Package ‘target’

April 22, 2026

Type Package

Title Predict Combined Function of Transcription Factors

Version 1.25.0

Description Implement the BETA algorithm for infering direct target genes from DNA-binding and perturbation expression data Wang et al. (2013) <doi:10.1038/nprot.2013.150>. Extend the algorithm to predict the combined function of two DNA-binding elements from comprable binding and expression data.

URL `https://github.com/MahShaaban/target`

BugReports `https://github.com/MahShaaban/target/issues`

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.6)

Imports BiocGenerics, GenomicRanges, IRanges, matrixStats, methods, stats, graphics, shiny

Suggests testthat (>= 2.1.0), knitr, rmarkdown, shinytest, shinyBS, covr

VignetteBuilder knitr

RoxygenNote 6.1.1

biocViews Software, StatisticalMethod, Transcription

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

git_branch devel

git_last_commit fd1fceb

git_last_commit_date 2025-10-29

Repository Bioconductor 3.24

Date/Publication 2026-04-21

Author Mahmoud Ahmed [aut, cre]

Maintainer Mahmoud Ahmed <`mahmoud.s.fahmy@students.kasralainy.edu.eg`>

Contents

`associated_peaks` *Predict associated peaks*

Description

This function selects overlapping peaks and regions, calculates the distance between them and score each peak.

Usage

```
associated_peaks(peaks, regions, regions_col, base = 1e+05)
```

Arguments

`peaks`	A GRanges object
`regions`	A GRanges object
`regions_col`	A character string
`base`	An integer to calculate distances relative to.

Value

A GRanges object. A similar object to peaks with three added metadata columns.

Examples

```
# load peaks and transcripts data
data("real_peaks")
data("real_transcripts")

# associated peaks
ap <- associated_peaks(real_peaks, real_transcripts, 'name2')
```

`direct_targets` *Predict direct targets*

Description

This function selects overlapping peaks and regions, calculates the distance between them, score each peak and region and calculate rank products of the regions.

Usage

```
direct_targets(peaks, regions, regions_col, stats_col, base = 1e+05)
```

Arguments

`peaks`	A GRanges object
`regions`	A GRanges object
`regions_col`	A character string
`stats_col`	A character string
`base`	An integer to calculate distances relative to.

Value

A GRanges object. A similar object to regions with several added metadata columns.

Examples

```
# load peaks and transcripts data
data("real_peaks")
data("real_transcripts")

# direct targets
dt <- direct_targets(real_peaks, real_transcripts, 'name2', 't')
```

`find_distance` *Find the distance between peaks and regions*

Description

Calculate the distance between the elements of two GRanges objects.

Usage

```
find_distance(peaks, regions, how = "center")
```

Arguments

`peaks`	A GRanges object
`regions`	A GRanges object
`how`	A character string, default 'center'

Value

A vector of integers

Examples

```
library(IRanges)

query <- IRanges(c(1, 4, 9), c(5, 7, 10))
subject <- IRanges(c(2, 2, 10), c(2, 3, 12))
find_distance(query, subject)
```

`merge_ranges` *Merge peaks and regions GRanges*

Description

Merge two GRanges objects by overlaps

Usage

```
merge_ranges(peaks, regions)
```

Arguments

`peaks`	A GRanges object
`regions`	A GRanges object

Value

A DataFrame

Examples

```
library(IRanges)

query <- IRanges(c(1, 4, 9), c(5, 7, 10))
subject <- IRanges(c(2, 2, 10), c(2, 3, 12))
mergeByOverlaps(query, subject)
```

`plot_predictions` *Plot the ECDF of ranks by groups*

Description

Plot the cumulative distribution function of choosen value (e.g. ranks) by a factor of the same lenght, group. Each group is given a color and a label.

Usage

```
plot_predictions(rank, group, colors, labels, ...)
```

Arguments

`rank`	A numeric vector
`group`	A factor of length equal that of rank
`colors`	A character vector of colors for each group
`labels`	A character vector of length equal the unique values in groups
`...`	Other arguments passed to points

Value

NULL.

Examples

```
# generate random values
rn1 <- rnorm(100)
rn2 <- rnorm(100, 2)
e <- c(rn1, rn2)

# generate grouping variable
g <- rep(c('up', 'down'), times = c(length(rn1), length(rn2)))

plot_predictions(e,
                 group = g,
                 colors = c('red', 'green'),
                 labels = c('up', 'down'))
```

`rank_product` *Calculate the regions rank products*

Description

Calculate the rank products of the rank of the distances and the statistics.

Usage

```
rank_product(region_score, region_stat, region_id)
```

Arguments

`region_score`	A vector of numerics
`region_stat`	A vector of numerics
`region_id`	A vector of characters

Value

A vector of numerics

Examples

```
library(IRanges)

query <- IRanges(c(1, 4, 9), c(5, 7, 10))
subject <- IRanges(c(2, 2, 10), c(2, 3, 12))
distance <- find_distance(query, subject)
peak_score <- score_peaks(distance, 100000)
region_id <- c('region1', 'region1', 'region2')
region_score <- score_regions(peak_score, region_id)
region_stat <- c(30, 30, -40)
rank_product(region_score, region_stat, region_id)
```

`real_peaks` *AR peaks in LNCaP cell line*

Description

Androgen recepor peaks from ChIP-Seq experiment in the LNCaP cell line.

Usage

```
real_peaks
```

Format

A `GRanges`

Source

`https://github.com/suwangbio/BETA/blob/master/BETA_test_data/3656_peaks.bed`

See Also

`real_transcripts`

`sim_peaks`

Examples

```
# load data
data('real_peaks')

# locate the raw data
system.file('extdata', '3656_peaks.bed.gz', package = 'target')

# locate the source code for preparing the data
system.file('extdata', 'make-data.R', package = 'target')
```

`real_transcripts` *Differential expression of DHT treated LNCaP cell line*

Description

The differential expression analysis output of LNCaP cell line treated with DHT for 16 hours compared to non-treated cells. The REFSEQ transcript identifiers were used to merge the data.frame with the transcript coordinates from the hg19 reference genome.

Usage

```
real_transcripts
```

Format

A `GRanges`

Source

`https://github.com/suwangbio/BETA/blob/master/BETA_test_data/AR_diff_expr.xls`

`https://github.com/suwangbio/BETA/blob/master/BETA_1.0.7/BETA/references/hg19.refseq`

See Also

`real_peaks`

`sim_transcripts`

Examples

```
# load data
data('real_transcripts')

# locate the raw data
system.file('extdata', 'AR_diff_expr.tsv.gz', package = 'target')
system.file('extdata', 'hg19.refseq', package = 'target')

# locate the source code for preparing the data
system.file('extdata', 'make-data.R', package = 'target')
```

---

score_peaks *Calculate peak scores*

Description

Calculate the peak score based on the distance to a region of interest.

Usage

```
score_peaks(distance, base)
```

Arguments

`distance`	A vector of integers
`base`	An integer to calculate distances relative to.

Value

A vector of integers

Examples

```
library(IRanges)

query <- IRanges(c(1, 4, 9), c(5, 7, 10))
subject <- IRanges(c(2, 2, 10), c(2, 3, 12))
distance <- find_distance(query, subject)
score_peaks(distance, 100000)
```

---

score_regions *Calculate region scores*

Description

Calculate the region score based on the distance to their assigned peaks.

Usage

```
score_regions(peak_score, region_id)
```

Arguments

`peak_score`	A vector of integers
`region_id`	A vector of character

Value

A vector of numerics

Examples

```
library(IRanges)

query <- IRanges(c(1, 4, 9), c(5, 7, 10))
subject <- IRanges(c(2, 2, 10), c(2, 3, 12))
distance <- find_distance(query, subject)
peak_score <- score_peaks(distance, 100000)
region_id <- c('region1', 'region1', 'region2')
region_score <- score_regions(peak_score, region_id)
```

`sim_peaks` *Simulated peaks*

Description

is randomly generated peaks with random distances from the transcripts start sites (TSS) of chromosome 1 of the mm10 mouse genome.

Usage

```
sim_peaks
```

Format

A `GRanges`

See Also

`real_peaks`

`sim_transcripts`

Examples

```
# load data
data('sim_peaks')

# locate the source code for preparing the data
system.file('extdata', 'make-data.R', package = 'target')
```

`sim_transcripts` *Simulated transcripts The transcripts chromosome 1 of the mm10 mouse genome with randomly singed statistics assigned to each.*

Description

Simulated transcripts The transcripts chromosome 1 of the mm10 mouse genome with randomly singed statistics assigned to each.

Usage

```
sim_transcripts
```

Format

A `GRanges`

See Also

`real_transcripts`

`sim_transcripts`

Examples

```
# load data
data('sim_transcripts')

# locate the source code for preparing the data
system.file('extdata', 'make-data.R', package = 'target')
```

`target` `target`*: Predict Combined Function of Transcription Factors.*

Description

Implement the BETA algorithm for infering direct target genes from DNA-binding and perturbation expression data Wang et al. (2013) <doi: 10.1038/nprot.2013.150>. Extend the algorithm to predict the combined effect of two DNA-binding elements from comprable binding and expression data.

Details

Predicting associated peaks and direct targets

`associated_peaks` `direct_targets`

Plotting and testing predictions `plot_predictions` `test_predictions`

Internal `target` functions: `merge_ranges` `find_distance` `score_peaks` `score_regions` `rank_product`

`target_app` *Run the shiny App*

Description

Run the shiny App

Usage

```
target_app()
```

Value

Runs the shiny app

`test_predictions` *Test the ECDF ranks of groups are from same distribution*

Description

Test whether the cumulative distribution function of two groups are drawn from the same distribution.

Usage

```
test_predictions(rank, group, compare, ...)
```

Arguments

`rank`	A numeric vector
`group`	A factor of length equal that of rank
`compare`	A character vector of length two
`...`	Other arguments passed to ks.test

Value

An htest object

Examples

```
# generate random values
rn1 <- rnorm(100)
rn2 <- rnorm(100, 2)
e <- c(rn1, rn2)

# generate grouping variable
g <- rep(c('up', 'down'), times = c(length(rn1), length(rn2)))

# test
test_predictions(e,
```

```
              group = g,
              compare = c('up', 'down'))
```

Index