

# Package ‘esetVis’

January 27, 2025

**Type** Package

**Title** Visualizations of expressionSet Bioconductor object

**Version** 1.32.0

**Date** 2023-12-15

**Author** Laure Cougnaud <laure.cougnaud@openanalytics.eu>

**Maintainer** Laure Cougnaud <laure.cougnaud@openanalytics.eu>

**Description** Utility functions for visualization of expressionSet (or SummarizedExperiment) Bioconductor object, including spectral map, tsne and linear discriminant analysis. Static plot via the ggplot2 package or interactive via the ggvis or rbokeh packages are available.

**Imports** mpm, hexbin, Rtsne, MLP, grid, Biobase, MASS, stats, utils, grDevices, methods

**Suggests** ggplot2, ggvis, plotly, ggrepel, knitr, rmarkdown, ALL, hgu95av2.db, AnnotationDbi, pander, SummarizedExperiment, GO.db

**biocViews** Visualization, DataRepresentation, DimensionReduction, PrincipalComponent, Pathways

**VignetteBuilder** knitr

**License** GPL-3

**NeedsCompilation** no

**RoxygenNote** 7.2.3

**git\_url** <https://git.bioconductor.org/packages/esetVis>

**git\_branch** RELEASE\_3\_20

**git\_last\_commit** e3637dd

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2025-01-27

## Contents

characterORexpressionOrCall-class . . . . .	2
esetLda . . . . .	2
esetPlot-class . . . . .	7

esetPlotInteractive-class . . . . .	10
esetPlotWrapper . . . . .	10
esetSpectralMap . . . . .	15
esetTsne . . . . .	21
formatManualScale . . . . .	24
formatOutput . . . . .	25
getAxesLimits . . . . .	25
getCoordGeneSets . . . . .	26
getDataPlotSamplesWithAnnotation . . . . .	27
getGeneSetsForPlot . . . . .	27
getMethodsInputObjectEsetVis . . . . .	29
getTopElements . . . . .	29
ggPlotEset . . . . .	30
ggplotEsetPlot-class . . . . .	31
ggvisEsetPlot-class . . . . .	32
ggvisPlotEset . . . . .	32
plotEset . . . . .	33
plotlyEsetPlot-class . . . . .	33
plotlyPlotEset . . . . .	34
plotTopElements . . . . .	34
setFixElement . . . . .	35
setManualScale . . . . .	36
simpleCap . . . . .	36
varToFm . . . . .	37

## Index 38

---

character0RexpressionOrCall-class  
*S4 Class Union with character/expression/call*

---

### Description

This is used for the definition of the title/axes labels for the ggplot2 version

---

esetLda *plot a biplot of a linear discriminant analysis of an eSet object*

---

### Description

esetLda reduces the dimension of the data contained in the [eSet](#) via a linear discriminant analysis on the specified grouping variable with the `lda` function and plot the subsequent biplot, possibly with sample annotation and gene annotation contained in the eSet.

**Usage**

```

esetLda(
  eset,
  ldaVar,
  psids = 1:nrow(eset),
  dim = c(1, 2),
  colorVar = character(),
  color = if (length(colorVar) == 0) "black" else character(),
  shapeVar = character(),
  shape = if (length(shapeVar) == 0) 15 else numeric(),
  sizeVar = character(),
  size = if (length(sizeVar) == 0) {
    ifelse(typePlot[1] == "interactive" &&
      packageInteractivity[1] == "plotly", 20, 2.5)
  } else {
    numeric()
  },
  sizeRange = numeric(),
  alphaVar = character(),
  alpha = if (length(alphaVar) == 0) 1 else numeric(),
  alphaRange = numeric(),
  title = "",
  symmetryAxes = c("combine", "separate", "none"),
  packageTextLabel = c("ggrepel", "ggplot2"),
  cloudGenes = TRUE,
  cloudGenesColor = "black",
  cloudGenesNBins = sqrt(length(psids)),
  cloudGenesIncludeLegend = FALSE,
  cloudGenesTitleLegend = "nGenes",
  topGenes = 10,
  topGenesCex = ifelse(typePlot[1] == "interactive" && packageInteractivity[1] ==
    "plotly", 10, 2.5),
  topGenesVar = character(),
  topGenesJust = c(0.5, 0.5),
  topGenesColor = "black",
  topSamples = 10,
  topSamplesCex = ifelse(typePlot[1] == "interactive" && packageInteractivity[1] ==
    "plotly", 10, 2.5),
  topSamplesVar = character(),
  topSamplesJust = c(0.5, 0.5),
  topSamplesColor = "black",
  geneSets = list(),
  geneSetsVar = character(),
  geneSetsMaxNChar = numeric(),
  topGeneSets = 10,
  topGeneSetsCex = ifelse(typePlot[1] == "interactive" && packageInteractivity[1] ==
    "plotly", 10, 2.5),
  topGeneSetsJust = c(0.5, 0.5),
  topGeneSetsColor = "black",
  includeLegend = TRUE,
  includeLineOrigin = TRUE,
  typePlot = c("static", "interactive"),

```

```

packageInteractivity = c("plotly", "ggvis"),
figInteractiveSize = c(600, 400),
ggvisAdjustLegend = TRUE,
interactiveTooltip = TRUE,
interactiveTooltipExtraVars = character(),
returnAnalysis = FALSE,
returnEsetPlot = FALSE
)

```

## Arguments

eset	expressionSet (or SummarizedExperiment) object with data
ldaVar	name of variable (in varLabels of the eset) used for grouping for lda
psids	featureNames of genes to include in the plot, all by default
dim	dimensions of the analysis to represent, first two dimensions by default
colorVar	name of variable (in varLabels of the eset) used for coloring, empty by default
color	character or factor with specified color(s) for the points, replicated if needed. This is used only if colorVar is empty. By default: 'black' if colorVar is not specified and default ggplot palette otherwise
shapeVar	name of variable (in varLabels of the eset) used for the shape, empty by default
shape	character or factor with specified shape(s) (pch) for the points, replicated if needed. This is used only if shapeVar is empty. By default: '15' (filled square) if shapeVar is not specified and default ggplot shape(s) otherwise
sizeVar	name of variable (in varLabels of the eset) used for the size, empty by default
size	character or factor with specified size(s) (cex) for the points, replicated if needed. This is used only if sizeVar is empty. By default: '2.5' if sizeVar is not specified (20 for a plotly plot) and default ggplot size(s) otherwise
sizeRange	size (cex) range used in the plot, possible only if the sizeVar is 'numeric' or 'integer'
alphaVar	name of variable (in varLabels of the eset) used for the transparency, empty by default. This parameter is currently only available for static plot and ggvis (only numeric in this case).
alpha	character or factor with specified transparency(s) for the points, replicated if needed. This is used only if shapeVar is empty. By default: '1' if alphaVar is not specified and default ggplot alpha otherwise This parameter is currently only available for static and ggvis.
alphaRange	transparency (alpha) range used in the plot, possible only if the alphaVar is 'numeric' or 'integer' This parameter is currently only available for static and ggvis plot.
title	plot title, "" by default
symmetryAxes	set symmetry for axes, either: <ul style="list-style-type: none"> <li>'combine' (by default): both axes are symmetric and with the same limits</li> <li>'separate': each axis is symmetric and has its own limits</li> <li>'none': axes by default (plot limits)</li> </ul>
packageTextLabel	package used to label the outlying genes/samples/gene sets, either ggrepel (by default, only used if package ggrepel is available), or ggplot2

cloudGenes	logical, if TRUE (by default), include the cloud of genes in the plot
cloudGenesColor	if cloudGenes is TRUE, color for the cloud of genes, black by default
cloudGenesNBins	number of bins to used for the clouds of genes, by default the square root of the number of genes
cloudGenesIncludeLegend	logical, if TRUE (FALSE by default) include the legend for the cloud of genes (in the top position if multiple legends)
cloudGenesTitleLegend	string with title for the legend for the cloud of genes 'nGenes' by default
topGenes	numeric indicating which percentile (if <1) or number (if >=1) of genes most distant to the origin of the plot to annotate, by default: 10 genes are selected If no genes should be annotated, set this parameter to 0 Currently only available for static plot.
topGenesCex	cex for gene annotation (used when topGenes > 0)
topGenesVar	variable of the featureData used to label the genes, by default: empty, the featureNames are used for labelling (used when topGenes > 0)
topGenesJust	text justification for the genes (used when topGenes > 0 and if packageTextLabel is ggplot2), by default: c(0.5, 0.5) so centered
topGenesColor	text color for the genes (used when topGenes > 0), black by default
topSamples	numeric indicating which percentile (if <1) or number (if >=1) of samples most distant to the origin of the plot to annotate, by default: 10 samples are selected If no samples should be annotated, set this parameter to 0. Currently available for static plot.
topSamplesCex	cex for sample annotation (used when topSamples > 0)
topSamplesVar	variable of the phenoData used to label the samples, by default: empty, the sampleNames are used for labelling (used when topSamples > 0)
topSamplesJust	text justification for the samples (used when topSamples > 0 and if packageTextLabel is ggplot2), by default: c(0.5, 0.5) so centered
topSamplesColor	text color for the samples (used when topSamples > 0), black by default
geneSets	list of gene sets/pathways, each containing identifiers of genes contained in the set. E.g. pathways from Gene Ontology databases output from the <a href="#">getGeneSetsForPlot</a> function or any custom list of pathways. The genes identifiers should correspond to the variable geneSetsVar contained in the phenoData, if not specified the featureNames are used. If several gene sets have the same name, they will be combine to extract the top gene sets.
geneSetsVar	variable of the featureData used to match the genes contained in geneSets, most probably ENTREZID, if not specified the featureNames of the eSet are used Only used when topGeneSets > 0 and the parameter geneSets is specified.
geneSetsMaxNChar	maximum number of characters for pathway names, by default keep entire names Only used when topGeneSets > 0 and the parameter geneSets is specified. If returnAnalysis is set to TRUE and geneSetsMaxNChar specified, the top pathways will be returned in the output object, named with the identifiers used in the plot (so with maximum geneSetsMaxNChar number of characters)

<code>topGeneSets</code>	numeric indicating which percentile (if $\leq 1$ ) or number (if $> 1$ ) of gene sets most distant to the origin of the plot to annotate, by default: 10 gene sets are selected. If no gene sets should be annotated, set this parameter to 0. Currently available for static plot. Only used when <code>topGeneSets &gt; 0</code> and the parameter <code>geneSets</code> is specified.
<code>topGeneSetsCex</code>	<code>cex</code> for gene sets annotation. Only used when <code>topGeneSets &gt; 0</code> and the parameter <code>geneSets</code> is specified.
<code>topGeneSetsJust</code>	text justification for the gene sets by default: <code>c(0.5, 0.5)</code> so centered. Only used when <code>topGeneSets &gt; 0</code> , the parameter <code>geneSets</code> is specified and if <code>packageTextLabel</code> is <code>ggplot2</code> .
<code>topGeneSetsColor</code>	color for the gene sets (used when <code>topGeneSets &gt; 0</code> and <code>geneSets</code> is specified), black by default. Only used when <code>topGeneSets &gt; 0</code> and the parameter <code>geneSets</code> is specified.
<code>includeLegend</code>	logical if TRUE (by default) include a legend, otherwise not
<code>includeLineOrigin</code>	if TRUE (by default) include vertical line at $x = 0$ and horizontal line at $y = 0$
<code>typePlot</code>	type of the plot returned, either 'static' (static) or 'interactive' (potentially interactive)
<code>packageInteractivity</code>	if <code>typePlot</code> is 'interactive', package used for interactive plot, either 'plotly' (by default) (by default) or 'ggvis'.
<code>figInteractiveSize</code>	vector containing the size of the interactive plot, as [width, height] by default: <code>c(600, 400)</code> . This is passed to the width and height parameters of: <ul style="list-style-type: none"> <li>• for plotly plots: the <code>ggplotly</code> function</li> <li>• for ggvis plots: the <code>ggvis::set_options</code> function</li> </ul>
<code>ggvisAdjustLegend</code>	logical, if TRUE (by default) adjust the legends in <code>ggvis</code> to avoid overlapping legends when multiple legends
<code>interactiveTooltip</code>	logical, if TRUE, add hover functionality showing sample annotation (variables used in the plot) in the plot
<code>interactiveTooltipExtraVars</code>	name of extra variable(s) (in <code>varLabels</code> of the <code>eset</code> ) to add in <code>plotlyEsetPlot</code> to label the samples, empty by default
<code>returnAnalysis</code>	logical, if TRUE (FALSE by default), return also the output of the analysis, and the outlying samples in the <code>topElements</code> element if any, otherwise only the plot object
<code>returnEsetPlot</code>	logical, if TRUE return also the <code>esetPlot</code> object

## Value

if `returnAnalysis` is TRUE, return a list:

- `analysis`: output of the spectral map analysis, whose parameters can be given as input to the `esetPlotWrapper` function
  - `dataPlotSamples`: coordinates of the samples

- dataPlotGenes: coordinates of the genes
- esetUsed: expressionSet used in the plot
- topElements: list with top outlying elements if any, possibly genes, samples and gene sets
- plot: the plot output

otherwise return only the plot

### Author(s)

Laure Cougnaud

### References

Fisher, R. A. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7 (2), 179–188

### See Also

the function used internally: [lda](#)

### Examples

```
# load data
library(ALL)
data(ALL)

# specify several variables in ldaVar (this might take a few minutes to run...)

# sample subsetting: currently cannot deal with missing values
samplesToRemove <- which(apply(pData(ALL)[, c("sex", "BT")], 1, anyNA))

# extract random features, because analysis is quite time consuming
retainedFeatures <- sample(featureNames(ALL), size = floor(nrow(ALL)/5))

# create the plot
esetLda(eset = ALL[retainedFeatures, -samplesToRemove],
        ldaVar = "BT", colorVar = "BT", shapeVar = "sex", sizeVar = "age",
        title = "Linear discriminant analysis on the ALL dataset")
```

---

esetPlot-class

*An S4 class to represent esetPlot object expressionSet with visualization data from dimension-reduction methods*

---

### Description

Constructor of the [esetPlot](#) class

### Usage

```
## S4 method for signature 'esetPlot'
initialize(Object, ...)
```

**Arguments**

.Object            [esetPlot](#) object  
 ...                additional class arguments

**Value**

S4 object of class `esetPlot`

**Slots**

`dataPlotSamples` `data.frame` with columns 'X', 'Y' with coordinates for the samples and with rownames which should correspond and be in the same order as the `sampleNames` of `esetUsed`

`dataPlotGenes` `data.frame` with two columns 'X' and 'Y' with coordinates for the genes

`eset` `expressionSet` (or `SummarizedExperiment`) object with data

`colorVar` name of variable (in `varLabels` of the `eset`) used for coloring, empty by default

`color` character or factor with specified color(s) for the points, replicated if needed. This is used only if `colorVar` is empty. By default: 'black' if `colorVar` is not specified and default `ggplot` palette otherwise

`shapeVar` name of variable (in `varLabels` of the `eset`) used for the shape, empty by default

`shape` character or factor with specified shape(s) (`pch`) for the points, replicated if needed. This is used only if `shapeVar` is empty. By default: '15' (filled square) if `shapeVar` is not specified and default `ggplot` shape(s) otherwise

`sizeVar` name of variable (in `varLabels` of the `eset`) used for the size, empty by default

`size` size character or factor with specified size(s) (`cex`) for the points, replicated if needed. This is used only if `sizeVar` is empty. By default: '2.5' if `sizeVar` is not specified and default `ggplot` size(s) otherwise

`sizeRange`, `size` (`cex`) range used in the plot, possible only if the `sizeVar` is 'numeric' or 'integer'

`alphaVar` name of variable (in `varLabels` of the `eset`) used for the transparency, empty by default.

`alpha` alpha character or factor with specified transparency(s) for the points, replicated if needed. This is used only if `alphaVar` is empty. By default: '1' if `alphaVar` is not specified and default `ggplot` alpha otherwise.

`alphaRange` transparency (alpha) range used in the plot, possible only if the `alphaVar` is 'numeric' or 'integer'

`symmetryAxes` set symmetry for axes, either:

- 'combine' (by default): both axes are symmetric and with the same limits
- 'separate': each axis is symmetric and has its own limits
- 'none': axes by default (plot limits)

`cloudGenes` logical, if TRUE (by default), include the cloud of genes in the spectral map

`cloudGenesColor` if `cloudGenes` is TRUE, color for the cloud of genes, black by default

`cloudGenesNBins` number of bins to used for the clouds of genes, by default the square root of the number of genes

`cloudGenesIncludeLegend` logical, if TRUE (FALSE by default) include the legend for the cloud of genes (in the top position if multiple legends)

`cloudGenesTitleLegend` string with title for the legend for the cloud of genes 'nGenes' by default

`packageTextLabel` package used to label the outlying genes/samples/gene sets, either `ggrepel` (by default, only used if package `ggrepel` is available), or `ggplot2`



- `topGenes` numeric indicating which percentile (if  $<1$ ) or number (if  $\geq 1$ ) of genes most distant to the origin of the plot to annotate, by default: 10 genes are selected If no genes should be annotated, set this parameter to 0 Currently only available for static plot.
- `topGenesCex` cex for gene annotation (used when `topGenes > 0`)
- `topGenesVar` variable of the `featureData` used to label the genes, by default: empty, the `featureNames` are used for labelling (used when `topGenes > 0`)
- `topGenesJust` text justification for the genes (used when `topGenes > 0` and if `packageTextLabel` is `ggplot2`), by default: `c(0.5, 0.5)` so centered
- `topGenesColor` text color for the genes (used when `topGenes > 0`), black by default
- `topSamples` numeric indicating which percentile (if  $<1$ ) or number (if  $\geq 1$ ) of samples most distant to the origin of the plot to annotate, by default: 10 samples are selected If no samples should be annotated, set this parameter to 0. Currently available for static plot.
- `topSamplesCex` cex for sample annotation (used when `topSamples > 0`)
- `topSamplesVar` variable of the `phenoData` used to label the samples, by default: empty, the `sampleNames` are used for labelling (used when `topSamples > 0`)
- `topSamplesJust` text justification for the samples (used when `topSamples > 0` and if `packageTextLabel` is `ggplot2`), by default: `c(0.5, 0.5)` so centered
- `topSamplesColor` text color for the samples (used when `topSamples > 0`), black by default
- `geneSets` list of gene sets/pathways, each containing identifiers of genes contained in the set. E.g. pathways from Gene Ontology databases output from the `getGeneSetsForPlot` function or any custom list of pathways. The genes identifiers should correspond to the variable `geneSetsVar` contained in the `phenoData`, if not specified the `featureNames` are used. If several gene sets have the same name, they will be combine to extract the top gene sets.
- `geneSetsVar` variable of the `featureData` used to match the genes contained in `geneSets`, most probably `ENTREZID`, if not specified the `featureNames` of the `eSet` are used Only used when `topGeneSets > 0` and the parameter `geneSets` is specified.
- `geneSetsMaxNChar` maximum number of characters for pathway names, by default keep entire names Only used when `topGeneSets > 0` and the parameter `geneSets` is specified.
- `topGeneSets` numeric indicating which percentile (if  $\leq 1$ ) or number (if  $> 1$ ) of gene sets most distant to the origin of the plot to annotate, by default: 10 gene sets are selected If no gene sets should be annotated, set this parameter to 0. Currently available for static plot. Only used when `topGeneSets > 0` and the parameter `geneSets` is specified.
- `topGeneSetsCex` cex for gene sets annotation Only used when `topGeneSets > 0` and the parameter `geneSets` is specified.
- `topGeneSetsJust` text justification for the gene sets by default: `c(0.5, 0.5)` so centered Only used when `topGeneSets > 0`, the parameter `geneSets` is specified and if `packageTextLabel` is `ggplot2`.
- `topGeneSetsColor` color for the gene sets (used when `topGeneSets > 0` and `geneSets` is specified), black by default Only used when `topGeneSets > 0` and the parameter `geneSets` is specified.
- `includeLegend` logical if `TRUE` (by default) include a legend, otherwise not
- `includeLineOrigin` if `TRUE` (by default) include vertical line at `x = 0` and horizontal line at `y = 0`

---

eSetPlotInteractive-class

*a S4 class to represent interactive plots*

---

### Description

a S4 class to represent interactive plots

### Value

S4 object of class eSetPlotInteractive

### Slots

`includeTooltip` logical, if TRUE, add hoover functionality showing sample annotation (variables used in the plot) in the plot

`tooltipVars` name of extra phenotypic variable(s) to add in `plotlyESetPlot` to label the samples

`sizePlot` vector containing the size of the interactive plot, as [width, height], by default: `c(600, 400)`.

`title` string plot title, " by default

`xlab` string label for the x axis

`ylab` string label for the y axis

### Author(s)

Laure Cougnaud

---

eSetPlotWrapper

*wrapper for biplot of features/samples contained in a eSet object*

---

### Description

Wrapper function used for all plots of the visualizations contained in the package.

### Usage

```
eSetPlotWrapper(
  dataPlotSamples,
  dataPlotGenes = data.frame(),
  eSetUsed,
  xlab = "",
  ylab = "",
  colorVar = character(0),
  color = if (length(colorVar) == 0) "black" else character(0),
  shapeVar = character(0),
  shape = if (length(shapeVar) == 0) 15 else numeric(0),
  sizeVar = character(0),
  size = if (length(sizeVar) == 0) {
```

```

        ifelse(typePlot[1] == "interactive" &&
              packageInteractivity[1] == "plotly", 20, 2.5)
    } else {
        numeric()
    },
    sizeRange = numeric(0),
    alphaVar = character(0),
    alpha = if (length(alphaVar) == 0) 1 else numeric(0),
    alphaRange = numeric(0),
    title = "",
    symmetryAxes = c("combine", "separate", "none"),
    cloudGenes = TRUE,
    cloudGenesColor = "black",
    cloudGenesNBins = if (nrow(dataPlotGenes) > 0) sqrt(nrow(dataPlotGenes)) else numeric(),
    cloudGenesIncludeLegend = FALSE,
    cloudGenesTitleLegend = "nGenes",
    packageTextLabel = c("ggrepel", "ggplot2"),
    topGenes = 10,
    topGenesCex = ifelse(typePlot[1] == "interactive" && packageInteractivity[1] ==
                        "plotly", 10, 2.5),
    topGenesVar = character(0),
    topGenesJust = c(0.5, 0.5),
    topGenesColor = "black",
    topSamples = 10,
    topSamplesCex = 2.5,
    topSamplesVar = character(0),
    topSamplesJust = c(0.5, 0.5),
    topSamplesColor = "black",
    geneSets = list(),
    geneSetsVar = character(0),
    geneSetsMaxNChar = numeric(0),
    topGeneSets = 10,
    topGeneSetsCex = 2.5,
    topGeneSetsJust = c(0.5, 0.5),
    topGeneSetsColor = "black",
    includeLegend = TRUE,
    includeLineOrigin = TRUE,
    typePlot = c("static", "interactive"),
    figInteractiveSize = c(600, 400),
    ggvisAdjustLegend = TRUE,
    interactiveTooltip = TRUE,
    interactiveTooltipExtraVars = character(0),
    packageInteractivity = c("plotly", "ggvis"),
    returnTopElements = FALSE,
    returnEsetPlot = FALSE
)

```

### Arguments

`dataPlotSamples`

data.frame with columns 'X', 'Y' with coordinates for the samples and with rownames which should correspond and be in the same order as the sample-Names of `esetUsed`

dataPlotGenes	data.frame with two columns 'X' and 'Y' with coordinates for the genes
esetUsed	expressionSet (or SummarizedExperiment) object with data
xlab	label for the x axis
ylab	label for the y axis
colorVar	name of variable (in varLabels of the eset) used for coloring, empty by default
color	character or factor with specified color(s) for the points, replicated if needed. This is used only if colorVar is empty. By default: 'black' if colorVar is not specified and default ggplot palette otherwise
shapeVar	name of variable (in varLabels of the eset) used for the shape, empty by default
shape	character or factor with specified shape(s) (pch) for the points, replicated if needed. This is used only if shapeVar is empty. By default: '15' (filled square) if shapeVar is not specified and default ggplot shape(s) otherwise
sizeVar	name of variable (in varLabels of the eset) used for the size, empty by default
size	character or factor with specified size(s) (cex) for the points, replicated if needed. This is used only if sizeVar is empty. By default: '2.5' if sizeVar is not specified (20 for a plotly plot) and default ggplot size(s) otherwise
sizeRange	size (cex) range used in the plot, possible only if the sizeVar is 'numeric' or 'integer'
alphaVar	name of variable (in varLabels of the eset) used for the transparency, empty by default. This parameter is currently only available for static plot and ggvis (only numeric in this case).
alpha	character or factor with specified transparency(s) for the points, replicated if needed. This is used only if shapeVar is empty. By default: '1' if alphaVar is not specified and default ggplot alpha otherwise This parameter is currently only available for static and ggvis.
alphaRange	transparency (alpha) range used in the plot, possible only if the alphaVar is 'numeric' or 'integer' This parameter is currently only available for static and ggvis plot.
title	plot title, "" by default
symmetryAxes	set symmetry for axes, either: <ul style="list-style-type: none"> <li>• 'combine' (by default): both axes are symmetric and with the same limits</li> <li>• 'separate': each axis is symmetric and has its own limits</li> <li>• 'none': axes by default (plot limits)</li> </ul>
cloudGenes	logical, if TRUE (by default), include the cloud of genes in the plot
cloudGenesColor	if cloudGenes is TRUE, color for the cloud of genes, black by default
cloudGenesNBins	number of bins to used for the clouds of genes, by default the square root of the number of genes
cloudGenesIncludeLegend	logical, if TRUE (FALSE by default) include the legend for the cloud of genes (in the top position if multiple legends)
cloudGenesTitleLegend	string with title for the legend for the cloud of genes 'nGenes' by default
packageTextLabel	package used to label the outlying genes/samples/gene sets, either ggrepel (by default, only used if package ggrepel is available), or ggplot2

topGenes	numeric indicating which percentile (if <1) or number (if >=1) of genes most distant to the origin of the plot to annotate, by default: 10 genes are selected If no genes should be annotated, set this parameter to 0 Currently only available for static plot.
topGenesCex	cex for gene annotation (used when topGenes > 0)
topGenesVar	variable of the featureData used to label the genes, by default: empty, the featureNames are used for labelling (used when topGenes > 0)
topGenesJust	text justification for the genes (used when topGenes > 0 and if packageTextLabel is ggplot2), by default: c(0.5, 0.5) so centered
topGenesColor	text color for the genes (used when topGenes > 0), black by default
topSamples	numeric indicating which percentile (if <1) or number (if >=1) of samples most distant to the origin of the plot to annotate, by default: 10 samples are selected If no samples should be annotated, set this parameter to 0. Currently available for static plot.
topSamplesCex	cex for sample annotation (used when topSamples > 0)
topSamplesVar	variable of the phenoData used to label the samples, by default: empty, the sampleNames are used for labelling (used when topSamples > 0)
topSamplesJust	text justification for the samples (used when topSamples > 0 and if packageTextLabel is ggplot2), by default: c(0.5, 0.5) so centered
topSamplesColor	text color for the samples (used when topSamples > 0), black by default
geneSets	list of gene sets/pathways, each containing identifiers of genes contained in the set. E.g. pathways from Gene Ontology databases output from the <a href="#">getGeneSetsForPlot</a> function or any custom list of pathways. The genes identifiers should correspond to the variable geneSetsVar contained in the phenoData, if not specified the featureNames are used. If several gene sets have the same name, they will be combine to extract the top gene sets.
geneSetsVar	variable of the featureData used to match the genes contained in geneSets, most probably ENTREZID, if not specified the featureNames of the eSet are used Only used when topGeneSets > 0 and the parameter geneSets is specified.
geneSetsMaxNChar	maximum number of characters for pathway names, by default keep entire names Only used when topGeneSets > 0 and the parameter geneSets is specified. If returnAnalysis is set to TRUE and geneSetsMaxNChar specified, the top pathways will be returned in the output object, named with the identifiers used in the plot (so with maximum geneSetsMaxNChar number of characters)
topGeneSets	numeric indicating which percentile (if <=1) or number (if >1) of gene sets most distant to the origin of the plot to annotate, by default: 10 gene sets are selected If no gene sets should be annotated, set this parameter to 0. Currently available for static plot. Only used when topGeneSets > 0 and the parameter geneSets is specified.
topGeneSetsCex	cex for gene sets annotation Only used when topGeneSets > 0 and the parameter geneSets is specified.
topGeneSetsJust	text justification for the gene sets by default: c(0.5, 0.5) so centered Only used when topGeneSets > 0, the parameter geneSets is specified and if packageTextLabel is ggplot2.

<code>topGeneSetsColor</code>	color for the gene sets (used when <code>topGeneSets &gt; 0</code> and <code>geneSets</code> is specified), black by default Only used when <code>topGeneSets &gt; 0</code> and the parameter <code>geneSets</code> is specified.
<code>includeLegend</code>	logical if TRUE (by default) include a legend, otherwise not
<code>includeLineOrigin</code>	if TRUE (by default) include vertical line at <code>x = 0</code> and horizontal line at <code>y = 0</code>
<code>typePlot</code>	type of the plot returned, either 'static' (static) or 'interactive' (potentially interactive)
<code>figInteractiveSize</code>	vector containing the size of the interactive plot, as [width, height] by default: <code>c(600, 400)</code> . This is passed to the <code>width</code> and <code>height</code> parameters of: <ul style="list-style-type: none"> <li>• for plotly plots: the <code>ggplotly</code> function</li> <li>• for ggvis plots: the <code>ggvis::set_options</code> function</li> </ul>
<code>ggvisAdjustLegend</code>	logical, if TRUE (by default) adjust the legends in ggvis to avoid overlapping legends when multiple legends
<code>interactiveTooltip</code>	logical, if TRUE, add hover functionality showing sample annotation (variables used in the plot) in the plot
<code>interactiveTooltipExtraVars</code>	name of extra variable(s) (in <code>varLabels</code> of the eset) to add in plotlyEsetPlot to label the samples, empty by default
<code>packageInteractivity</code>	if <code>typePlot</code> is 'interactive', package used for interactive plot, either 'plotly' (by default) (by default) or 'ggvis'.
<code>returnTopElements</code>	logical, if TRUE return also the top elements
<code>returnEsetPlot</code>	logical, if TRUE return also the <code>esetPlot</code> object

**Value**

if `typePlot` is:

- `static`:
  - if `returnTopElements` is TRUE, and top elements can be displayed, a list with:
    - \* 'topElements': the top elements labelled in the plot
    - \* 'plot': the ggplot object
  - otherwise, the ggplot object only
- `interactive`: a ggvis or plotly object, depending on the `packageInteractivity` parameter

**Author(s)**

Laure Cougnaud

**Examples**

```

library(ALL)
data(ALL)

## run one spectral map analysis

# create custom color palette
colorPalette <- c("dodgerblue", colorRampPalette(c("white", "dodgerblue2", "darkblue"))(5)[-1],
  "red", colorRampPalette(c("white", "red3", "darkred"))(5)[-1])

# run the analysis
# with 'returnAnalysis' set to TRUE to have all objects required for the esetPlotWrapper
outputEsetSPM <- esetSpectralMap(eset = ALL,
  title = "Acute lymphoblastic leukemia dataset \n Spectral map complete",
  colorVar = "BT", color = colorPalette,
  shapeVar = "sex", shape = 15:16,
  sizeVar = "age", sizeRange = c(2, 6),
  symmetryAxes = "separate",
  topGenes = 10, topGenesJust = c(1, 0), topGenesCex = 2, topGenesColor = "darkgrey",
  topSamples = 15, topSamplesVar = "cod", topSamplesColor = "black",
  topSamplesJust = c(1, 0), topSamplesCex = 3, returnAnalysis = TRUE)

# plot the biplot
print(outputEsetSPM$plot)

## re-call the plot function, to change some visualizations parameters
esetPlotWrapper(
  dataPlotSamples = outputEsetSPM$analysis$dataPlotSamples,
  dataPlotGenes = outputEsetSPM$analysis$dataPlotGenes,
  esetUsed = outputEsetSPM$analysis$esetUsed,
  title = paste("Acute lymphoblastic leukemia dataset \n Spectral map"),
  colorVar = "BT", color = colorPalette,
  shapeVar = "relapse",
  sizeVar = "age", sizeRange = c(2, 6),
  topSamplesVar = "cod", topGenesVar = "SYMBOL"
)

```

---

esetSpectralMap

*plot a spectral map biplot of an [eSet](#).*


---

**Description**

esetSpectralMap reduces the dimension of the data contained in the [eSet](#) with the [mpm](#) function and plot the subsequent biplot of the specified dimensions, possibly with gene and sample annotation contained in the [eSet](#). A spectral map with the default parameters is equivalent to a principal component analysis on the log-transformed, double centered and global normalized data (from documentation of the [mpm](#) function).

**Usage**

```

esetSpectralMap(
  eset,

```

```

psids = 1:nrow(eset),
dim = c(1, 2),
colorVar = character(),
color = if (length(colorVar) == 0) "black" else character(),
shapeVar = character(),
shape = if (length(shapeVar) == 0) 15 else numeric(),
sizeVar = character(),
size = if (length(sizeVar) == 0) {
  ifelse(typePlot[1] == "interactive" &&
    packageInteractivity[1] == "plotly", 20, 2.5)
} else {
  numeric()
},
sizeRange = numeric(),
alphaVar = character(),
alpha = if (length(alphaVar) == 0) 1 else numeric(),
alphaRange = numeric(),
title = "",
mpm.args = list(closure = "none", center = "double", normal = "global", row.weight =
  "mean", col.weight = "constant", logtrans = FALSE),
plot.mpm.args = list(scale = "uvc"),
symmetryAxes = c("combine", "separate", "none"),
packageTextLabel = c("ggrepel", "ggplot2"),
cloudGenes = TRUE,
cloudGenesColor = "black",
cloudGenesNBins = sqrt(length(psids)),
cloudGenesIncludeLegend = FALSE,
cloudGenesTitleLegend = "nGenes",
topGenes = 10,
topGenesCex = ifelse(typePlot[1] == "interactive" && packageInteractivity[1] ==
  "plotly", 10, 2.5),
topGenesVar = character(),
topGenesJust = c(0.5, 0.5),
topGenesColor = "black",
topSamples = 10,
topSamplesCex = ifelse(typePlot[1] == "interactive" && packageInteractivity[1] ==
  "plotly", 10, 2.5),
topSamplesVar = character(),
topSamplesJust = c(0.5, 0.5),
topSamplesColor = "black",
geneSets = list(),
geneSetsVar = character(),
geneSetsMaxNChar = numeric(),
topGeneSets = 10,
topGeneSetsCex = ifelse(typePlot[1] == "interactive" && packageInteractivity[1] ==
  "plotly", 10, 2.5),
topGeneSetsJust = c(0.5, 0.5),
topGeneSetsColor = "black",
includeLegend = TRUE,
includeLineOrigin = TRUE,
typePlot = c("static", "interactive"),
packageInteractivity = c("plotly", "ggvis"),

```



```

figInteractiveSize = c(600, 400),
ggvisAdjustLegend = TRUE,
interactiveTooltip = TRUE,
interactiveTooltipExtraVars = character(),
returnAnalysis = FALSE,
returnEsetPlot = FALSE
)

```

### Arguments

eset	expressionSet (or SummarizedExperiment) object with data
psids	featureNames of genes to include in the plot, all by default
dim	dimensions of the analysis to represent, first two dimensions by default
colorVar	name of variable (in varLabels of the eset) used for coloring, empty by default
color	character or factor with specified color(s) for the points, replicated if needed. This is used only if colorVar is empty. By default: 'black' if colorVar is not specified and default ggplot palette otherwise
shapeVar	name of variable (in varLabels of the eset) used for the shape, empty by default
shape	character or factor with specified shape(s) (pch) for the points, replicated if needed. This is used only if shapeVar is empty. By default: '15' (filled square) if shapeVar is not specified and default ggplot shape(s) otherwise
sizeVar	name of variable (in varLabels of the eset) used for the size, empty by default
size	character or factor with specified size(s) (cex) for the points, replicated if needed. This is used only if sizeVar is empty. By default: '2.5' if sizeVar is not specified (20 for a plotly plot) and default ggplot size(s) otherwise
sizeRange	size (cex) range used in the plot, possible only if the sizeVar is 'numeric' or 'integer'
alphaVar	name of variable (in varLabels of the eset) used for the transparency, empty by default. This parameter is currently only available for static plot and ggvis (only numeric in this case).
alpha	character or factor with specified transparency(s) for the points, replicated if needed. This is used only if shapeVar is empty. By default: '1' if alphaVar is not specified and default ggplot alpha otherwise This parameter is currently only available for static and ggvis.
alphaRange	transparency (alpha) range used in the plot, possible only if the alphaVar is 'numeric' or 'integer' This parameter is currently only available for static and ggvis plot.
title	plot title, "" by default
mpm.args	list with input parameters for the <code>mpm</code> function. The default value is: <code>list(closure = 'none', center = 'double', normal = 'global', 'row.weight' = 'mean', col.weight = 'constant', logtrans = FALSE)</code> . This assumes that the data are already in a log scale.
plot.mpm.args	list with input parameters for the <code>plot.mpm</code> function. The default value is: <code>list(scale = "uvc")</code> .
symmetryAxes	set symmetry for axes, either: <ul style="list-style-type: none"> <li>• 'combine' (by default): both axes are symmetric and with the same limits</li> <li>• 'separate': each axis is symmetric and has its own limits</li> </ul>

- 'none': axes by default (plot limits)

packageTextLabel	package used to label the outlying genes/samples/gene sets, either ggrepel (by default, only used if package ggrepel is available), or ggplot2
cloudGenes	logical, if TRUE (by default), include the cloud of genes in the plot
cloudGenesColor	if cloudGenes is TRUE, color for the cloud of genes, black by default
cloudGenesNBins	number of bins to used for the clouds of genes, by default the square root of the number of genes
cloudGenesIncludeLegend	logical, if TRUE (FALSE by default) include the legend for the cloud of genes (in the top position if multiple legends)
cloudGenesTitleLegend	string with title for the legend for the cloud of genes 'nGenes' by default
topGenes	numeric indicating which percentile (if <1) or number (if >=1) of genes most distant to the origin of the plot to annotate, by default: 10 genes are selected If no genes should be annotated, set this parameter to 0 Currently only available for static plot.
topGenesCex	cex for gene annotation (used when topGenes > 0)
topGenesVar	variable of the featureData used to label the genes, by default: empty, the featureNames are used for labelling (used when topGenes > 0)
topGenesJust	text justification for the genes (used when topGenes > 0 and if packageTextLabel is ggplot2), by default: c(0.5, 0.5) so centered
topGenesColor	text color for the genes (used when topGenes > 0), black by default
topSamples	numeric indicating which percentile (if <1) or number (if >=1) of samples most distant to the origin of the plot to annotate, by default: 10 samples are selected If no samples should be annotated, set this parameter to 0. Currently available for static plot.
topSamplesCex	cex for sample annotation (used when topSamples > 0)
topSamplesVar	variable of the phenoData used to label the samples, by default: empty, the sampleNames are used for labelling (used when topSamples > 0)
topSamplesJust	text justification for the samples (used when topSamples > 0 and if packageTextLabel is ggplot2), by default: c(0.5, 0.5) so centered
topSamplesColor	text color for the samples (used when topSamples > 0), black by default
geneSets	list of gene sets/pathways, each containing identifiers of genes contained in the set. E.g. pathways from Gene Ontology databases output from the <a href="#">getGeneSetsForPlot</a> function or any custom list of pathways. The genes identifiers should correspond to the variable geneSetsVar contained in the phenoData, if not specified the featureNames are used. If several gene sets have the same name, they will be combine to extract the top gene sets.
geneSetsVar	variable of the featureData used to match the genes contained in geneSets, most probably ENTREZID, if not specified the featureNames of the eSet are used Only used when topGeneSets > 0 and the parameter geneSets is specified.
geneSetsMaxNChar	maximum number of characters for pathway names, by default keep entire names Only used when topGeneSets > 0 and the parameter geneSets is specified.

If `returnAnalysis` is set to `TRUE` and `geneSetsMaxNChar` specified, the top pathways will be returned in the output object, named with the identifiers used in the plot (so with maximum `geneSetsMaxNChar` number of characters)

<code>topGeneSets</code>	numeric indicating which percentile (if $\leq 1$ ) or number (if $> 1$ ) of gene sets most distant to the origin of the plot to annotate, by default: 10 gene sets are selected. If no gene sets should be annotated, set this parameter to 0. Currently available for static plot. Only used when <code>topGeneSets</code> $> 0$ and the parameter <code>geneSets</code> is specified.
<code>topGeneSetsCex</code>	cex for gene sets annotation. Only used when <code>topGeneSets</code> $> 0$ and the parameter <code>geneSets</code> is specified.
<code>topGeneSetsJust</code>	text justification for the gene sets by default: <code>c(0.5, 0.5)</code> so centered. Only used when <code>topGeneSets</code> $> 0$ , the parameter <code>geneSets</code> is specified and if <code>packageTextLabel</code> is <code>ggplot2</code> .
<code>topGeneSetsColor</code>	color for the gene sets (used when <code>topGeneSets</code> $> 0$ and <code>geneSets</code> is specified), black by default. Only used when <code>topGeneSets</code> $> 0$ and the parameter <code>geneSets</code> is specified.
<code>includeLegend</code>	logical if <code>TRUE</code> (by default) include a legend, otherwise not
<code>includeLineOrigin</code>	if <code>TRUE</code> (by default) include vertical line at $x = 0$ and horizontal line at $y = 0$
<code>typePlot</code>	type of the plot returned, either 'static' (static) or 'interactive' (potentially interactive)
<code>packageInteractivity</code>	if <code>typePlot</code> is 'interactive', package used for interactive plot, either 'plotly' (by default) (by default) or 'ggvis'.
<code>figInteractiveSize</code>	vector containing the size of the interactive plot, as [width, height] by default: <code>c(600, 400)</code> . This is passed to the <code>width</code> and <code>height</code> parameters of: <ul style="list-style-type: none"> <li>• for plotly plots: the <code>ggplotly</code> function</li> <li>• for ggvis plots: the <code>ggvis::set_options</code> function</li> </ul>
<code>ggvisAdjustLegend</code>	logical, if <code>TRUE</code> (by default) adjust the legends in <code>ggvis</code> to avoid overlapping legends when multiple legends
<code>interactiveTooltip</code>	logical, if <code>TRUE</code> , add hover functionality showing sample annotation (variables used in the plot) in the plot
<code>interactiveTooltipExtraVars</code>	name of extra variable(s) (in <code>varLabels</code> of the <code>eset</code> ) to add in <code>plotlyEsetPlot</code> to label the samples, empty by default
<code>returnAnalysis</code>	logical, if <code>TRUE</code> ( <code>FALSE</code> by default), return also the output of the analysis, and the outlying samples in the <code>topElements</code> element if any, otherwise only the plot object
<code>returnEsetPlot</code>	logical, if <code>TRUE</code> return also the <code>esetPlot</code> object

**Value**

if `returnAnalysis` is `TRUE`, return a list:

- analysis: output of the spectral map analysis, can be given as input to the `esetPlotWrapper` function
  - `dataPlotSamples`: coordinates of the samples
  - `dataPlotGenes`: coordinates of the genes
  - `esetUsed`: expressionSet used in the plot
  - `axisLabels`: axes labels indicating percentage of variance explained by the selected axes
  - `axesContributionsPercentages`: percentages of variance explained by each axis (not only the ones specified in `dim`)
- `topElements`: list with top outlying elements if any, possibly genes, samples and gene sets
- `plot`: the plot output

otherwise return only the plot

### Author(s)

Laure Cougnaud

### References

Lewi, P.J. (1976). Spectral mapping, a technique for classifying biological activity profiles of chemical compounds. *Arzneimittel Forschung (Drug Research)*, 26, 1295–1300

### See Also

the function used internally: [mpm](#) and [spectralMap](#) for spectral map in base R graphics

### Examples

```
library(ALL)
data(ALL)

## complete example (most of the parameters are optional)
# create custom color palette
colorPalette <- c("dodgerblue", colorRampPalette(c("white", "dodgerblue2", "darkblue"))(5)[-1],
  "red", colorRampPalette(c("white", "red3", "darkred"))(5)[-1])
# plot the spectral map
print(esetSpectralMap(eset = ALL,
  title = "Acute lymphoblastic leukemia dataset \n Spectral map complete",
  colorVar = "BT", color = colorPalette,
  shapeVar = "sex", shape = 15:16,
  sizeVar = "age", sizeRange = c(2, 6),
  symmetryAxes = "separate",
  topGenes = 10, topGenesJust = c(1, 0), topGenesCex = 2, topGenesColor = "darkgrey",
  topSamples = 15, topSamplesVar = "cod", topSamplesColor = "black",
  topSamplesJust = c(1, 0), topSamplesCex = 3)
)

# see vignette for other examples, especially one with gene sets specification
```

---

esetTsne *plot a t-SNE of an eSet object*

---

## Description

esetTsne reduces the dimension of the data contained in the [eSet](#) via t-Distributed Stochastic Neighbor Embedding with the [Rtsne](#) function and plot the subsequent biplot, possibly with sample annotation contained in the eSet.

## Usage

```

esetTsne(
  eset,
  psids = 1:nrow(eset),
  trace = TRUE,
  colorVar = character(),
  color = if (length(colorVar) == 0) "black" else character(),
  shapeVar = character(),
  shape = if (length(shapeVar) == 0) 15 else numeric(),
  sizeVar = character(),
  size = if (length(sizeVar) == 0) {
    ifelse(typePlot[1] == "interactive" &&
      packageInteractivity[1] == "plotly", 20, 2.5)
  } else {
    numeric()
  },
  sizeRange = numeric(),
  alphaVar = character(),
  alpha = if (length(alphaVar) == 0) 1 else numeric(),
  alphaRange = numeric(),
  title = "",
  Rtsne.args = list(perplexity = floor((ncol(eset) - 1)/3), theta = 0.5, dims = 2,
    initial_dims = 50),
  fctTransformDataForInputTsne = NULL,
  symmetryAxes = c("combine", "separate", "none"),
  packageTextLabel = c("ggrepel", "ggplot2"),
  topSamples = 10,
  topSamplesCex = ifelse(typePlot[1] == "interactive" && packageInteractivity[1] ==
    "plotly", 10, 2.5),
  topSamplesVar = character(),
  topSamplesJust = c(0.5, 0.5),
  topSamplesColor = "black",
  includeLegend = TRUE,
  includeLineOrigin = TRUE,
  typePlot = c("static", "interactive"),
  packageInteractivity = c("plotly", "ggvis"),
  figInteractiveSize = c(600, 400),
  ggvisAdjustLegend = TRUE,
  interactiveTooltip = TRUE,
  interactiveTooltipExtraVars = character(),
  returnAnalysis = FALSE,

```

```

    returnEsetPlot = FALSE
  )

```

### Arguments

eset	expressionSet (or SummarizedExperiment) object with data
psids	featureNames of genes to include in the plot, all by default
trace	logical, if TRUE (by default), print some messages during tsne is running
colorVar	name of variable (in varLabels of the eset) used for coloring, empty by default
color	character or factor with specified color(s) for the points, replicated if needed. This is used only if colorVar is empty. By default: 'black' if colorVar is not specified and default ggplot palette otherwise
shapeVar	name of variable (in varLabels of the eset) used for the shape, empty by default
shape	character or factor with specified shape(s) (pch) for the points, replicated if needed. This is used only if shapeVar is empty. By default: '15' (filled square) if shapeVar is not specified and default ggplot shape(s) otherwise
sizeVar	name of variable (in varLabels of the eset) used for the size, empty by default
size	character or factor with specified size(s) (cex) for the points, replicated if needed. This is used only if sizeVar is empty. By default: '2.5' if sizeVar is not specified (20 for a plotly plot) and default ggplot size(s) otherwise
sizeRange	size (cex) range used in the plot, possible only if the sizeVar is 'numeric' or 'integer'
alphaVar	name of variable (in varLabels of the eset) used for the transparency, empty by default. This parameter is currently only available for static plot and ggvis (only numeric in this case).
alpha	character or factor with specified transparency(s) for the points, replicated if needed. This is used only if shapeVar is empty. By default: '1' if alphaVar is not specified and default ggplot alpha otherwise This parameter is currently only available for static and ggvis.
alphaRange	transparency (alpha) range used in the plot, possible only if the alphaVar is 'numeric' or 'integer' This parameter is currently only available for static and ggvis plot.
title	plot title, "" by default
Rtsne.args	arguments for the Rtsne function, by default: perplexite parameter = optimal number of neighbours, theta = speed/accuracy trade-off (increase for less accuracy), set to 0.0 for exact TSNE
fctTransformDataForInputTsne	function which transform the data in the eSet object before calling the <a href="#">Rtsne</a> function. This should be a function which takes a matrix as input and return a matrix, e.g. the dist function.
symmetryAxes	set symmetry for axes, either: <ul style="list-style-type: none"> <li>'combine' (by default): both axes are symmetric and with the same limits</li> <li>'separate': each axis is symmetric and has its own limits</li> <li>'none': axes by default (plot limits)</li> </ul>
packageTextLabel	package used to label the outlying genes/samples/gene sets, either ggrepel (by default, only used if package ggrepel is available), or ggplot2

<code>topSamples</code>	numeric indicating which percentile (if <1) or number (if >=1) of samples most distant to the origin of the plot to annotate, by default: 10 samples are selected. If no samples should be annotated, set this parameter to 0. Currently available for static plot.
<code>topSamplesCex</code>	cex for sample annotation (used when <code>topSamples &gt; 0</code> )
<code>topSamplesVar</code>	variable of the <code>phenoData</code> used to label the samples, by default: empty, the <code>sampleNames</code> are used for labelling (used when <code>topSamples &gt; 0</code> )
<code>topSamplesJust</code>	text justification for the samples (used when <code>topSamples &gt; 0</code> and if <code>packageTextLabel</code> is <code>ggplot2</code> ), by default: <code>c(0.5, 0.5)</code> so centered
<code>topSamplesColor</code>	text color for the samples (used when <code>topSamples &gt; 0</code> ), black by default
<code>includeLegend</code>	logical if TRUE (by default) include a legend, otherwise not
<code>includeLineOrigin</code>	if TRUE (by default) include vertical line at <code>x = 0</code> and horizontal line at <code>y = 0</code>
<code>typePlot</code>	type of the plot returned, either 'static' (static) or 'interactive' (potentially interactive)
<code>packageInteractivity</code>	if <code>typePlot</code> is 'interactive', package used for interactive plot, either 'plotly' (by default) (by default) or 'ggvis'.
<code>figInteractiveSize</code>	vector containing the size of the interactive plot, as [width, height] by default: <code>c(600, 400)</code> . This is passed to the width and height parameters of: <ul style="list-style-type: none"> <li>• for plotly plots: the <code>ggplotly</code> function</li> <li>• for ggvis plots: the <code>ggvis::set_options</code> function</li> </ul>
<code>ggvisAdjustLegend</code>	logical, if TRUE (by default) adjust the legends in <code>ggvis</code> to avoid overlapping legends when multiple legends
<code>interactiveTooltip</code>	logical, if TRUE, add hover functionality showing sample annotation (variables used in the plot) in the plot
<code>interactiveTooltipExtraVars</code>	name of extra variable(s) (in <code>varLabels</code> of the <code>eset</code> ) to add in <code>plotlyEsetPlot</code> to label the samples, empty by default
<code>returnAnalysis</code>	logical, if TRUE (FALSE by default), return also the output of the analysis, and the outlying samples in the <code>topElements</code> element if any, otherwise only the plot object
<code>returnEsetPlot</code>	logical, if TRUE return also the <code>esetPlot</code> object

## Value

if `returnAnalysis` is TRUE, return a list:

- `analysis`: output of the spectral map analysis, whose elements can be given to the `esetPlotWrapper` function
  - `dataPlotSamples`: coordinates of the samples
  - `esetUsed`: expressionSet used in the plot
- `topElements`: list with top outlying elements if any, possibly genes, samples and gene sets
- `plot`: the plot output

otherwise return only the plot

**Author(s)**

Laure Cougnaud

**References**

L.J.P. van der Maaten and G.E. Hinton (2008). Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research, 2579–2605

**See Also**

the function used internally: [Rtsne](#) or <http://homepage.tudelft.nl/19j49/t-SNE.html> for further explanations about this technique.

**Examples**

```
library(ALL)
data(ALL)

## complete example (most of the parameters are optional)

# create custom color palette
colorPalette <- c("dodgerblue", colorRampPalette(c("white", "dodgerblue2", "darkblue"))(5)[-1],
  "red", colorRampPalette(c("white", "red3", "darkred"))(5)[-1])

# create tsne
print(esetTsne(eset = ALL,
  title = "Acute lymphoblastic leukemia dataset \n Tsne complete",
  colorVar = "BT", color = colorPalette,
  shapeVar = "sex", shape = 15:16,
  sizeVar = "age", sizeRange = c(2, 6),
  symmetryAxes = "separate",
  topSamples = 15, topSamplesVar = "cod", topSamplesColor = "black",
  topSamplesJust = c(1, 0), topSamplesCex = 3)
)
```

---

formatManualScale	<i>extend manual scale values if required</i>
-------------------	---

---

**Description**

extend manual scale values if required

**Usage**

```
formatManualScale(x, valVar, nameVar)
```

**Arguments**

x	data.frame with nameVar
valVar	fixed value of variable of aesthetic
nameVar	name of variable for aesthetic



**Value**

vector of manual scales

**Author(s)**

Laure Cougnaud

---

formatOutput	<i>format output of plotEset function</i>
--------------	---

---

**Description**

format output of [plotEset](#) function

**Usage**

```
formatOutput(res, object, type, returnEsetPlot)
```

**Arguments**

res	result of specific <a href="#">plotEset</a> function
object	esetPlot object or extended class
type	string type of plot
returnEsetPlot	logical, should the object be returned in the output function?

**Value**

result

**Author(s)**

Laure Cougnaud

---

getAxesLimits	<i>generic for get axes limits</i>
---------------	------------------------------------

---

**Description**

generic for get axes limits

**Usage**

```
getAxesLimits(object)
```

```
## S4 method for signature 'esetPlot'  
getAxesLimits(object)
```

**Arguments**

object	<a href="#">plotEset</a> object
--------	---------------------------------

**Value**

matrix with limits for axes: columns x and y

**Author(s)**

Laure Cougnaud

---

getCoordGeneSets      *extract coordinates gene sets*

---

**Description**

extract coordinates gene sets

**Usage**

```
getCoordGeneSets(dataPlotGenes, geneSets, esetUsed, geneSetsVar = list())
```

**Arguments**

dataPlotGenes	data.frame with two columns 'X' and 'Y' with coordinates for the genes
geneSets	geneSets list of gene sets, e.g. gene pathways, output from the 'getGeneSets' function in MLP the genes IDs must correspond to the sampleNames in the eset
esetUsed	expressionSet (or SummarizedExperiment) object with data
geneSetsVar	variable of the featureData used to match the genes contained in geneSets, most probably ENTREZID, if NULL the featureNames of the eSet are used
...	Any parameters passed to the <a href="#">getTopElements</a>

**Value**

data.frame with two columns 'X' and 'Y' with coordinates for the gene sets

**Author(s)**

Laure Cougnaud

```
getDataPlotSamplesWithAnnotation
      get sample data for plot
```

**Description**

get sample data for plot

**Usage**

```
getDataPlotSamplesWithAnnotation(object)

## S4 method for signature 'eSetPlot'
getDataPlotSamplesWithAnnotation(object)

## S4 method for signature 'ggvisEsetPlot'
getDataPlotSamplesWithAnnotation(object)

## S4 method for signature 'plotlyEsetPlot'
getDataPlotSamplesWithAnnotation(object)
```

**Arguments**

object                    [plotEset](#) object

**Value**

data.frame with 'dataPlotSamples' binded with variables displayed in the plot

**Author(s)**

Laure Cougnaud

```
getGeneSetsForPlot        get gene sets for plot of eSet object.
```

**Description**

get and format gene sets to be used as geneSets for the functions: [eSetSpectralMap](#), [eSetLda](#), or [eSetPlotWrapper](#) Use the [getGeneSets](#) function to get the gene sets, combine all databases, and format the gene sets name if required.

**Usage**

```
getGeneSetsForPlot(
  entrezIdentifiers,
  species = "Human",
  geneSetSource = c("GOBP", "GOMF", "GOCC", "KEGG"),
  useDescription = TRUE,
  trace = TRUE
)
```

**Arguments**

<code>entrezIdentifiers</code>	string with Entrez Gene identifiers of the genes of interest
<code>species</code>	species to use, given to the <a href="#">getGeneSets</a> function
<code>geneSetSource</code>	gene set source, either 'GOBP', 'GOMF', 'GOCC' or 'KEGG'. Multiple choices are available
<code>useDescription</code>	logical, if TRUE (by default) use the description to label the gene sets, otherwise use the original gene set identifiers Function 'substr' is used.
<code>trace</code>	logical, if TRUE (by default) a few extra information are printed during the process

**Value**

list with gene sets, each element is a gene set and contains the ENTREZ IDs of the genes contained in this set. If `useDescription` is:

- FALSE: pathways are labelled with identifiers (Gene Ontology IDs for GOBP, GOMF and GOCC, KEGG IDs for KEGG)
- TRUE: pathways are labelled with gene sets descriptions

**Author(s)**

Laure Cougnaud

**See Also**

the function used internally: [getGeneSets](#)

**Examples**

```
# example dataset
library(ALL)
data(ALL)

# get gene annotation from probe IDs
library("hgu95av2.db")
probeIDs <- featureNames(ALL)
geneInfo <- select(hgu95av2.db, probeIDs, "ENTREZID", "PROBEID")

# get pathway annotation for the genes contained in the ALL dataset (can take a few minutes)
geneSets <- getGeneSetsForPlot(entrezIdentifiers = geneInfo$ENTREZID, species = "Human",
  geneSetSource = 'GOBP',
  useDescription = FALSE, trace = TRUE)
head(geneSets) # returns a pathway list of genes

# gene sets labelled with gene sets description
geneSets <- getGeneSetsForPlot(entrezIdentifiers = geneInfo$ENTREZID, species = "Human",
  geneSetSource = 'GOBP', useDescription = TRUE, trace = TRUE)
head(geneSets) # returns a pathway list of genes

# see also vignette for an example of the use of this function as input for the esetSpectralMap, esetLda or esetP
```

---

getMethodsInputObjectEsetVis

*wrapper to extract useful functions, depending if the object is an ExpressionSet or a SummarizedExperiment.*

---

### Description

This returns an error if x is not of the correct class. The package SummarizedExperiment should be available if x is of class SummarizedExperiment.

### Usage

```
getMethodsInputObjectEsetVis(x)
```

### Arguments

x                    object

### Value

if the object is an ExpressionSet or a SummarizedExperiment, returns a list with the functions specific of the class of x, and equivalent of the ExpressionSet functions: 'sampleNames', 'featureNames', 'fData', 'pData', 'exprs'

- sampleNames: sample names
- featureNames: feature names
- fData: feature annotation
- pData: sample annotation
- exprs: data matrix
- varLabels: sample annotation variables
- fvarLabels: feature annotation variables

### Author(s)

Laure Cougnaud

---

getTopElements

*create geom\_text object with top genes/sample/pathways*

---

### Description

create geom\_text object with top genes/sample/pathways

**Usage**

```

getTopElements(
  top,
  type = c("gene", "sample", "geneSets"),
  var = character(),
  dataPlotGenes = data.frame(),
  dataPlotSamples = data.frame(),
  esetUsed,
  geneSets = list(),
  geneSetsVar = character(),
  geneSetsMaxNChar = numeric()
)

```

**Arguments**

top	numeric, number of top elements
type	type of elements to plot, either 'gene', 'sample', or 'geneSets'
var	variable used to annotate the elements, only used for 'gene' and 'sample'
dataPlotGenes	data.frame with two columns 'X' and 'Y' with coordinates for the genes
dataPlotSamples	data.frame with two columns 'X' and 'Y' with coordinates for the samples
esetUsed	expressionSet (or SummarizedExperiment) object with data
geneSets	list of gene sets, e.g. gene pathways, output from the 'getGeneSets' function in MLP the genes IDs must correspond to the sampleNames in the eset. If several gene sets have the same name, they will be combine to extract the top gene sets.
geneSetsVar	variable of the featureData used to match the genes contained in geneSets, most probably ENTREZID, if not specified the featureNames of the eSet are used
geneSetsMaxNChar	maximum number of characters for pathway names, by default keep entire names
returnTopElements	logical if TRUE (FALSE by default) return the outlying elements

**Value**

Data.frame with coordinates and labels of the top elements

**Author(s)**

Laure Cougnaud

---

ggPlotEset

*visualize and esetPlot with the 'ggplot2' package*

---

**Description**

visualize and [esetPlot](#) with the 'ggplot2' package

**Usage**

`ggPlotEset(object)`

**Arguments**

`object`            object of class `esetPlot`

**Value**

ggplot object

**Author(s)**

Laure Cougnaud

---

`ggplotEsetPlot-class`    *a S4 class to represent ggplot plots*

---

**Description**

a S4 class to represent ggplot plots

**Value**

S4 object of class `ggplotEsetPlot`

**Slots**

`returnTopElements` logical, if TRUE (FALSE by default) return the outlying elements labelled in the plot (if any)

`title` string or expression with plot title, " by default

`xlab` string or expression with label for the x axis

`ylab` string or expression with label for the y axis

**Author(s)**

Laure Cougnaud

ggvisEsetPlot-class    *a S4 class for ggvis plot*

---

**Description**

a S4 class for ggvis plot

**Value**

S4 object of class ggvisEsetPlot

**Slots**

adjustLegend logical, if TRUE (by default) adjust the legends in ggvis to avoid overlapping legends when multiple legends

alphaRange transparency (alpha) range used in the plot, c(0.1, 1) by default.

**Author(s)**

Laure Cougnaud

---

ggvisPlotEset    *visualize and [esetPlot](#) with the the 'ggvis' package*

---

**Description**

visualize and [esetPlot](#) with the the 'ggvis' package

**Usage**

```
ggvisPlotEset(object)
```

**Arguments**

object            object of class [esetPlot](#)

**Value**

ggvis plot object

**Author(s)**

Laure Cougnaud



---

plotEset                      *plot an [plotEset](#) object*

---

### Description

plot an [plotEset](#) object

### Usage

```
plotEset(object, returnEsetPlot = FALSE)

## S4 method for signature 'ggplotEsetPlot'
plotEset(object, returnEsetPlot = FALSE)

## S4 method for signature 'ggvisEsetPlot'
plotEset(object, returnEsetPlot = FALSE)

## S4 method for signature 'plotlyEsetPlot'
plotEset(object, returnEsetPlot = FALSE)
```

### Arguments

object                      object of class [esetPlot](#)  
returnEsetPlot            logical, if TRUE return also the [esetPlot](#) object, such as can be re-use for future call to [plotEset](#)

### Value

the plot object if returnEsetPlot is FALSE, otherwise a list with 'plot': the plot object and 'eset-Plot': the [esetPlot](#) object

### Author(s)

Laure Cougnaud

---

plotlyEsetPlot-class    *a S4 class to represent plotly plots*

---

### Description

a S4 class to represent plotly plots

### Value

S4 object of class plotlyEsetPlot

**Slots**

returnTopElements logical, if TRUE (FALSE by default) return the outlying elements labelled in the plot (if any)

size specified size(s) (cex) for the points, replicated if needed, used only if sizeVar is empty, a factor or character by default: '20' if sizeVar is not specified

**Author(s)**

Laure Cougnaud

---

plotlyPlotEset      *visualize and [esetPlot](#) with the the 'plotly' package*

---

**Description**

visualize and [esetPlot](#) with the the 'plotly' package

**Usage**

```
plotlyPlotEset(object)
```

**Arguments**

object      object of class [esetPlot](#)

**Value**

plotly plot

**Author(s)**

Laure Cougnaud

---

plotTopElements      *plot top elements for a static plot*

---

**Description**

This create geom\_text object with top genes/sample/pathways

**Usage**

```
plotTopElements(
  packageTextLabel = c("ggrepel", "ggplot2"),
  cex = 1,
  just = c(0.5, 0.5),
  color = "black",
  returnTopElements = FALSE,
  ...
)
```

**Arguments**

packageTextLabel	package used to label the outlying genes/samples/gene sets, either 'ggrepel' (by default, only used if package ggrepel is available), or 'ggplot2'
cex	cex of text in the plot
just	justification of elements in the plot, only use if packageTextLabel is 'ggplot2'
color	color for the elements in the plot
returnTopElements	logical if TRUE (FALSE by default) return the outlying elements

**Value**

- if the elements are present in the data: if returnTopElements is:
  - TRUE: return a list with two arguments:
    - \* topElements: string with top elements labelled in the plot
    - \* geomText: output of geom\_text
  - FALSE: only return the output of geom\_text
- if not, return NULL

**Author(s)**

Laure Cougnaud

---

setFixElement	<i>check if the aesthetic is fixed (e.g. color, shape, size 'palette')</i>
---------------	--

---

**Description**

check if the aesthetic is fixed (e.g. color, shape, size 'palette')

**Usage**

```
setFixElement(typeVar, valVar)
```

**Arguments**

typeVar	name of variable for aesthetic
valVar	fixed value of variable of aesthetic

**Value**

logical, if TRUE the element is fixed

**Author(s)**

Laure Cougnaud

---

setManualScale	<i>check if manual aesthetic should be set</i>
----------------	--

---

**Description**

This is the case only if `typeVar` and `valVar` are specified, and if the variable is not numeric or integer (doesn't work with `ggplot2`)

**Usage**

```
setManualScale(x, typeVar, valVar)
```

**Arguments**

<code>x</code>	data.frame with <code>typeVar</code>
<code>typeVar</code>	name of variable for aesthetic
<code>valVar</code>	fixed value of variable of aesthetic

**Value**

logical, if TRUE the manual scale should be set

**Author(s)**

Laure Cougnaud

---

simpleCap	<i>capitalize the first letter of a word</i>
-----------	--

---

**Description**

capitalize the first letter of a word

**Usage**

```
simpleCap(x)
```

**Arguments**

<code>x</code>	string
----------------	--------

**Value**

string with first letter capitalized

---

varToFm	<i>Get formula for a specific variable, to be used in aesthetic specification in <a href="#">plot_ly</a>.</i>
---------	---

---

**Description**

Get formula for a specific variable, to be used in aesthetic specification in [plot\\_ly](#).

**Usage**

```
varToFm(var)
```

**Arguments**

var                    Character vector with variable to combine. Otherwise with the '+' operator.

**Value**

[as.formula](#)

**Author(s)**

Laure Cougnaud

# Index

## \* internal

- characterORexpressionORcall-class, [2](#)
- esetPlot-class, [7](#)
- esetPlotInteractive-class, [10](#)
- formatManualScale, [24](#)
- formatOutput, [25](#)
- getAxesLimits, [25](#)
- getCoordGeneSets, [26](#)
- getDataPlotSamplesWithAnnotation, [27](#)
- getMethodsInputObjectEsetVis, [29](#)
- getTopElements, [29](#)
- ggPlotEset, [30](#)
- ggvisEsetPlot-class, [32](#)
- ggvisPlotEset, [32](#)
- plotEset, [33](#)
- plotlyEsetPlot-class, [33](#)
- plotlyPlotEset, [34](#)
- plotTopElements, [34](#)
- setFixElement, [35](#)
- setManualScale, [36](#)
- simpleCap, [36](#)
- varToFm, [37](#)

as. formula, [37](#)

characterORexpressionORcall-class, [2](#)

eSet, [2](#), [10](#), [15](#), [21](#), [27](#)

esetLda, [2](#), [27](#)

esetPlot, [6–8](#), [14](#), [19](#), [23](#), [30–34](#)

esetPlot (esetPlot-class), [7](#)

esetPlot-class, [7](#)

esetPlotInteractive (esetPlotInteractive-class), [10](#)

esetPlotInteractive-class, [10](#)

esetPlotWrapper, [6](#), [10](#), [23](#), [27](#)

esetSpectralMap, [15](#), [27](#)

esetTsne, [21](#)

formatManualScale, [24](#)

formatOutput, [25](#)

getAxesLimits, [25](#)

getAxesLimits, esetPlot-method (getAxesLimits), [25](#)

getCoordGeneSets, [26](#)

getDataPlotSamplesWithAnnotation, [27](#)

getDataPlotSamplesWithAnnotation, esetPlot-method (getDataPlotSamplesWithAnnotation), [27](#)

getDataPlotSamplesWithAnnotation, ggvisEsetPlot-method (getDataPlotSamplesWithAnnotation), [27](#)

getDataPlotSamplesWithAnnotation, plotlyEsetPlot-method (getDataPlotSamplesWithAnnotation), [27](#)

getGeneSets, [27](#), [28](#)

getGeneSetsForPlot, [5](#), [9](#), [13](#), [18](#), [27](#)

getMethodsInputObjectEsetVis, [29](#)

getTopElements, [26](#), [29](#)

ggPlotEset, [30](#)

ggplotEsetPlot (ggplotEsetPlot-class), [31](#)

ggplotEsetPlot-class, [31](#)

ggplotly, [6](#), [14](#), [19](#), [23](#)

ggvisEsetPlot (ggvisEsetPlot-class), [32](#)

ggvisEsetPlot-class, [32](#)

ggvisPlotEset, [32](#)

initialize, esetPlot-method (esetPlot-class), [7](#)

lda, [7](#)

mpm, [15](#), [17](#), [20](#)

plot.mpm, [17](#)

plot\_ly, [37](#)

plotEset, [25](#), [27](#), [33](#), [33](#)

plotEset, ggplotEsetPlot-method (plotEset), [33](#)

plotEset, ggvisEsetPlot-method (plotEset), [33](#)

plotEset, plotlyEsetPlot-method (plotEset), [33](#)

plotlyEsetPlot (plotlyEsetPlot-class), [33](#)

plotlyEsetPlot-class, [33](#)  
plotlyPlotEset, [34](#)  
plotTopElements, [34](#)

Rtsne, [21](#), [22](#), [24](#)

setFixElement, [35](#)  
setManualScale, [36](#)  
simpleCap, [36](#)  
spectralMap, [20](#)

varToFm, [37](#)