Package 'mt.surv'

May 9, 2026

Type Package

Title Multi-Threshold Survival Analysis

Version 1.1.1

Description Implements survival analyses across multiple abundance thresholds, repeatedly partitioning samples into groups and evaluating survival differences to assess taxonomic associations with outcomes.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Suggests testthat, rmarkdown, survminer

Imports dplyr, survival, broom, forcats, purrr, tidyr, ggplot2, magrittr

NeedsCompilation no

Author Rebecca Hoyd [aut] (ORCID: <https://orcid.org/0000-0003-1210-4491>), Alexander Loncar [aut, cre] (ORCID: <https://orcid.org/0009-0000-8079-2092>), YunZhou Liu [aut] (ORCID: <https://orcid.org/0000-0002-9892-2438>), Caroline Wheeler [aut], Daniel Spakowicz [aut] (ORCID: <https://orcid.org/0000-0003-2314-6435>)

Maintainer Alexander Loncar <alexander.loncar@osumc.edu>

Repository CRAN

Date/Publication 2026-04-21 18:42:13 UTC

Contents

`butterfly_input` *Generate input for butterfly plot*

Description

Generate input for butterfly plot

Usage

```
butterfly_input(area.input, subtype, num_micro)
```

Arguments

`area.input`	a dataframe with area for all taxonomy level generated by 'calculateArea'
`subtype`	a string object specifying what cancer subtype to be included
`num_micro`	an integer indicating how many microbes shoul be included for zoomed up version

Value

a list with two ggplot2 objects. One for all taxonomy level and another for selected number speficied in 'num_micro'

Examples

```
# Create mock area input data
area.input <- data.frame(
  type    = c("T_DDLPS", "T_DDLPS", "T_DDLPS",
              "T_LMS",   "T_LMS",
              "O_DDLPS", "O_DDLPS", "O_DDLPS",
              "O_LMS",   "O_LMS"),
  species = c("Proteobacteria", "Firmicutes", "Bacteroidetes",
              "Proteobacteria", "Firmicutes",
              "Proteobacteria", "Firmicutes", "Bacteroidetes",
              "Proteobacteria", "Firmicutes"),
  area    = c(0.45, 0.30, 0.25,
              0.60, 0.40,
              0.35, 0.40, 0.25,
              0.50, 0.50)
)

# Run for DDLPS subtype, showing top 3 microbes
result <- butterfly_input(area.input, subtype = "DDLPS", num_micro = 3)

# Result is a list with two ggplot objects
# result$common.taxarea.all   - stacked bar chart of all shared taxa
# result$common.taxarea.head  - dodged bar chart of top num_micro taxa
```

`calculateArea` *Quantify taxonomy significance using multi-threshold results*

Description

This function calculates the distance between alpha (default 0.05) and p-values below the significance threshold as an indicator of area under the curve.

Usage

```
calculateArea(taxlev, threshold.mat, alpha = 0.05)
```

Arguments

`taxlev`	a character object specifying which subject survival analysis is applying to.
`threshold.mat`	output from function survivalByQuantile. It includes information about survival under various threshold
`alpha`	a numeric value representing the statistically significant p-value

Value

A data frame containing the calculated area values for the specified taxonomy level.

Examples

```
# Create mock threshold matrix for two taxonomy levels
threshold.mat <- list(
  Phylum = data.frame(
    pval             = c(0.01, 0.04, 0.10, 0.02, 0.50),
    hazard.direction = c("<1", "<1", ">1", ">1", "<1")
  ),
  Family = data.frame(
    pval             = c(0.03, 0.20, 0.001, 0.04, 0.08),
    hazard.direction = c(">1", "<1",  "<1",  ">1", ">1")
  )
)

# Calculate area for Phylum level at default alpha = 0.05
result <- calculateArea(taxlev = "Phylum", threshold.mat = threshold.mat)

# Calculate area for Family level with stricter alpha
result_strict <- calculateArea(
  taxlev    = "Family",
  threshold.mat = threshold.mat,
  alpha     = 0.01
)
```

`StackBarinput` *Generate input objects for stacked bar plots*

Description

Generate input objects for stacked bar plots

Usage

```
StackBarinput(TCGA.list, ORIEN.list)
```

Arguments

`TCGA.list`	a list object containing survival and taxonomy information for TCGA
`ORIEN.list`	a list object containing survival and taxonomy information for ORIEN

Value

a dataframe that can be used to make stacked bar plot

Examples

```
# Create mock TCGA taxonomic data
tcga_tax <- data.frame(
  ID = c("S1", "S2", "S3"),
  p__Proteobacteria = c(0.4, 0.3, 0.5),
  p__Firmicutes     = c(0.2, 0.4, 0.1),
  p__Bacteroidetes  = c(0.2, 0.1, 0.2),
  p__Actinobacteria = c(0.1, 0.1, 0.1),
  p__Chordata       = c(0.1, 0.1, 0.1)
)

# Create mock TCGA clinical data
tcga_clin <- data.frame(
  ID          = c("S1", "S2", "S3"),
  diagnosis   = c("Dedifferentiated liposarcoma",
                  "Leiomyosarcoma, NOS",
                  "Other sarcoma"),
  vitalstatus = c(1, 0, 1),
  days        = c(365, 200, 500)
)

# Create mock ORIEN taxonomic data
orien_tax <- data.frame(
  ID = c("P1", "P2", "P3"),
  p__Proteobacteria = c(0.5, 0.2, 0.3),
  p__Firmicutes     = c(0.1, 0.5, 0.2),
  p__Bacteroidetes  = c(0.2, 0.2, 0.3),
  p__Actinobacteria = c(0.1, 0.0, 0.1),
```

```
  p__Chordata       = c(0.1, 0.1, 0.1)
)

# Create mock ORIEN clinical data
orien_clin <- data.frame(
  ID                = c("P1", "P2", "P3"),
  diagnosis         = c("Dedifferentiated liposarcoma",
                        "Leiomyosarcoma, NOS",
                        "Other sarcoma"),
  AgeCollect        = c(55, 62, 48),
  AvatarKey         = c("A1", "A2", "A3"),
  AgeAtLastContact  = c(57, 63, 50),
  PrimaryMet = c("Primary", "Met", "Primary"),
  vitalstatus       = c(1, 0, 1),
  days              = c(400, 150, 600)
)

# Bundle into lists and run
TCGA.list  <- list(tcga_tax, tcga_clin)
ORIEN.list <- list(orien_tax, orien_clin)

result <- StackBarinput(TCGA.list, ORIEN.list)
```

`survivalByQuantile` *Compute cox model p-value under various threshold*

Description

Compute cox model p-value under various threshold

Usage

```
survivalByQuantile(
  input.var,
  input.tax,
  surv.dat,
  percentiles = seq(0.01, 0.99, 0.01)
)
```

Arguments

`input.var`	a character object specifying which subject survival analysis is applying to.
`input.tax`	data frame containing information about an object to be tested. The sample order must agree with surv.dat's sample order
`surv.dat`	a data frame containing all survival information. The sample order must agree with input.dat's sample order
`percentiles`	a list of numeric values specifying all of the percentiles to be tested on

Details

The threshold used in this function is to categorize data into two groups

Value

The function will generate a data frame containing hazard ratio, lower bound, upper bound, percentile, cutoff threshold, and p-value. This data frame can then be manipulated for various plots

Examples

```
set.seed(123)

## Simulated survival data
surv.dat <- data.frame(
  ID = paste0("S", 1:50),
  days = runif(50, 100, 3000),
  vitalstatus = sample(c(0, 1), 50, replace = TRUE)
)

## Simulated taxonomic abundance matrix
input.tax <- data.frame(
  genus_A = rlnorm(50, meanlog = -2, sdlog = 0.5),
  genus_B = rlnorm(50, meanlog = -1.5, sdlog = 0.6)
)

## Compute hazard ratios across quantiles
res <- survivalByQuantile(
  input.var = "genus_A",
  input.tax = input.tax,
  surv.dat  = surv.dat,
  percentiles = seq(0.2, 0.8, 0.2)
)

res
```

---

survival_plot *Make survival curve*

Description

Make survival curve

Usage

```
survival_plot(modified_input, taxlev, title.input)
```

Arguments

modified_input	the output list from 'generate_surv_input' function
taxlev	an output string object from 'generate_surv_input' specifying a taxonomy level to be plotted
title.input	a string object to speficy the title of the plot

Value

a ggplot object

Examples

```
## Example requires survival and survminer
set.seed(123)

## Simulated survival data
surv_df <- data.frame(
  days = runif(60, min = 100, max = 4000),
  vitalstatus = sample(c(0, 1), 60, replace = TRUE)
)

## Simulated taxonomic matrix (one column per taxonomic level)
tax_mat <- data.frame(
  "k__Bacteria" = rlnorm(60, meanlog = -2, sdlog = 0.5),
  "p__Firmicutes" = rlnorm(60, meanlog = -1.8, sdlog = 0.6)
)

## Simulated threshold table for each taxonomic level
thresh_k <- data.frame(
  percentile = c(0.25, 0.50, 0.75),
  pval = c(0.01, 0.05, 0.10)
)

thresh_p <- data.frame(
  percentile = c(0.30, 0.60, 0.90),
  pval = c(0.02, 0.04, 0.08)
)

## Assemble modified_input list
modified_input <- list(
  tresh = list(
    "k__Bacteria" = thresh_k,
    "p__Firmicutes" = thresh_p
  ),
  tax.mat = tax_mat,
  surv = surv_df
)

## Run survival plot
p <- survival_plot(
  modified_input = modified_input,
```

```
  taxlev = "k__Bacteria",
  title.input = "Example Cohort"
)

## View plot
p
```

ToWide *Change data frame into wide format and match survival data*

Description

Change data frame into wide format and match survival data

Usage

```
ToWide(
  data.long,
  surv.dat,
 taxalevels = c("domain", "kingdom", "phylum", "class", "order", "family", "genus",
    "species")
)
```

Arguments

data.long	input taxonomy matrix in long format
surv.dat	survival information for the data.long
taxalevels	character vector representing every taxonomy level to be changed into wide format

Value

a long format of the taxonomy data with matching survival information

Examples

```
## Reproducible example
set.seed(123)

## Simulated long-format exogenous relative abundance data
data.long <- data.frame(
  ID = rep(paste0("S", 1:10), each = 4),
  exo.ra = runif(40),
  domain  = rep("d__Bacteria", 40),
  kingdom = rep("k__Bacteria", 40),
  phylum  = rep(c("p__Firmicutes", "p__Proteobacteria"), each = 20),
  class   = rep(c("c__Bacilli", "c__Gammaproteobacteria"), each = 20),
```

```
  order   = rep("o__TestOrder", 40),
  family  = rep("f__TestFamily", 40),
  genus   = rep(c("g__A", "g__B"), times = 20),
  species = rep(c("s__A1", "s__B1"), times = 20)
)

## Simulated survival metadata
surv.dat <- data.frame(
  ID = paste0("S", 1:10),
  days = runif(10, 500, 3000),
  vitalstatus = sample(c(0, 1), 10, replace = TRUE)
)

## Convert to wide format across taxonomic levels
wide_list <- ToWide(
  data.long = data.long,
  surv.dat  = surv.dat
)

## Inspect output
names(wide_list)
wide_list$phylum
```

Index