values to be shown on the legend.
height	Height of the legend bar in inches.
width	Width of the legend bar in inches.
tick.length	The length of the legend's ticks.
cex	Legend size scaler.
box.args	the box's arguments.
horizontal	Legend orientation. Not yet implemented
at	Where should the legend be drawn in the z dimension?
label	What are the labels

Value

The function has no return value.

Examples

```
# example with colored points
# basic points
v <- seq(0,20, 0.01)
sine <- sin(v)

# visualize as a plot
plot(v,sine)

# colors for sine values
levs<- data.frame(color=gradinv(5), z=c(-1, -0.2, 0, 0.2, 1))
```

```
ramp<- expand(levs, n=256)

# colored points
colorpoints(x=v, y=sine, z=sine, cex=6, pch=16, legend=NULL)

# the legend
ramplegend(x=0, y=0.3, ramp=ramp, cex=0.5, box.args=list(border=NA, col=NA))


# example with histogram
set.seed(1)
x <- rnorm(3000, 3,1)
levs<- data.frame(color=gradinv(7), z=c(-1, 1,1.04, 3, 4.96, 5, 7))
ramp <-expand(levs, n=400)

# histogram showing distribution
hist(x, col=ramp$col, breaks=ramp$breaks, border=NA)
ramplegend("topleft", ramp=ramp, at=c(1.04, 3, 4.96), label=c("-1.96 SD", "mean", "+1.96 SD"))


# example with volcano
data(volcano)
data(topos)

# create ramp
levs <- topos$jakarta[topos$jakarta$z>0,]
levs$z <- c(200, 180, 165, 130, 80)
ramp <-expand(levs, n=100)

image(volcano, col=ramp$col, breaks=ramp$breaks)
ramplegend(x=0.8, y=0.8, ramp=ramp, cex=0.9)
```

`ramps` *Color gradient ramps*

Description

Contains functions produced by the `colorRampPalette` function.

Usage

```
gradinv(n)
```

Arguments

`n`	(`numeric`) Number of different colors to generate from the palette

Details

You can also view single palettes individually. The following color palettes are implemented:

- `gradinv()`: inverse heatmap, primarily intended to emphasize distinctions between no-change (yellow) and change (blue/red) cases. Based on the color blindness simulator Coblis, the palette is very color disability-friendly, except for monochromacy/achromatopsia, where the two change scenarios (red and blue) become very difficult to distinguish. Additional markings (e.g. labels indicating high and low) are recommended for these cases.

Value

A character vector of color values.

Examples

```
cols <- gradinv(20)
plot(1:20, col=cols, pch=16, cex=2)
```

---

topos *Topographic color palettes with tiepoints*

Description

The object contains `data.frame`-class objects to be used with the `expand` function to produce full calibrated color ramps.

Usage

```
data(topos)
```

Format

A `list` with 7 `data.frame` elements:

`demcmap` : The "demcmap" theme, based on MatLab's `demcmap`.

`etopo` : The "etopo" theme, approximate elevation-color assignments based on the the ETOPO Global Relief Model poster (https://www.ncei.noaa.gov/media/3340).

`jakarta` : The "Jakarta" theme, color values by Deviantart user *Arcanographia*.

`havanna2` : The "Havanna-2" theme, color values by Deviantart user *Arcanographia*.

`ptolemy` : The "Ptolemy" theme, color values extracted from the 'The world map after Ptolemy's first projection from a Greek manuscript edition of the Geography' (Burney MS 111).

`tokio1` : The "Tokio-1" theme, color values by Deviantart user *Arcanographia*.

`zagreb` : The "Zagreb" theme, color values by Deviantart user *Arcanographia*.

Examples

```
data(topos)
jakExp <- expand(topos$jakarta, n=200)
plot(jakExp)
```

`trimramp` *Trimming a calibrated color ramp object.*

Description

Modify the minimum and maximum values in a `calibramp`-class object produced by the `expand` function.

Usage

```
trimramp(x, low = NULL, high = NULL)
```

Arguments

`x`	A calibrated color ramp (e.g. `calibramp`-class object.
`low`	A single `numeric` value, the minimum value in the calibrated ramp.
`high`	A single `numeric` value, the maximum value in the calibrated ramp.

Value

A trimmed version of `x`, another `calibramp`-class object.

Examples

```
data(paleomap)
trimmed <- trimramp(paleomap, low=-500, high=1500)
plot(trimmed)
```

Index

∗ **datasets**
bathymetry, 2
paleomap, 6
topos, 11

bathymetry, 2

colorpoints, 3
colorRampPalette, *4*, *10*

expand, *2*, 4, *11*, *12*

gradinv *(*ramps*)*, 10

limit, 5

paleomap, 6
plot.calibramp, 7
points, *3*

rampage, 8
rampage-package *(*rampage*)*, 8
ramplegend, *3*, 8
rampplot *(*plot.calibramp*)*, 7
ramps, 10

topos, 11
trimramp, 12