Package 'RTNsurvival'

May 9, 2026

Type Package

Title Survival analysis using transcriptional networks inferred by the RTN package

Version 1.37.0

Author Clarice S. Groeneveld, Vinicius S. Chagas, Mauro A. A. Castro

Maintainer Clarice Groeneveld <clari.groeneveld@gmail.com>, Mauro A. A. Castro <mauro.a.castro@gmail.com>

Depends R(>= 4.4.0), RTN(>= 2.32), RTNduals(>= 1.32), methods

Imports survival, RColorBrewer, grDevices, graphics, stats, utils, scales, data.table, egg, ggplot2, pheatmap, dunn.test

Suggests knitr, rmarkdown, BiocStyle, RUnit, BiocGenerics

Description

RTNsurvival integrates regulons inferred by the RTN package with survival data. For each regulon, a two-tailed GSEA framework computes a differential Enrichment Score (dES) at the individual-sample level. The resulting dES distribution across samples is then used to evaluate survival associations within the cohort. Two primary workflows are supported: (i) Cox proportional hazards models, in which regulon activities are treated as predictors of survival time, and (ii) Kaplan–Meier analyses assessing cohort stratification based on regulon activity. All graphical outputs are customizable according to user specifications.

License Artistic-2.0

biocViews NetworkEnrichment, Survival, GeneRegulation, GeneSetEnrichment, NetworkInference, GraphAndNetwork

LazyData TRUE

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.2

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

git_branch devel

git_last_commit 0723cd1

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-08

Contents

RTNsurvival-package . . . 2
hclust_semisupervised . . . 3
tni2tnsPreprocess,TNI-method . . . 4
TNS-class . . . 6
TNS.data . . . 6
tnsAREA3,TNS-method . . . 7
tnsCox,TNS-method . . . 8
tnsCoxInteraction,TNS-method . . . 8
tnsGet,TNS-method . . . 10
tnsGSEA2,TNS-method . . . 11
tnsInteraction,TNS-method . . . 12
tnsKM,TNS-method . . . 12
tnsKmInteraction,TNS-method . . . 13
tnsPlotCovariates,TNS-method . . . 15
tnsPlotCox,TNS-method . . . 16
tnsPlotCoxInteraction,TNS-method . . . 18
tnsPlotGSEA2,TNS-method . . . 19
tnsPlotKM,TNS-method . . . 21
tnsPlotKmInteraction,TNS-method . . . 22
tnsPlotSRE,TNS-method . . . 23
tnsSRD,TNS-method . . . 25
tnsSRE,TNS-method . . . 26
tnsStratification . . . 27

Index **28**

`RTNsurvival-package` *Performs survival analysis using transcriptional networks inferred by the RTN package.*

Description

This package provides classes and methods to perform survival analysis using transcriptional networks inferred by the RTN package, including Kaplan-Meier and multivariate survival analysis using Cox's regression model.

Details

Package:	RTNsurvival
Type:	Package
License:	Artistic-2.0

Index

TNS-class:	an S4 class for survival survival analysis using RTN transcriptional networks.
tni2tnsPreprocess:	a preprocessing method for objects of class TNS.
tnsGSEA2:	compute regulon activity by calling 'GSEA2' algorithm.
tnsPlotGSEA2:	plot results from the two-tailed GSEA.
tnsKM:	Kaplan-Meier analysis for TNS class objects.
tnsPlotKM:	Kaplan-Meier plots for TNS class objects.
tnsCox:	Cox regression analysis for TNS class objects.
tnsPlotCox:	Cox plots for TNS class objects.
tnsGet:	Get information from slots in a TNS object.
tnsInteraction:	A generic call to 'tnsCoxInteraction' and 'tnsKmInteraction'.
tnsKmInteraction:	Kaplan-Meier analysis for dual regulons.
tnsPlotKmInteraction:	Plot results from Kaplan-Meier analysis for dual regulons.
tnsCoxInteraction:	Cox regression analysis for dual regulons.
tnsPlotCoxInteraction:	Plot results from Cox regression analysis for dual regulons.
tnsPlotGSEA2:	Plot 2-tailed GSEA for a sample from a TNS.
tnsAREA3:	compute regulon activity by calling 'aREA3' algorithm.

Further information is available in the vignettes by typing `vignette("RTNsurvival")`. Documented topics are also available in HTML by typing `help.start()` and selecting the RTNsurvival package from the menu.

Author(s)

Clarice S. Groeneveld, Vinicius S. Chagas, Gordon Robertson, ..., Kerstin Meyer, Mauro A. A. Castro

References

Fletcher M.N.C. et al., *Master regulators of FGFR2 signalling and breast cancer risk.* Nature Communications, 4:2464, 2013.

Castro M.A.A. et al., *Regulators of genetic risk of breast cancer identified by integrative network analysis.* Nature Genetics, 48:12-21, 2016.

`hclust_semisupervised` *Semi-supervised hierarchical clustering*

Description

This function will run a semi-supervised hierarchical clustering, using prior knowledge to initialize clusters, and then unsupervised clustering on the unlabeled data.

Usage

```
hclust_semisupervised(
  data,
  groups,
  dist_method = "euclidean",
  hclust_method = "complete"
)
```

Arguments

data	a data.frame to be clustered by rows
groups	a list of vectors. If we unlist(groups), all elements must be present in the rownames of data. Each vector in the list will be treated as a separate group for the hierarchical clustering, and rejoined in order at the end.
dist_method	a distance computation method. Must be one of "euclidean", "maximum", "manhattan", "canberra", "binary", "minkowski", "pearson", "spearman"
hclust_method	an agglomeration method. Should be a method supported by hclust, one of: "ward.D", "ward.D2", "single", "complete", "average" (= UPGMA), "mcquitty" (= WPGMA), "median" (= WPGMC) or "centroid" (= UPGMC).

Value

hclust_semisupervised returns a list. The first element of the list is the data, reordered so that the merged hclust object will work. The second element is the result of the semi-supervised hierarchical clustering.

Examples

```
#--- add  unique labels to 'iris' data
rownames(iris) <- paste(iris$Species, formatC(1:nrow(iris), width=3, flag="0"), sep="_")

#--- unsupervised clustering
hc <- hclust(dist(iris[, -5]))
plot(hc, hang=-1, cex=0.5)

#--- semi-supervised clustering
hc_ssuper <- hclust_semisupervised(data = iris[, -5],
                                   groups = split(rownames(iris), iris$Species))
plot(hc_ssuper, hang=-1, cex=0.5)
```

---

tni2tnsPreprocess,TNI-method

Preprocessing of TNS class objects

Description

Creates TNS class onbjects for regulons an survival data.

Usage

```
## S4 method for signature 'TNI'
tni2tnsPreprocess(
  tni,
  survivalData = NULL,
  regulatoryElements = NULL,
  time = 1,
  event = 2,
  endpoint = NULL,
```

```
  pAdjustMethod = "BH",
  keycovar = NULL,
  samples = NULL,
  excludeMid = FALSE,
  excludeAttribs = NULL
)
```

Arguments

tni	A TNI class, already processed with the same samples listed in the survival data.frame.
survivalData	A named data.frame with samples in rows and survival data in the columns (this does not need to be provided if avaibale in the 'TNI' object).
regulatoryElements	A character vector specifying which 'TNI' regulatory elements should be evaluated.
time	A numeric or character value corresponding to the column of the data.frame where the time of last observation is given.
event	A numeric or character value, corresponding to the columm of the data.frame where the 'event' information is given.
endpoint	A numeric value. It represents the cut-off point for the 'time', if any.
pAdjustMethod	A single character value specifying the p-value adjustment method to be used (see 'p.adjust' function for details).
keycovar	A character vector of 'keycovars' listed in 'survivalData' columns.
samples	An optional character vector listing samples to be analyzed.
excludeMid	A logical value. If TRUE, inconclusive dES values is not consired in the survival analysis.
excludeAttribs	A character vector of attributes listed in the column names of the survivalData, indicating sample groups to be excluded from the survival analysis. All attributes should be binary encoded. Available attributes can be checked by running colnames(tnsGet(tns, "survivalData"))

Value

A preprocessed TNS class

See Also

tni.preprocess for similar preprocessing.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

# create a new TNS object
stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
        keycovar = c('Grade','Age'), time = 1, event = 2)
```

TNS-class *TNS: An S4 class for survival analysis using transcriptional networks inferred by the RTN package.*

Description

TNS: An S4 class for survival analysis using transcriptional networks inferred by the RTN package.

Slots

`TNI` a previously computed TNI-class object.

`survivalData` a data frame containing the survival data for all samples. Samples must be in the rows and the survival variables in the columns. Time of last update and event in last update (0 for alive, 1 for deceased).

`para` a list with the parameters used to compute the GSEA2 analysis.

`results` a list with results from TNS methods.

`status` a vector containing the processing status of the TNS object.

Constructor

see `tni2tnsPreprocess` constructor.

TNS.data *A pre-processed dataset for demonstration purposes only.*

Description

A minimum dataset used to demonstrate RTNsurvival main features.

Usage

```
data(survival.data)
```

Format

A data.frame with a subset of samples in the Fletcher2013b package.

Details

The dataset consists of data.frame with survival and clinical variables used in the RTNsurvival vignettes. It should be regarded as a toy example for demonstration purposes only, despite being extracted, pre-processed and size-reduced from Fletcher et al. (2013) and Curtis et al. (2012).

Value

a data.frame.

References

Fletcher M.N.C. et al., *Master regulators of FGFR2 signalling and breast cancer risk.* Nature Communications, 4:2464, 2013.

Curtis C. et al., *The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups.* Nature 486, 7403. 2012.

Examples

```
data(survival.data)
```

`tnsAREA3,TNS-method` *Compute regulon activity by calling aREA (analytic Rank-based Enrichment Analysis) algorithm*

Description

Uses `tni.area3` function to compute regulon activity for TNS class objects.

Usage

```
## S4 method for signature 'TNS'
tnsAREA3(tns, ...)
```

Arguments

`tns`	A TNS class, which has been preprocessed
`...`	Additional parameters passed to `tni.area3` function.

Value

A TNS class, with added regulon activity scores.

References

Alvarez et al. Functional characterization of somatic mutations in cancer using network-based inference of protein activity. Nature Genetics, 48(8):838-847, 2016.

See Also

`tni.area3` for additional details.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Grade','Age'), time = 1, event = 2)

stns <- tnsAREA3(stns)
```

tnsCox,TNS-method *Cox regression analysis for TNS class objects*

Description

Run Cox multivariate regression for regulons and other covariates.

Usage

```
## S4 method for signature 'TNS'
tnsCox(tns, regs = NULL, qqkeycovar = FALSE, verbose = TRUE)
```

Arguments

tns	A TNS object, which must have passed GSEA2 analysis.
regs	An optional string vector listing regulons to be tested.
qqkeycovar	A logical value. If TRUE, only the samples in the 2nd and 3rd quartils of 'keycovar' are used in the analysis. If FALSE, all samples are used (see tni2tnsPreprocess).
verbose	A logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).

Value

Cox hazard models and statistics.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Age','Grade'), time = 1, event = 2)
stns <- tnsGSEA2(stns)
stns <- tnsCox(stns, regs = c('PTTG1','E2F2','FOXM1'))
tnsGet(stns, "coxTable")
```

tnsCoxInteraction,TNS-method *Cox regression analysis for dual regulons*

Description

Cox regression analysis for dual regulons, including the interaction term.

Usage

```
## S4 method for signature 'TNS'
tnsCoxInteraction(
  tns,
  regs1 = NULL,
  regs2 = NULL,
  stepFilter = TRUE,
  pValueCutoff = 0.05,
  phiThreshold = 0.5,
  method = c("multipl", "additive"),
  verbose = TRUE
)
```

Arguments

tns	A TNS object with regulons used to compute the dual regulons.
regs1	An optional character vector specifying regulons to test for interaction. If missing, it will be used all regulons listed in the 'TNS' object.
regs2	An optional character vector specifying regulons to test for interaction. If missing, it will be used all regulons listed in the 'TNS' object.
stepFilter	A single logical value specifying to use a step-filter algorithm, testing dual regulons that have at least one significant predictor in the 'tnsCox' method (when stepFilter=TRUE) or not (when stepFilter=FALSE).
pValueCutoff	A numeric value in [0,1]. The p-value cutoff applied to the results from the previous steps of the analysis pipeline (when stepFilter=TRUE).
phiThreshold	A numeric value in [0,1]. The 'phi' coefficient is a measure of association between two dichotomous variables. Here, a threshold on 'phi' is used to remove regulon pairs with 'strong' dependencies.
method	Assess either additive or multiplicative interactions.
verbose	A logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).

Value

Cox hazard models and statistics.

An updated TNS-class object containing Cox regression models for all given duals

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

# perform survival analysis for regulons
stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
time = 1, event = 2)
stns <- tnsGSEA2(stns)

# run Cox regression for dual regulons
```

```
stns <- tnsCoxInteraction(stns, stepFilter = FALSE)
tnsGet(stns, "coxInteractionTable")
```

`tnsGet,TNS-method` *Get information from slots in a TNS object*

Description

Get information from individual slots in a TNS object and any available results from a previous analysis.

Usage

```
## S4 method for signature 'TNS'
tnsGet(tns, what)
```

Arguments

`tns`	A TNS object.
`what`	A string specifying what should be retrieved from the object. Options: 'status','survivalData', 'regulonActivity', 'TNI', 'para', 'kmTable', 'kmFit', 'coxTable', 'coxFit', 'kmInteractionTable', 'kmInteractionFit', 'coxInteractionTable', 'coxInteractionFit', and 'regulatoryElements'.

Value

Content from slots in the TNS object.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)
regulonActivity <- tnsGet(stns, 'regulonActivity')
```

tnsGSEA2,TNS-method *Compute regulon activity using 2-tailed Gene Set Enrichment Analysis*

Description

Works as a wrapper for tni.gsea2, performing a 2-tailed GSEA analysis on a TNI class object and integrating the results into the TNS class object.

Usage

```
## S4 method for signature 'TNS'
tnsGSEA2(tns, ...)
```

Arguments

tns	A TNS class, which has been preprocessed
...	Additional parameters passed to tni.gsea2 function.

Value

A TNS class, with added regulon activity scores.

See Also

tni.gsea2 for information on all parameters.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)

## Not run:

# parallel version with SNOW package!
library(snow)
options(cluster=snow::makeCluster(3, "SOCK"))
stns <- tnsGSEA2(stns)
stopCluster(getOption("cluster"))


## End(Not run)
```

tnsInteraction,TNS-method

Survival analysis for dual regulons

Description

A generic call to 'tnsCoxInteraction' and 'tnsKmInteraction' functions.

Usage

```
## S4 method for signature 'TNS'
tnsInteraction(tns, ..., verbose = TRUE)
```

Arguments

tns	A TNS object, which must have passed GSEA2 analysis.
...	Parameters passed to tnsKmInteraction and tnsCoxInteraction functions.
verbose	A logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).

Value

A TNS object evaluated by the 'tnsKmInteraction' and 'tnsCoxInteraction' functions.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)

# survival analysis for dual regulons
# stns <- tnsInteraction(stns, stepFilter = FALSE)
```

tnsKM,TNS-method

Kaplan-Meier analysis for TNS class objects

Description

Creates survival curves and tests if there is a difference between curves using 'survfit' and 'survdiff' functions, respectivelly.

Usage

```
## S4 method for signature 'TNS'
tnsKM(
  tns,
  regs = NULL,
  sections = 1,
  undetermined.status = TRUE,
  verbose = TRUE
)
```

Arguments

tns	A TNS object, which must have passed GSEA2 analysis.
regs	An optional string vector listing regulons to be tested.
sections	A numeric value for sample stratification. The larger the number, the more subdivisions will be created for the Kaplan-Meier analysis.
undetermined.status	a logical value. If TRUE, regulons assigned as 'undetermined' will form a group.
verbose	A logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).

Value

Results from 'survfit' and 'survdiff', including log-rank statistics.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
        keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)
stns <- tnsKM(stns)
tnsGet(stns, "kmTable")
```

---

tnsKmInteraction,TNS-method

Kaplan-Meier analysis for dual regulons

Description

Kaplan-Meier analysis for dual regulons, assessing the interaction between regulons.

Usage

```
## S4 method for signature 'TNS'
tnsKmInteraction(
  tns,
  regs1 = NULL,
  regs2 = NULL,
  stepFilter = TRUE,
  pValueCutoff = 0.05,
  phiThreshold = 0.5,
  verbose = TRUE
)
```

Arguments

tns	A TNS object, which must have passed GSEA2 analysis.
regs1	An optional character vector specifying regulons to test for interaction. If missing, it will be used all regulons listed in the 'TNS' object.
regs2	An optional character vector specifying regulons to test for interaction. If missing, it will be used all regulons listed in the 'TNS' object.
stepFilter	A single logical value specifying to use a step-filter algorithm, testing dual regulons that have at least one significant predictor in the 'tnsKM' method (when stepFilter=TRUE) or not (when stepFilter=FALSE).
pValueCutoff	A numeric value in [0,1]. The p-value cutoff applied to the results from the previous steps of the analysis pipeline (when stepFilter=TRUE).
phiThreshold	A numeric value in [0,1]. The 'phi' coefficient is a measure of association between two dichotomous variables. Here, a threshold on 'phi' is used to remove regulon pairs with 'strong' dependencies.
verbose	A logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).

Value

Results from 'survfit' and 'survdiff', including log-rank statistics.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)

# KM analysis for dual regulons
# stns <- tnsKmInteraction(stns, stepFilter = FALSE)
# tnsGet(stns, "kmInteractionTable")
```

`tnsPlotCovariates,TNS-method`

Plot regulon activity and categorical covariates

Description

This method plots regulon activity for a given regulon in all samples and adds covariate tracks to evaluate the regulon activity distribution. The samples are order by regulon activity for that particular regulon.

Usage

```
## S4 method for signature 'TNS'
tnsPlotCovariates(
  tns,
  regs = NULL,
  attribs = NULL,
  fname = "covarplot",
  fpath = ".",
  ylab = "Regulon activity (dES)",
  xlab = "Samples",
  plotpdf = FALSE,
  plotbatch = FALSE,
  panelHeights = c(1, 1),
  width = 5.3,
  height = 4,
  dummyEncode = TRUE,
  divs = NULL
)
```

Arguments

`tns`	A A TNS object.
`regs`	An optional string vector specifying regulons to plot.
`attribs`	A character vector of attributes listed in the column names of the survivalData. All attributes should be either binary encoded or categorical variables for plotting. Available attributes can be checked by running colnames(tnsGet(tns, "survivalData")). Alternatively, attributes can be grouped when provided within a list.
`fname`	A string. The name of the file in which the plot will be saved
`fpath`	A string. The path to the directory where the plot will be saved
`ylab`	A string. The label of the y-axis, describing what is represented.
`xlab`	A string. The label of the x-axis.
`plotpdf`	A logical value. If TRUE, a pdf file is created instead of plotting to the graphics device.
`plotbatch`	A logical value. If TRUE, plots for all regulons are saved in the same file. If FALSE, each plot for each regulon is saved in a different file.

panelHeights	A numeric vector of length 2 specifying the relative heights of the panels (regulon activity plot, and covariate tracks)
width	A numeric value. Represents the width of the plot.
height	A numeric value. Represents the height of the plot.
dummyEncode	A logical value. If TRUE, all categorical variables are dummy encoded. If FALSE, categorical variables are represented as one track and a legend is added to the plot.
divs	A numeric vector of division positions in the covariate tracks.

Details

Automatic dummy encoding is available for categorical variables.

Value

A plot of regulon activity and covariate tracks.

Examples

```
# load survival data
data(survival.data)
# load TNI-object
data(stni, package = "RTN")

# create TNS object
stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)

# plot only binary covariates
tnsPlotCovariates(stns, "MYB",
attribs = c("ER+", "ER-", "PR+", "PR-", "LumA", "LumB", "Basal",
"Her2", "Normal"), divs = c(2, 4))

# also dummy encode categorical variables (LN and Grade)
tnsPlotCovariates(stns, "MYB",
attribs = c("ER+", "ER-", "PR+", "PR-", "LumA", "LumB", "Basal",
"Her2", "Normal", "LN", "Grade"), divs = c(2, 4, 9, 12))

# don't dummy encode categorical variables
tnsPlotCovariates(stns, "MYB", attribs = c("ER+", "ER-", "PR+", "PR-",
"LumA", "LumB", "Basal", "Her2", "Normal", "Grade"), divs = c(2, 4, 9),
dummyEncode = FALSE)
```

---

tnsPlotCox,TNS-method *Cox plots for TNS class objects*

Description

Plot results from the 'tnsCox' function.

Usage

```
## S4 method for signature 'TNS'
tnsPlotCox(
  tns,
  regs = NULL,
  pValueCutoff = 1,
  fname = "coxplot",
  fpath = ".",
  ylab = "Regulons and other covariates",
  xlab = "Hazard Ratio (95% CI)",
  width = 5,
  height = 5,
  xlim = c(0.3, 3),
  sortregs = TRUE,
  plotpdf = FALSE
)
```

Arguments

tns	A TNS object, which must have passed GSEA2 analysis.
regs	An optional string vector specifying regulons to make the plot.
pValueCutoff	A numeric value in [0,1]. The p-value cutoff applied to the results from the Cox analysis pipeline.
fname	A string. The name of the PDF file which will contain the plot.
fpath	A string. The directory where the file will be saved.
ylab	A string. The label of the y-axis, describing what is represented.
xlab	A string. The label of the x-axis.
width	A numeric value. The width of the plot.
height	A numeric value. The height of the plot.
xlim	A vector with 2 values indicating lowest and highest HR values.
sortregs	A logical value. If TRUE, regulons are sorted from most negatively associated-with hazard to most positively associated with hazard.
plotpdf	A logical value.

Value

A Cox hazard model plot and statistics.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Age','Grade'), time = 1, event = 2)
stns <- tnsGSEA2(stns)
stns <- tnsCox(stns, regs = c('PTTG1','E2F2','FOXM1'))
```

```
tnsPlotCox(stns)
```

`tnsPlotCoxInteraction,TNS-method`
Plot results from Cox regression analysis for dual regulons

Description

Plot results from Cox regression analysis for dual regulons

Usage

```
## S4 method for signature 'TNS'
tnsPlotCoxInteraction(
  tns,
  dualreg,
  xlim = NULL,
  ylim = NULL,
  zlim = NULL,
  zcols = c("#008080ff", "#d45500ff"),
  colorPalette = "bluered",
  fname = "coxInteraction",
  fpath = ".",
  width = 4.5,
  height = 4,
  plotype = "p1",
  plotpdf = FALSE
)
```

Arguments

`tns`	A 'TNS' object.
`dualreg`	A character string with the name of a dual regulon.
`xlim`	A numeric vector of length 2, i.e. xlim = c(x1, x2), indicating regulon activity limits of the plot for the first member of the dual regulon. If xlim = NULL, it will be derived from the observed data ranges. Values must be in the range [-2,2].
`ylim`	A numeric vector of length 2, i.e. ylim = c(y1, y2), indicating regulon activity limits of the plot for the second member of the dual regulon. If ylim = NULL, it will be derived from the observed data ranges. Values must be in the range [-2,2].
`zlim`	A numeric vector of length 2, i.e. zlim = c(z1, z2), indicating the limits of the plot for the Hazard Ratio (HR) values. If zlim = NULL, it will be derived from the observed data ranges.
`zcols`	A vector of length 2 indicating a diverging color scheme for the Hazard Ratio (HR).
`colorPalette`	A string, which can be 'red', 'blue', 'redblue', or 'bluered'. Alternatively, it can be a vector of five colors or hex values.

fname	A string. The name of the PDF file (when plotpdf=TRUE).
fpath	A string. The directory where the file will be saved.
width	A numeric value. The width of the plot.
height	A numeric value. The height of the plot.
plotype	A string indicating a plot type (see mbrPlotInteraction).
plotpdf	A logical value.

Value

A Cox hazard model plot and statistics.

A 3D heatmap plot.

See Also

tnsKM, tnsCox

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

# perform survival analysis for regulons
stns <- tni2tnsPreprocess(stni, survivalData = survival.data, time = 1, event = 2)
stns <- tnsGSEA2(stns, verbose=FALSE)

# run Cox regression for dual regulons
# stns <- tnsCoxInteraction(stns, stepFilter = FALSE)
# tnsPlotCoxInteraction(stns, dualreg = "FOXM1~PTTG1")
```

---

tnsPlotGSEA2,TNS-method

Plot 2-tailed GSEA for a sample from a TNS

Description

Makes a 2-tailed GSEA plot for a certain phenotype (sample) present in a TNS. A wrapper of tna.plot.gsea2

Usage

```
## S4 method for signature 'TNS'
tnsPlotGSEA2(
  tns,
  aSample,
  regs = NULL,
  refsamp = NULL,
  checklog = FALSE,
  ntop = NULL,
```

```
  pValueCutoff = 0.05,
  pAdjustMethod = "BH",
  verbose = TRUE,
  plotpdf = FALSE,
  ...
)
```

Arguments

tns	A TNS object
aSample	A string specifying a given sample number present in the 'survivalData' table.
regs	An optional string vector specifying regulons to make the plot.
refsamp	A character vector.
checklog	A logical value. If TRUE, expression values are transformed into log space.
ntop	An optional integer value. The number of regulons for which the GSEA2 will be plotted.
pValueCutoff	A numeric value in [0,1]. The p-value cutoff for the analysis.
pAdjustMethod	A character. Specifies the adjustment method for the pvalue. See p.adjust
verbose	A logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).
plotpdf	A single logical value.
...	parameters which will be passed to tna.plot.gsea2, such as ylimPanels, heightPanels, width, height, ylabPanels, xlab...

Value

A plot containing the 2-tailed GSEA analysis for a phenotype.

See Also

tna.plot.gsea2 for all plot parameters

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
        keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns, verbose=FALSE)
tnsPlotGSEA2(stns, 'MB-5115', regs = 'FOXM1', plotpdf = FALSE)
```

`tnsPlotKM,TNS-method` *Kaplan-Meier plots for TNS class objects*

Description

Plot results from the 'tnsKM' function. The 'tnsPlotKM' function makes a 2 or 3 panel plot for survival analysis. The first panel shows the differential Enrichment score (dES) for all samples, ranked by dES in their sections. The second (optional) panel shows the status of other attributes which may be present in the survival data frame for all samples. The third panel shows a Kaplan-Meier plot computed for the given survival data, with a curve for each section.

Usage

```
## S4 method for signature 'TNS'
tnsPlotKM(
  tns,
  regs = NULL,
  attribs = NULL,
  pValueCutoff = 1,
  fname = "survplot",
  fpath = ".",
  xlab = "Months",
  ylab = "Survival probability",
  colorPalette = "bluered",
  plotpdf = FALSE,
  plotbatch = FALSE,
  width = 6.3,
  height = 3.6,
  panelWidths = c(3, 2, 2, 4)
)
```

Arguments

`tns`	A TNS object, which must have passed GSEA2 analysis.
`regs`	An optional string vector specifying regulons to make the plot.
`attribs`	A character vector of attributes listed in the column names of the survivalData. All attributes should be binary encoded for plotting. Available attributes can be checked by running colnames(tnsGet(tns, "survivalData")). Alternatively, attributes can be grouped when provided within a list.
`pValueCutoff`	A numeric value in [0,1]. The p-value cutoff applied to the results from the KM analysis pipeline.
`fname`	A string. The name of the file in which the plot will be saved
`fpath`	A string. The path to the directory where the plot will be saved
`xlab`	A string. The label for the x axis on the third panel. This should be the measure of time shown in the survival data frame after the last check-up.
`ylab`	A string. The label for the y axis on the third panel
`colorPalette`	A string, which can be 'redblue' or 'bluered'. Alternatively, it can be colors or hex values.

plotpdf	A logical value. If TRUE, the plot is saved as a pdf file. If false, it is plotted in the plotting area.
plotbatch	A logical value. If TRUE, plots for all regulons are saved in the same file. If FALSE, each plot for each regulon is saved in a different file.
width	A numeric value. Represents the width of the plot.
height	A numeric value. Represents the height of the plot.
panelWidths	A numeric vector of length=3 specifying the relative width of the internal panels.

Value

A plot, showing a graphical analysis for the 'tnsKM' function.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
        keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)
stns <- tnsKM(stns)
tnsPlotKM(stns)
```

---

tnsPlotKmInteraction,TNS-method

Plot results from Kaplan-Meier analysis for dual regulons

Description

Plot results from Kaplan-Meier analysis for dual regulons

Usage

```
## S4 method for signature 'TNS'
tnsPlotKmInteraction(
  tns,
  dualreg,
  fname = "kmInteraction",
  fpath = ".",
  xlab = "Months",
  ylab = "Survival probability",
  colorPalette = "bluered",
  width = 4,
  height = 4,
  plotpdf = FALSE
)
```

Arguments

tns	A TNS object, which must have passed GSEA2 analysis.
dualreg	A character string with the name of a dual regulon.
fname	A string. The name of the file in which the plot will be saved
fpath	A string. The path to the directory where the plot will be saved
xlab	A string. The label for the x axis on the third panel. This should be the measure of time shown in the survival data.frame after the last check-up.
ylab	A string. The label for the y axis on the third panel
colorPalette	A string, which can be 'redblue' or 'bluered'. Alternatively, it can be a vector of five colors or hex values.
width	A numeric value. Represents the width of the plot.
height	A numeric value. Represents the height of the plot.
plotpdf	A logical value. If TRUE, the plot is saved as a pdf file. If false, it is plotted in the plotting area.

Value

A plot, showing a graphical analysis for the 'tnsKmInteraction' function.

Examples

```
# load survival data
data(survival.data, package = "RTNsurvival")

# load TNI-object
data(stni, package = "RTN")

stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)

# KM analysis for dual regulons
# stns <- tnsKmInteraction(stns, stepFilter=FALSE)
# tnsPlotKmInteraction(stns, dualreg = "FOXM1~PTTG1")
```

---

tnsPlotSRE,TNS-method *Plot Subgroup Regulon Enrichment for TNS-class objects*

Description

This method plots the results of the subgroup regulon enrichment analysis in a heatmap. The rows of the heatmap represent enriched regulons, while the columns show the subgroups. The plotted values correspond to average regulon activity for a regulon in a subgroup. Enriched values can be marked.

Usage

```
## S4 method for signature 'TNS'
tnsPlotSRE(
  tns,
  subgroup = NULL,
  by = "nGroups",
  nGroupsEnriched = 1,
  nTopEnriched = 10,
  breaks = seq(-1.5, 1.5, 0.1),
  markEnriched = FALSE,
  ...
)
```

Arguments

tns	A A TNS object.
subgroup	a character vector. It must be the name of a column in the survivalData featuring the grouping information as a categorical variable.
by	one of 'nGroups' or 'groupTop'. If by = 'nGroups', the nGroupsEnriched value will be used to select regulons. If by = 'groupTop', 'nTopEnriched' will be used to select regulons for plotting.
nGroupsEnriched	a single integer. It represents in how many subgroups a regulon has to be enriched for it to appear in the rows of the heatmap.
nTopEnriched	a single integer. If by = 'groupTop', this represents how regulons will be shown for each group (duplicates are removed. The top regulons are chosen by significance.
breaks	a numerical vector of breaks for the heatmap.
markEnriched	a single logical value. If TRUE, asterisks are added to cells of heatmap that were found to be significant by tnsSRE.
...	parameters passed to pheatmap::pheatmap for customization.

Value

A heatmap of the subgroup regulon enrichment results.

Examples

```
# load survival data
data(survival.data)
# load TNI-object
data(stni, package = "RTN")

# create TNS object
stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
                          keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)

# run subgroup regulon enrichment analysis
stns <- tnsSRE(stns, "ER+")
tnsPlotSRE(stns)
```

tnsSRD,TNS-method *Subgroup Regulon Difference for TNS-class objects*

Description

This regulon evaluates differences between regulon activity of subgroups of samples, given a grouping variable. It performs Wilcoxon-Mann-Whitney (2 subgroups) or Kruskal-Wallis (3+ subgroups) Rank Sum Tests to check whether the activity scores of a given regulon are different between subgroups of samples.

Usage

```
## S4 method for signature 'TNS'
tnsSRD(
  tns,
  subgroup,
  pValueCutoff = 0.05,
  pAdjustMethod = "BH",
  regs = NULL,
  verbose = TRUE
)
```

Arguments

tns	A A TNS object.
subgroup	a character vector. It must be the name of a column in the survivalData featuring the grouping information as a categorical variable.
pValueCutoff	a single numeric value specifying the cutoff for p-values considered significant.
pAdjustMethod	a single character value specifying the p-value adjustment method to be used (see 'p.adjust' for details).
regs	An optional string vector specifying regulons to use for the analysis.
verbose	a logical value specifying whether to display messages and progress bar.

Value

A TNS-class object with the results of the subgroup regulon difference added to the results slot. To recover the results, use tnsGet(tns, "regulonDifference")

Examples

```
# load survival data
data(survival.data)
# load TNI-object
data(stni, package = "RTN")

# create TNS object
stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
                          keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)

# run subgroup regulon enrichment analysis
```

```
stns <- tnsSRD(stns, "ER+")
```

`tnsSRE,TNS-method` *Subgroup Regulon Enrichment for TNS-class objects*

Description

This method evaluates which regulons are enriched in sample groups, given a grouping variable. It performs Fisher's Exact Test whether a regulon is positively or negatively enriched in a subgroup using regulon activity.

Usage

```
## S4 method for signature 'TNS'
tnsSRE(tns, subgroup, regs = NULL, pValueCutoff = 0.05, pAdjustMethod = "BH")
```

Arguments

`tns`	A A TNS object.
`subgroup`	a character vector. It must be the name of a column in the survivalData featuring the grouping information as a categorical variable.
`regs`	An optional string vector specifying regulons to use for the analysis.
`pValueCutoff`	a single numeric value specifying the cutoff for p-values considered significant.
`pAdjustMethod`	a single character value specifying the p-value adjustment method to be used (see 'p.adjust' for details).

Value

A TNS-class object with the results of the subgroup regulon enrichment added to the results slot. To recover the results, use tnsGet(tns, "regulonEnrichment")

Examples

```
# load survival data
data(survival.data)
# load TNI-object
data(stni, package = "RTN")

# create TNS object
stns <- tni2tnsPreprocess(stni, survivalData = survival.data,
                          keycovar = c('Grade','Age'), time = 1, event = 2)
stns <- tnsGSEA2(stns)

# run subgroup regulon enrichment analysis
stns <- tnsSRE(stns, "ER+")

# plot the result
tnsPlotSRE(stns)
```

`tnsStratification` *Sample stratification for a TNS object*

Description

Internal function, used for sample stratification.

Usage

```
tnsStratification(
  tns,
  sections = 1,
  center = FALSE,
  undetermined.status = TRUE
)
```

Arguments

`tns`	a TNS object, which must have passed GSEA2 analysis.
`sections`	A numeric value for the stratification of the sample. The larger the number, the more subdivisions will be created for the Kaplan-Meier analysis.
`center`	a logical value. If TRUE, numbers assigned to each group is centralized on regulon activity scale.
`undetermined.status`	a logical value. If TRUE, regulons assigned as 'undetermined' will form a group.

Value

An updated TNS object.

See Also

`tnsKM`

Examples

```
# see tnsKM method.
```

Index

∗ **dataset**
TNS.data, 6
∗ **package**
RTNsurvival-package, 2

hclust_semisupervised, 3

mbrPlotInteraction, *19*

p.adjust, *20*

RTNsurvival *(*RTNsurvival-package*)*, 2
RTNsurvival-package, 2

survival.data *(*TNS.data*)*, 6

tna.plot.gsea2, *19, 20*
TNI, *5, 6*, *11*
tni.area3, *7*
tni.gsea2, *11*
tni.preprocess, *5*
tni2tnsPreprocess, *3*, *6*, *8*
tni2tnsPreprocess *(*tni2tnsPreprocess,TNI-method*)*, 4
tni2tnsPreprocess,TNI-method, 4
TNS, *5*, *7–15*, *17*, *20, 21*, *23–27*
TNS *(*TNS-class*)*, 6
TNS-class, *3*, 6
TNS.data, 6
tnsAREA3, *3*
tnsAREA3 *(*tnsAREA3,TNS-method*)*, 7
tnsAREA3,TNS-method, 7
tnsCox, *3*, *19*
tnsCox *(*tnsCox,TNS-method*)*, 8
tnsCox,TNS-method, 8
tnsCoxInteraction, *3*, *12*
tnsCoxInteraction *(*tnsCoxInteraction,TNS-method*)*, 8
tnsCoxInteraction,TNS-method, 8
tnsGet, *3*
tnsGet *(*tnsGet,TNS-method*)*, 10
tnsGet,TNS-method, 10
tnsGSEA2, *3*
tnsGSEA2 *(*tnsGSEA2,TNS-method*)*, 11
tnsGSEA2,TNS-method, 11
tnsInteraction, *3*
tnsInteraction *(*tnsInteraction,TNS-method*)*, 12
tnsInteraction,TNS-method, 12
tnsKM, *3*, *19*, *27*
tnsKM *(*tnsKM,TNS-method*)*, 12
tnsKM,TNS-method, 12
tnsKmInteraction, *3*, *12*
tnsKmInteraction *(*tnsKmInteraction,TNS-method*)*, 13
tnsKmInteraction,TNS-method, 13
tnsPlotCovariates *(*tnsPlotCovariates,TNS-method*)*, 15
tnsPlotCovariates,TNS-method, 15
tnsPlotCox, *3*
tnsPlotCox *(*tnsPlotCox,TNS-method*)*, 16
tnsPlotCox,TNS-method, 16
tnsPlotCoxInteraction, *3*
tnsPlotCoxInteraction *(*tnsPlotCoxInteraction,TNS-method*)*, 18
tnsPlotCoxInteraction,TNS-method, 18
tnsPlotGSEA2, *3*
tnsPlotGSEA2 *(*tnsPlotGSEA2,TNS-method*)*, 19
tnsPlotGSEA2,TNS-method, 19
tnsPlotKM, *3*
tnsPlotKM *(*tnsPlotKM,TNS-method*)*, 21
tnsPlotKM,TNS-method, 21
tnsPlotKmInteraction, *3*
tnsPlotKmInteraction *(*tnsPlotKmInteraction,TNS-method*)*, 22
tnsPlotKmInteraction,TNS-method, 22
tnsPlotSRE *(*tnsPlotSRE,TNS-method*)*, 23
tnsPlotSRE,TNS-method, 23
tnsSRD *(*tnsSRD,TNS-method*)*, 25
tnsSRD,TNS-method, 25
tnsSRE *(*tnsSRE,TNS-method*)*, 26

tnsSRE,TNS-method, 26
tnsStratification, 27