Package ‘EnhancedVolcano’

April 5, 2026

Type Package

Title Publication-ready volcano plots with enhanced colouring and labeling

Version 1.29.1

Maintainer Jared Andrews `<jared.andrews07@gmail.com>`

Description Volcano plots represent a useful way to visualise the results of differential expression analyses. Here, we present a highly-configurable function that produces publication-ready volcano plots. EnhancedVolcano will attempt to fit as many point labels in the plot window as possible, thus avoiding 'clogging' up the plot with labels that could not otherwise have been read. Other functionality allows the user to identify up to 4 different types of attributes in the same plot space via colour, shape, size, and shade parameter configurations.

License GPL-3

Encoding UTF-8

Depends ggplot2, ggrepel

Imports methods, scales, grid, grDevices

Suggests RUnit, ggrastr, BiocGenerics, knitr, DESeq2, pasilla, airway, org.Hs.eg.db, gridExtra, magrittr, rmarkdown

URL `https://github.com/kevinblighe/EnhancedVolcano`

biocViews RNASeq, GeneExpression, Transcription, DifferentialExpression, ImmunoOncology

VignetteBuilder knitr

RoxygenNote 7.3.2

git_url https://git.bioconductor.org/packages/EnhancedVolcano

git_branch devel

git_last_commit b1f351b

git_last_commit_date 2025-12-03

Repository Bioconductor 3.23

Date/Publication 2026-04-05

Author Kevin Blighe [aut],
Jared Andrews [cre, ctb] (ORCID: <https://orcid.org/0000-0002-0780-6248>),
Sharmila Rana [aut],
Emir Turkes [ctb],
Benjamin Ostendorf [ctb],
Andrea Grioni [ctb],
Myles Lewis [aut]

Contents

`EnhancedVolcano` *Publication-ready volcano plots with enhanced colouring and labeling.*

Description

Volcano plots represent a useful way to visualise the results of differential expression analyses. Here, we present a highly-configurable function that produces publication-ready volcano plots [@EnhancedVolcano]. `EnhancedVolcano` will attempt to fit as many variable names in the plot window as possible, thus avoiding 'clogging' up the plot with labels that could not otherwise have been read.

Usage

```
EnhancedVolcano(
  toptable,
  lab,
  x,
  y,
  selectLab = NULL,
 xlim = c(min(toptable[[x]], na.rm = TRUE) - 1.5, max(toptable[[x]], na.rm = TRUE) +
    1.5),
  ylim = c(0, max(-log10(toptable[[y]]), na.rm = TRUE) + 5),
  xlab = bquote(~Log[2] ~ "fold change"),
  ylab = bquote(~-Log[10] ~ italic(P)),
  axisLabSize = 18,
  title = "Volcano plot",
  subtitle = bquote(italic(EnhancedVolcano)),
  caption = paste0("total = ", nrow(toptable), " variables"),
  titleLabSize = 18,
```

```
  subtitleLabSize = 14,
  captionLabSize = 14,
  pCutoff = 1e-05,
  pCutoffCol = y,
  FCcutoff = 1,
  cutoffLineType = "longdash",
  cutoffLineCol = "black",
  cutoffLineWidth = 0.4,
  pointSize = 2,
  labSize = 5,
  labCol = "black",
  labFace = "plain",
  boxedLabels = FALSE,
  parseLabels = FALSE,
  shape = 19,
  shapeCustom = NULL,
  col = c("grey30", "forestgreen", "royalblue", "red2"),
  colCustom = NULL,
  colAlpha = 1/2,
  colGradient = NULL,
  colGradientBreaks = c(pCutoff, 1),
  colGradientLabels = c("0", "1.0"),
  colGradientLimits = c(0, 1),
 legendLabels = c("NS", expression(Log[2] ~ FC), "p-value", expression(p - value ~ and ~
    log[2] ~ FC)),
  legendPosition = "top",
  legendLabSize = 14,
  legendIconSize = 5,
  legendDropLevels = TRUE,
  encircle = NULL,
  encircleCol = "black",
  encircleFill = "pink",
  encircleAlpha = 3/4,
  encircleSize = 2.5,
  shade = NULL,
  shadeFill = "grey",
  shadeAlpha = 1/2,
  shadeSize = 0.01,
  shadeBins = 2,
  drawConnectors = FALSE,
  widthConnectors = 0.5,
  typeConnectors = "closed",
  endsConnectors = "first",
  lengthConnectors = unit(0.01, "npc"),
  colConnectors = "grey10",
  max.overlaps = 15,
  maxoverlapsConnectors = NULL,
  min.segment.length = 0,
```

```
  directionConnectors = "both",
  arrowheads = TRUE,
  hline = NULL,
  hlineType = "longdash",
  hlineCol = "black",
  hlineWidth = 0.4,
  vline = NULL,
  vlineType = "longdash",
  vlineCol = "black",
  vlineWidth = 0.4,
  gridlines.major = TRUE,
  gridlines.minor = TRUE,
  border = "partial",
  borderWidth = 0.8,
  borderColour = "black",
  raster = FALSE
)
```

Arguments

`toptable`	A data-frame of test statistics (if not, a data frame, an attempt will be made to convert it to one). Requires at least the following: column for variable names (can be rownames); a column for log2 fold changes; a column for nominal or adjusted p-value.
`lab`	A column name in `toptable` containing variable names. Can be `rownames(toptable)`.
`x`	A column name in `toptable` containing log2 fold changes.
`y`	A column name in `toptable` containing nominal or adjusted p-values.
`selectLab`	A vector containing a subset of lab.
`xlim`	Limits of the x-axis.
`ylim`	Limits of the y-axis.
`xlab`	Label for x-axis.
`ylab`	Label for y-axis.
`axisLabSize`	Size of x- and y-axis labels.
`title`	Plot title.
`subtitle`	Plot subtitle.
`caption`	Plot caption.
`titleLabSize`	Size of plot title.
`subtitleLabSize`	Size of plot subtitle.
`captionLabSize`	Size of plot caption.
`pCutoff`	Cut-off for statistical significance. A horizontal line will be drawn at -log10(pCutoff).
`pCutoffCol`	Column name of statistical significance values to be used as the cut-off. A typical usage situation would be to pass nominal [un-adjusted] p-values as ’y’, but adjusted p-values as pCutoffCol. In this way, a plot is generated via -log10(unadjusted p-value), but cut-offs based on adjusted p-values.

`FCcutoff`	Cut-off for absolute log2 fold-change. Vertical lines will be drawn at the negative and positive values of log2FCcutoff.
`cutoffLineType`	Line type for `FCcutoff` and `pCutoff` ('blank', 'solid', 'dashed', 'dotted', 'dotdash', 'longdash', 'twodash').
`cutoffLineCol`	Line colour for `FCcutoff` and `pCutoff`.
`cutoffLineWidth`	Line width for `FCcutoff` and `pCutoff`.
`pointSize`	Size of plotted points for each variable. Can be a single value or a vector of sizes.
`labSize`	Size of labels for each variable.
`labCol`	Colour of labels for each variable.
`labFace`	Font face of labels for each variable.
`boxedLabels`	Logical, indicating whether or not to draw labels in boxes.
`parseLabels`	Logical, indicating whether or not to parse expressions in labels
`shape`	Shape of the plotted points. Either a single value for all points, or 4 values corresponding to the default 4 legend labels specified by `legendLabels`.
`shapeCustom`	Named vector / key-value pairs that will over-ride the default shape scheme. The order must match that of `toptable`. Names / keys relate to groups / categories; values relate to shape encodings.
`col`	Colour shading for plotted points, corresponding to the default 4 legend labels specified by `legendLabels`.
`colCustom`	Named vector / key-value pairs that will over-ride the default colour scheme. The order must match that of `toptable`. Names / keys relate to groups / categories; values relate to colour.
`colAlpha`	Alpha for purposes of controlling colour transparency of variable points.
`colGradient`	If activated, over-rides the default discrete colour scheme and replaces it with a continous scheme that shades based on nominal or adjusted p-value specified by `y`. For example, c('red2', 'blue2').
`colGradientBreaks`	Break-points for the two colours specified by colGradient.
`colGradientLabels`	Labels for the break-points specified by colGradientBreaks.
`colGradientLimits`	Limits of the colour scheme specified by colGradient, i.e., max and min possible p-values.
`legendLabels`	Plot legend text labels.
`legendPosition`	Position of legend ('top', 'bottom', 'left', 'right').
`legendLabSize`	Size of plot legend text.
`legendIconSize`	Size of plot legend icons / symbols.
`legendDropLevels`	Logical, drop unused factor levels from legend.
`encircle`	A vector of variable names to encircle.

`encircleCol`	Colour of the encircled line.
`encircleFill`	Colour fill of the encircled region.
`encircleAlpha`	Alpha for purposes of controlling colour transparency of encircled region.
`encircleSize`	Line width of the encircled line.
`shade`	A vector of variable names to shade.
`shadeFill`	Colour of shaded regions.
`shadeAlpha`	Alpha for purposes of controlling colour transparency of shaded region.
`shadeSize`	Size of the shade contour lines.
`shadeBins`	Number of bins for the density of the shade.
`drawConnectors`	Logical, indicating whether or not to connect plot labels to their corresponding points by line connectors.
`widthConnectors`	Line width of connectors.
`typeConnectors`	Have the arrow head open ('open') or filled ('closed')?
`endsConnectors`	Which end of connectors to draw arrow head? ('last', 'first', 'both').
`lengthConnectors`	Length (size) of the connector arrowheads.
`colConnectors`	Line colour of connectors and line segments.
`max.overlaps`	Equivalent of max.overlaps in ggrepel. Set to 'Inf' to always display all labels when drawConnectors = TRUE.
`maxoverlapsConnectors`	See max.overlaps.
`min.segment.length`	When drawConnectors = TRUE, specifies the minimum length of the connector line segments.
`directionConnectors`	direction in which to draw connectors. 'both', 'x', or 'y'.
`arrowheads`	Logical, indicating whether or not to draw arrow heads or or just have straight lines.
`hline`	Draw one or more horizontal lines passing through this/these values on y-axis. For single values, only a single numerical value is necessary. For multiple lines, pass these as a vector, e.g., c(60,90).
`hlineType`	Line type for `hline` ('blank', 'solid', 'dashed', 'dotted', 'dotdash', 'longdash', 'twodash').
`hlineCol`	Colour of `hline`.
`hlineWidth`	Width of `hline`.
`vline`	Draw one or more vertical lines passing through this/these values on x-axis. For single values, only a single numerical value is necessary. For multiple lines, pass these as a vector, e.g., c(60,90).
`vlineType`	Line type for `vline` ('blank', 'solid', 'dashed', 'dotted', 'dotdash', 'longdash', 'twodash').

`vlineCol`	Colour of `vline`.
`vlineWidth`	Width of `vline`.
`gridlines.major`	Logical, indicating whether or not to draw major gridlines.
`gridlines.minor`	Logical, indicating whether or not to draw minor gridlines.
`border`	Add a border for just the x and y axes ('partial') or the entire plot grid ('full')?
`borderWidth`	Width of the border on the x and y axes.
`borderColour`	Colour of the border on the x and y axes.
`raster`	Logical, indicating whether to rasterize the geom_point layer. Requires installation of `ggrastr`.

Details

Volcano plots represent a useful way to visualise the results of differential expression analyses. Here, we present a highly-configurable function that produces publication-ready volcano plots [@EnhancedVolcano]. `EnhancedVolcano` will attempt to fit as many variable names in the plot window as possible, thus avoiding 'clogging' up the plot with labels that could not otherwise have been read.

Value

A `ggplot2` object.

Author(s)

Kevin Blighe <kevin@clinicalbioinformatics.co.uk>

Examples

```
library('pasilla')
pasCts <- system.file('extdata', 'pasilla_gene_counts.tsv',
  package='pasilla', mustWork=TRUE)
pasAnno <- system.file('extdata', 'pasilla_sample_annotation.csv',
  package='pasilla', mustWork=TRUE)
cts <- as.matrix(read.csv(pasCts,sep='\t',row.names='gene_id'))
coldata <- read.csv(pasAnno, row.names=1)
coldata <- coldata[,c('condition','type')]
rownames(coldata) <- sub('fb', '', rownames(coldata))
cts <- cts[, rownames(coldata)]
library('DESeq2')
dds <- DESeqDataSetFromMatrix(countData = cts,
  colData = coldata,
  design = ~ condition)

featureData <- data.frame(gene=rownames(cts))
mcols(dds) <- DataFrame(mcols(dds), featureData)
dds <- DESeq(dds)
res <- results(dds)
```

```
EnhancedVolcano(res,
  lab = rownames(res),
  x = 'log2FoldChange',
  y = 'pvalue',
  pCutoff = 10e-4,
  FCcutoff = 1.333,
  xlim = c(-5.5, 5.5),
  ylim = c(0, -log10(10e-12)),
  pointSize = 1.5,
  labSize = 2.5,
  title = 'DESeq2 results',
  subtitle = 'Differential expression',
  caption = 'FC cutoff, 1.333; p-value cutoff, 10e-4',
  legendPosition = "right",
  legendLabSize = 14,
  col = c('grey30', 'forestgreen', 'royalblue', 'red2'),
  colAlpha = 0.9,
  drawConnectors = TRUE,
  hline = c(10e-8),
  widthConnectors = 0.5)
```

`GeomEncircle` *Custom Geom for Encircling Points in ggplot2*

Description

Custom Geom for Encircling Points in ggplot2

Author(s)

Jared Andrews, heavily based on ggalt code from Ben Bolker (https://github.com/hrbrmstr/ggalt/blob/master/R/geom_encircle.r)

`geom_encircle` *Automatically enclose points in a polygon*

Description

Creates a smooth encircling polygon around a set of points using convex hull calculation and xspline smoothing. Useful for highlighting groups of points in scatter plots.

Usage

```
geom_encircle(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot.
`data`	The data to be displayed in this layer. If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot`.
`stat`	The statistical transformation to use on the data for this layer, as a string.
`position`	Position adjustment, either as a string, or the result of a call to a position adjustment function.
`na.rm`	If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.
`show.legend`	Logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped.
`inherit.aes`	If `FALSE`, overrides the default aesthetics, rather than combining with them.
`...`	Other arguments passed on to `layer`. These are often aesthetics, used to set an aesthetic to a fixed value, like `colour = "red"` or `size = 3`. They may also be parameters to the paired geom/stat. Additional parameters include: **s_shape** Controls the shape of the spline (default = 0.5). **s_open** Logical indicating whether the spline should be open (default = FALSE). **expand** Amount to expand the encircling polygon outward (default = 0.05). **spread** Spread factor for single or double point sets (default = 0.1).

Value

A ggplot2 layer that can be added to a plot.

Author(s)

Jared Andrews, heavily based on ggalt code from Ben Bolker (`https://github.com/hrbrmstr/ggalt/blob/master/R/geom_encircle.r`)

Examples

```
## Not run:
library(ggplot2)

d <- data.frame(x=c(1,1,2),y=c(1,2,2)*100)

gg <- ggplot(d,aes(x,y))
gg <- gg + scale_x_continuous(expand=c(0.5,1))
gg <- gg + scale_y_continuous(expand=c(0.5,1))

gg + geom_encircle(s_shape=1, expand=0) + geom_point()

gg + geom_encircle(s_shape=1, expand=0.1, colour="red") + geom_point()

gg + geom_encircle(s_shape=0.5, expand=0.1, colour="purple") + geom_point()

gg + geom_encircle(data=subset(d, x==1), colour="blue", spread=0.02) +
  geom_point()

gg + geom_encircle(data=subset(d, x==2), colour="cyan", spread=0.04) +
  geom_point()

gg <- ggplot(mpg, aes(displ, hwy))
gg + geom_encircle(data=subset(mpg, hwy>40)) + geom_point()
gg + geom_encircle(aes(group=manufacturer)) + geom_point()
gg + geom_encircle(aes(group=manufacturer,fill=manufacturer),alpha=0.4)+
       geom_point()
gg + geom_encircle(aes(group=manufacturer,colour=manufacturer))+
       geom_point()

ss <- subset(mpg,hwy>31 & displ<2)

gg + geom_encircle(data=ss, colour="blue", s_shape=0.9, expand=0.07) +
  geom_point() + geom_point(data=ss, colour="blue")

## End(Not run)
```

Index