

Package ‘FindIT2’

May 24, 2024

Title find influential TF and Target based on multi-omics data

Version 1.11.0

Description This package implements functions to find influential TF and target based on different input type. It have five module:
Multi-peak multi-gene annotaion(mmPeakAnno module),
Calculate regulation potential(calcRP module),
Find influential Target based on ChIP-Seq and RNA-Seq data(Find influential Target module),
Find influential TF based on different input(Find influential TF module),
Calculate peak-gene or peak-peak correlation(peakGeneCor module).
And there are also some other useful function like integrate different source information, calculate jaccard similarity for your TF.

License Artistic-2.0

URL <https://github.com/shangguandong1996/FindIT2>

BugReports <https://support.bioconductor.org/t/FindIT2>

biocViews Software, Annotation, ChIPSeq, ATACSeq, GeneRegulation,
MultipleComparison, GeneTarget

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

Suggests BiocStyle, knitr, rmarkdown, sessioninfo, testthat (>= 3.0.0), TxDb.Athaliana.BioMart.plantsmart28

VignetteBuilder knitr

Depends GenomicRanges, R (>= 3.5.0)

Imports withr, BiocGenerics, GenomeInfoDb, rtracklayer, S4Vectors,
GenomicFeatures, dplyr, rlang, patchwork, ggplot2,
BiocParallel, qvalue, stringr, utils, stats, ggrepel, tibble,
tidyr, SummarizedExperiment, MultiAssayExperiment, IRanges,
progress, purrr, glmnet, methods

Config/testthat/edition 3

git_url <https://git.bioconductor.org/packages/FindIT2>

git_branch devel
git_last_commit f468499
git_last_commit_date 2024-04-30
Repository Bioconductor 3.20
Date/Publication 2024-05-24
Author Guandong Shang [aut, cre] (<<https://orcid.org/0000-0002-9509-0314>>)
Maintainer Guandong Shang <shangguandong1996@163.com>

Contents

| | |
|---------------------------------------|----|
| ATAC_normCount | 3 |
| calcRP_coverage | 3 |
| calcRP_region | 4 |
| calcRP_TFHit | 6 |
| enhancerPromoterCor | 7 |
| findIT_enrichFisher | 8 |
| findIT_enrichWilcox | 9 |
| findIT_MARA | 10 |
| findIT_regionRP | 11 |
| findIT_TFHit | 13 |
| findIT_TTPair | 14 |
| getAssocPairNumber | 15 |
| integrate_ChIP_RNA | 16 |
| integrate_replicates | 17 |
| jaccard_findIT_enrichFisher | 18 |
| jaccard_findIT_TTPair | 19 |
| loadPeakFile | 20 |
| mm_geneBound | 21 |
| mm_geneScan | 22 |
| mm_nearestGene | 23 |
| peakGeneCor | 23 |
| plot_annoDistance | 25 |
| plot_peakGeneAlias_summary | 25 |
| plot_peakGeneCor | 26 |
| RNADiff_LEC2_GR | 28 |
| RNA_normCount | 28 |
| test_featureSet | 29 |
| test_geneSet | 29 |
| TF_target_database | 30 |

Index

32

| | |
|----------------|---|
| ATAC_normCount | <i>ATAC normCount of E50h-72h in Chr5</i> |
|----------------|---|

Description

ATAC normCount of E50h-72h in Chr5

Usage

```
data(ATAC_normCount)
```

Format

A matrix

Source

<https://doi.org/10.1016/j.devcel.2020.07.003>

| | |
|-----------------|------------------------|
| calcRP_coverage | <i>calcRP_coverage</i> |
|-----------------|------------------------|

Description

calculate regulatory potential using big wig files, which is useful for ATAC or H3K27ac histone modification data.

Usage

```
calcRP_coverage(  
  bwFile,  
  Txdb,  
  gene_included,  
  Chrs_included,  
  decay_dist = 1000,  
  scan_dist = 20000,  
  verbose = TRUE  
)
```

Arguments

| | |
|---------------|---|
| bwFile | bw file |
| Txdb | Txdb |
| gene_included | a character vector which represent gene set which you want to calculate RP for |
| Chrs_included | a character vector which represent chromosomes where you want to calculate gene RP in |
| decay_dist | decay distance |
| scan_dist | scan distance |
| verbose | whether you want to report detailed running message |

Details

Please note that because of `rtracklayer::import` has some issue on 32 bit R of windows, so the `calcRP_coverage` can not work on this system. But if your R is 64 bit, which now be applied on the most windows R, this function still work.

Value

data.frame

Examples

```
if (.Platform$OS.type != "windows" & require(Txdb.Athaliana.BioMart.plantsmart28)) {
  Txdb <- Txdb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))
  bwFile <- system.file("extdata", "E50h_sampleChr5.bw", package = "FindIT2")

  RP_df <- calcRP_coverage(
    bwFile = bwFile,
    Txdb = Txdb,
    Chrs_included = "Chr5"
  )
}
```

calcRP_region

calcRP_region

Description

calculate regulatory potential based on `mm_geneScan` result and `peakCount` matrix, which is useful for ATAC or H3K27ac histone modification data.

Usage

```
calcRP_region(
  mmAnno,
  peakScoreMt,
  Txdb,
  Chrs_included,
  decay_dist = 1000,
  log_transform = FALSE,
  verbose = TRUE
)
```

Arguments

| | |
|---------------|--|
| mmAnno | the annotated GRange object from mm_geneScan |
| peakScoreMt | peak count matrix. The rownames are feature_id in mmAnno, while the colnames are sample names |
| Txdb | Txdb |
| Chrs_included | a character vector which represent chromosome where you want to calculate gene RP in. If Chromosome is not be set, it will calculate gene RP in all chromosomes in Txdb. |
| decay_dist | decay distance |
| log_transform | whether you want to log and norm your RP |
| verbose | whether you want to report detailed running message |

Value

a MultiAssayExperiment object containg detailed peak-RP-gene relationship and sumRP info

Examples

```
if (require(TxDb.Athaliana.BioMart.plantsmart28)) {
  data("ATAC_normCount")
  library(SummarizedExperiment)
  Txdb <- TxDb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))

  peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)
  mmAnno <- mm_geneScan(peak_GR, Txdb)

  regionRP <- calcRP_region(
    mmAnno = mmAnno,
    peakScoreMt = ATAC_normCount,
    Txdb = Txdb,
    Chrs_included = "Chr5"
  )

  sumRP <- assays(regionRP)$sumRP
}
```

```

    fullRP <- assays(regionRP)$fullRP
  }

```

 calcRP_TFHit

calcRP_TFHit

Description

calculate regulatory potential based on ChIP-Seq peak data, which is useful for TF ChIP-seq data.

Usage

```

calcRP_TFHit(
  mmAnno,
  Txdb,
  decay_dist = 1000,
  report_fullInfo = FALSE,
  verbose = TRUE
)

```

Arguments

| | |
|-----------------|---|
| mmAnno | the annotated GRange object from mm_geneScan |
| Txdb | Txdb |
| decay_dist | decay distance |
| report_fullInfo | whether you want to report full peak-RP-gene info |
| verbose | whether you want to report detailed running message |

Details

If your origin peak_GR of mmAnno have column named feature_score, calcRP_TFHit will consider this column when calculating sumRP. Otherwise, it will consider all peak Hit feature_score is 1.

Value

if report_fullInfo is TRUE, it will output GRanges with detailed info. While FALSE, it will output data frame

Examples

```

if (require(TxDb.Athaliana.BioMart.plantsmart28)){
  Txdb <- TxDb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))
  peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)
  mmAnno <- mm_geneScan(peak_GR, Txdb)

  # if you just want to get RP_df, you can set report_fullInfo FALSE
  fullRP_hit <- calcRP_TFHit(
    mmAnno = mmAnno,
    Txdb = Txdb,
    report_fullInfo = TRUE
  )

  RP_df <- metadata(fullRP_hit)$peakRP_gene
}

```

enhancerPromoterCor *enhancerPromoterCor*

Description

enhancerPromoterCor

Usage

```

enhancerPromoterCor(
  peak_GR,
  Txdb,
  up_scanPromoter = 500,
  down_scanPromoter = 500,
  up_scanEnhancer = 20000,
  down_scanEnhancer = 20000,
  peakScoreMt,
  parallel = FALSE,
  verbose = TRUE
)

```

Arguments

| | |
|-------------------|---|
| peak_GR | peak GRange with a column named feature_id representing you peak name |
| Txdb | Txdb |
| up_scanPromoter | the scan distance which is used to scan nearest promoter |
| down_scanPromoter | the scan distance which is used to scan nearest promoter |

up_scanEnhancer the scan distance which is used to scan feature
 down_scanEnhacner the scan distance which is used to scan feature
 peakScoreMt peak count matrix. The rownames are feature_id in peak_GR
 parallel whether you want to parallel to speed up
 verbose whether you want to report detailed running message

Value

mmAnno with Cor, pvalue, padj, qvalue column

Examples

```
if (require(Txdb.Athaliana.BioMart.plantsmart28)){
  data("ATAC_normCount")
  Txdb <- Txdb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))
  peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)[1:100]
  mm_ePLink <- enhancerPromoterCor(
    peak_GR = peak_GR,
    Txdb = Txdb,
    peakScoreMt = ATAC_normCount,
    parallel = FALSE)
}
```

findIT_enrichFisher *findI(nfluentia)T(F)_enrichFisher*

Description

find influential TF of your input peak set compared with your whole peak sets based on TF ChIP-Seq or motif data.

Usage

```
findIT_enrichFisher(input_feature_id, peak_GR, TF_GR_database)
```

Arguments

input_feature_id a character vector which represent peaks set which you want to find influential TF for
 peak_GR a GRange object represent your whole feature location with a column named feature_id, which your input_feature_id should a part of it.
 TF_GR_database TF peak GRange with a column named TF_id representing you TF name

Value

data.frame

Examples

```

data("test_featureSet")
peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
peak_GR <- loadPeakFile(peak_path)
ChIP_peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
ChIP_peak_GR <- loadPeakFile(ChIP_peak_path)
ChIP_peak_GR$TF_id <- "AT1G28300"

result_findIT_enrichFisher <- findIT_enrichFisher(
  input_feature_id = test_featureSet,
  peak_GR = peak_GR,
  TF_GR_database = ChIP_peak_GR
)

```

findIT_enrichWilcox *findIT_enrichWilcox*

Description

findIT_enrichWilcox

Usage

```

findIT_enrichWilcox(
  input_feature_id,
  peak_GR,
  TF_GR_database,
  background_peaks = NULL,
  background_number = 3000
)

```

Arguments

| | |
|-------------------|--|
| input_feature_id | a character vector which represent peaks set which you want to find influential TF for |
| peak_GR | a GRange object represent your whole feature location with a column named feature_id, which your input_feature_id should a part of it. |
| TF_GR_database | TF peak GRange with a column named TF_id representing you TF name |
| background_peaks | a character vector which represent background peak set. If you do not assign background peaks, program will sample background_number peaks as background peaks from all feature_id in your peak_GR |
| background_number | background peaks number |

Value

data.frame

Examples

```
data("test_featureSet")
peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
peak_GR <- loadPeakFile(peak_path)
ChIP_peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
ChIP_peak_GR <- loadPeakFile(ChIP_peak_path)
ChIP_peak_GR$TF_id <- "AT1G28300"

result_findIT_enrichWilcox <- findIT_enrichWilcox(
  input_feature_id = test_featureSet,
  peak_GR = peak_GR,
  TF_GR_database = ChIP_peak_GR
)
```

findIT_MARA

findIT_MARA

Description

findIT_MARA

Usage

```
findIT_MARA(
  input_feature_id,
  peak_GR,
  peakScoreMt,
  TF_GR_database,
  log = TRUE,
  meanScale = TRUE,
  output = c("coef", "cor"),
  verbose = TRUE
)
```

Arguments

| | |
|------------------|---|
| input_feature_id | a character vector which represent peaks set which you want to find influential TF for |
| peak_GR | a GRRange object represent your whole feature location with a column named feature_id, which your input_feature_id should a part of it. |
| peakScoreMt | peak count matrix. |

TF_GR_database TF peak GRRange with a column named TF_id representing you TF name. If you have TF_score column, MARA will consider it. otherwise, MARA will consider each hit is 1.

log whether you want to log your peakScoreMt

meanScale whether you want to mean-centered per row

output one of 'coef' and 'cor'. Default is coef

verbose whether you want to report detailed running message

Value

a data.frame

Examples

```
data("ATAC_normCount")
data("test_featureSet")

peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
peak_GR <- loadPeakFile(peak_path)

ChIP_peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
ChIP_peak_GR <- loadPeakFile(ChIP_peak_path)
ChIP_peak_GR$TF_id <- "AT1G28300"

set.seed(20160806)

result_findIT_MARA <- findIT_MARA(
  input_feature_id = test_featureSet,
  peak_GR = peak_GR,
  peakScoreMt = ATAC_normCount,
  TF_GR_database = ChIP_peak_GR
)
```

findIT_regionRP *findI(nfluentia)T(F)_regionRP*

Description

find Influential TF of your input gene set based on regulatory potential data and TF ChIP-Seq or motif data

Usage

```
findIT_regionRP(
  regionRP,
  Txdb,
  TF_GR_database,
```

```

    input_genes,
    background_genes = NULL,
    background_number = 3000,
    verbose = TRUE
  )

```

Arguments

| | |
|-------------------|---|
| regionRP | the MultiAssayExperiment object from calcRP_region |
| Txdb | Txdb |
| TF_GR_database | TF peak GRange with a column named TF_id representing you TF name |
| input_genes | a character vector which represent genes set which you want to find influential TF for |
| background_genes | a character vector which represent background genes set. If you do not assign background gene , program will sample background_number genes as background genes from all gene sets. |
| background_number | background genes number |
| verbose | whether you want to report detailed running message |

Value

a MultiAssayExperiment object containing detailed TF-percent and TF-pvalue

Examples

```

if (require(Txdb.Athaliana.BioMart.plantsmart28)) {
  data("ATAC_normCount")
  data("test_geneSet")
  Txdb <- Txdb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))

  peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)

  ChIP_peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  ChIP_peak_GR <- loadPeakFile(ChIP_peak_path)
  ChIP_peak_GR$TF_id <- "AT1G28300"

  mmAnno <- mm_geneScan(peak_GR, Txdb)

  regionRP <- calcRP_region(
    mmAnno = mmAnno,
    peakScoreMt = ATAC_normCount,
    Txdb = Txdb,
    Chrs_included = "Chr5"
  )

  set.seed(20160806)
}

```

```

    result_findIT_regionRP <- findIT_regionRP(
      regionRP = regionRP,
      Txdb = Txdb,
      TF_GR_database = ChIP_peak_GR,
      input_genes = test_geneSet,
      background_number = 3000
    )
  }

```

 findIT_TFHit

findI(nfluentia)T(F)_TFHit

Description

find influential TF of your input gene set based on TF ChIP-Seq or motif data

Usage

```

findIT_TFHit(
  input_genes,
  Txdb,
  TF_GR_database,
  scan_dist = 20000,
  decay_dist = 1000,
  Chrs_included,
  background_genes = NULL,
  background_number = 3000,
  verbose = TRUE
)

```

Arguments

| | |
|-------------------|---|
| input_genes | a character vector which represent genes set which you want to find influential TF for |
| Txdb | Txdb |
| TF_GR_database | TF peak GRRange with a column named TF_id representing you TF name |
| scan_dist | scan distance |
| decay_dist | decay distance |
| Chrs_included | a character vector represent chromosomes which you want to sample background genes from |
| background_genes | a character vector which represent background genes set. If you do not assign background gene , program will sample background_number genes as background genes from all gene sets. |
| background_number | background genes number |
| verbose | whether you want to report detailed running message |

Value

data.frame

Examples

```

if (require(Txdb.Athaliana.BioMart.plantsmart28)) {
  data("test_geneSet")
  Txdb <- Txdb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))

  CHIP_peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  CHIP_peak_GR <- loadPeakFile(CHIP_peak_path)
  CHIP_peak_GR$TF_id <- "AT1G28300"

  set.seed(20160806)
  result_findIT_TFHit <- findIT_TFHit(
    input_genes = test_geneSet,
    Txdb = Txdb,
    TF_GR_database = CHIP_peak_GR
  )
}

```

findIT_TTPair

*findI(nfluentia)T(F)_T(F)T(arget)Pair***Description**

find influential TF of your input gene set based on public TF-Target data

Usage

```

findIT_TTPair(
  input_genes,
  TF_target_database,
  gene_background = NULL,
  TFHit_min = 5,
  TFHit_max = 10000
)

```

Arguments

`input_genes` a character vector which represent genes set which you want to find influential TF for

`TF_target_database` TF_target pair data with two column named TF_id and target_gene

`gene_background` a character vector represent your bakcaground gene. If you do not assign back-ground gene, program will consider all target gene as background

TFHit_min minimal size of target gene regulated by TF
TFHit_max maximal size of target gene regulated by TF

Value

data.frame

Examples

```
data("TF_target_database")
data("test_geneSet")

result_findIT_TTPair <- findIT_TTPair(
  input_genes = test_geneSet,
  TF_target_database = TF_target_database
)
```

getAssocPairNumber *getAssocPairNumber*

Description

get associated peak number of gene and vice verse.

Usage

```
getAssocPairNumber(
  mmAnno,
  output_type = c("gene_id", "feature_id"),
  output_summary = FALSE
)
```

Arguments

mmAnno the annotated GRange object from mm_geneScan or mm_nearestGene
output_type one of 'gene_id' or 'feature_id'
output_summary whether you want to detailed info

Value

data.frame

Examples

```

if (require(Txdb.Athaliana.BioMart.plantsmart28)) {
  Txdb <- Txdb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))

  peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)
  peakAnno <- mm_nearestGene(peak_GR, Txdb)

  getAssocPairNumber(peakAnno)
}

```

```
integrate_ChIP_RNA    integrate_ChIP_RNA
```

Description

integrate ChIP-Seq and RNA-Seq data to find TF target genes

Usage

```

integrate_ChIP_RNA(
  result_geneRP,
  result_geneDiff,
  lfc_threshold = 1,
  padj_threshold = 0.05
)

```

Arguments

`result_geneRP` the simplify result from `calcRP_TFHit(report_fullInfo = FALSE)` or `RP_df <- metadata(fullRP_hit)$peakRP_gene`.

`result_geneDiff` the result from RNA diff result with three column `gene_id`, `log2FoldChange`, `padj`

`lfc_threshold` the threshold which decide significant genes

`padj_threshold` the threshold which decide significant genes

Value

a ggplot object if having significant genes in your result. If not, it will report a data.frame with integrated info.

Examples

```

if (require(TxDb.Athaliana.BioMart.plantsmart28)) {
  data("RNADiff_LEC2_GR")
  Txdb <- TxDb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))
  peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)
  mmAnno <- mm_geneScan(peak_GR, Txdb)

  result_geneRP <- calcRP_TFHit(
    mmAnno = mmAnno,
    Txdb = Txdb
  )
  # output a plot
  merge_data <- integrate_ChIP_RNA(
    result_geneRP = result_geneRP,
    result_geneDiff = RNADiff_LEC2_GR
  )
  # if you want to extract merge target data
  target_data <- merge_data$data
}

```

integrate_replicates *integrate_replicates*

Description

integrate value from replicates

Usage

```

integrate_replicates(
  mt,
  colData,
  fun = NULL,
  type = c("value", "rank", "rank_zscore", "pvalue")
)

```

Arguments

| | |
|---------|--|
| mt | value matrix |
| colData | a data.frame with a single column named with "type". Rows of colData correspond to columns of mt. |
| fun | the function you want to use. If set NULL, program will decide integrate method according to your 'type' parameter. |
| type | one of 'value', 'rank', 'rank_zscore', 'pvalue'. value will use mean to integrate replicates, rank will use product, rank_zscore will use Stouffer's method and pvalue will use CCT(Cauchy distribution) |

Value

matrix

Examples

```
mt <- matrix(runif(100, 0, 1), nrow = 10)
colnames(mt) <- paste0(paste0("type", 1:5), "_", rep(1:2, 5))
rownames(mt) <- paste0("TF", 1:10)
```

```
colData <- data.frame(
  type = gsub("_[0-9]", "", colnames(mt)),
  row.names = colnames(mt)
)
```

```
integrate_replicates(mt, colData, type = "value")
```

jaccard_findIT_enrichFisher

jaccard_findIT_enrichFisher

Description

jaccard_findIT_enrichFisher

Usage

```
jaccard_findIT_enrichFisher(
  input_feature_id,
  peak_GR,
  TF_GR_database,
  input_TF_id
)
```

Arguments

| | |
|------------------|---|
| input_feature_id | a character vector which represent peaks set which you want to find influential TF for (same as your find_IT_enrichFisher parameter) |
| peak_GR | a GRRange object represent your whole feature location with a column named feature_id, which your input_feature_id should a part of it. |
| TF_GR_database | TF peak GRRange with a column named TF_id representing you TF name |
| input_TF_id | TF_id which you want to calculate jaccard index for |

Value

jaccard similarity matrix

Examples

```

data("test_featureSet")
peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
peak_GR <- loadPeakFile(peak_path)

ChIP_peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
ChIP_peak_GR <- loadPeakFile(ChIP_peak_path)
ChIP_peak_GR$TF_id <- "AT1G28300"
result_findIT_enrichFisher <- findIT_enrichFisher(
  input_feature_id = test_featureSet,
  peak_GR = peak_GR,
  TF_GR_database = ChIP_peak_GR
)

jaccard_findIT_enrichFisher(
  input_feature_id = test_featureSet,
  peak_GR = peak_GR,
  TF_GR_database = ChIP_peak_GR,
  input_TF_id = result_findIT_enrichFisher$TF_id[1]
)

```

jaccard_findIT_TTpair *jaccard_findIT_TTpair*

Description

jaccard_findIT_TTpair

Usage

```
jaccard_findIT_TTpair(input_genes, TF_target_database, input_TF_id)
```

Arguments

input_genes a character vector which represent gene set which you want to find influential TF for (same as your find_IT_TTpair parameter)

TF_target_database TF_target pair data

input_TF_id TF_id which you want to calculate jaccard index for

Value

jaccard similarity matrix

Examples

```

data("TF_target_database")
data("test_geneSet")
result_findIT_TTPair <- findIT_TTPair(
  input_genes = test_geneSet,
  TF_target_database = TF_target_database
)

jaccard_findIT_TTPair(
  input_genes = test_geneSet,
  TF_target_database = TF_target_database,
  input_TF_id = result_findIT_TTPair$TF_id[1:3]
)

```

loadPeakFile

loadPeakFile

Description

read peak file and transform it into GRanges object

Usage

```
loadPeakFile(filePath, TFBS_database = FALSE)
```

Arguments

| | |
|---------------|--|
| filePath | peak Path |
| TFBS_database | whether your peak file is a TFBS database file. If you want the final GRanges have a column named "TF_id", you should set TFBS_database TRUE. The GRanges with TF_id can be applied in "TF_GR_database" parameter of findIT_TFHit, findIT_enrichFisher, findIT_enrichWilcox, findIT_regionRP. If FALSE, the GRanges will have a column named "feature_id", which always be the input of "peak_GR" parameter. |

Details

The GRanges with TF_id always be the input of "TF_GR_database" parameter. It represents the TFBS database like motif scan result, public database ChIP-seq site and so on.

The GRanges with feature_id always be the input of "peak_GR" parameter.

Value

GRanges object with a column named feature_id or TF_id

Examples

```
peakfile <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
loadPeakFile(peakfile)
```

| | |
|--------------|---------------------|
| mm_geneBound | <i>mm_geneBound</i> |
|--------------|---------------------|

Description

find related peaks of your input genes, which is useful when you want to plot volcano plot or heatmap of peaks.

Usage

```
mm_geneBound(peak_GR, Txdb, input_genes, verbose = TRUE, ...)
```

Arguments

| | |
|-------------|--|
| peak_GR | peak GRange with a column named feature_id representing you peak name |
| Txdb | Txdb |
| input_genes | a character vector which represent genes set which you want to find related peak for |
| verbose | whether you want to report detailed running message |
| ... | additional arguments in distanceToNearest |

Value

data.frame with three column: related peak id, your input gene id, and distance

Examples

```
if (require(TxDb.Athaliana.BioMart.plantsmart28)) {
  Txdb <- TxDb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))
  peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)
  peak_pair <- mm_geneBound(peak_GR, Txdb, c("AT5G01015", "AT5G67570"))
  peak_pair
}
```

| | |
|-------------|--------------------|
| mm_geneScan | <i>mm_geneScan</i> |
|-------------|--------------------|

Description

Annotate peaks using geneScan mode, which means every peak have more than one related genes.

Usage

```
mm_geneScan(
  peak_GR,
  Txdb,
  upstream = 3000,
  downstream = 3000,
  reportGeneInfo = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

| | |
|----------------|---|
| peak_GR | peak GRange with a column named feature_id representing you peak name |
| Txdb | Txdb |
| upstream | distance to start site(upstream) |
| downstream | distance to start site(downstream) |
| reportGeneInfo | whether you want to add gene info |
| verbose | whether you want to report detailed running message |
| ... | additional arguments in findOverlaps |

Value

Granges object with annotated info

Examples

```
if (require(TxDb.Athaliana.BioMart.plantsmart28)) {
  Txdb <- TxDb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))
  peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)
  peakAnno <- mm_geneScan(peak_GR, Txdb)
  peakAnno
}
```

| | |
|----------------|-----------------------|
| mm_nearestGene | <i>mm_nearestGene</i> |
|----------------|-----------------------|

Description

Annotate peaks using nearest gene mode, which means every peak only have one related gene.

Usage

```
mm_nearestGene(peak_GR, Txdb, reportGeneInfo = FALSE, verbose = TRUE, ...)
```

Arguments

| | |
|----------------|---|
| peak_GR | peak GRange with a column named feature_id representing you peak name |
| Txdb | Txdb |
| reportGeneInfo | whether you want to report full gene info |
| verbose | whether you want to report detailed running message |
| ... | additional arguments in distanceToNearest |

Value

Granges object with annotated info

Examples

```
if (require(Txdb.Athaliana.BioMart.plantsmart28)) {
  Txdb <- Txdb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))

  peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)
  peakAnno <- mm_nearestGene(peak_GR, Txdb)
  peakAnno
}
```

| | |
|-------------|--------------------|
| peakGeneCor | <i>peakGeneCor</i> |
|-------------|--------------------|

Description

peakGeneCor

Usage

```
peakGeneCor(mmAnno, peakScoreMt, geneScoreMt, parallel = FALSE, verbose = TRUE)
```

Arguments

| | |
|-------------|--|
| mmAnno | the annotated GRange object from mm_geneScan or mm_nearestGene |
| peakScoreMt | peak count matrix. The rownames are feature_id in mmAnno, while the colnames are sample names. |
| geneScoreMt | gene count matrix. The rownames are gene_id in mmAnno, while the colnames are sample names. |
| parallel | whether you want to use bplapply to speed up calculation |
| verbose | whether you want to report detailed running message |

Value

mmAnno with Cor, pvalue, padj, qvalue column

Examples

```

if (require(TxDb.Athaliana.BioMart.plantsmart28)){
  Txdb <- TxDb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))
  data("RNA_normCount")
  data("ATAC_normCount")
  peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)[1:100]
  mmAnno <- mm_geneScan(peak_GR, Txdb)

  ATAC_colData <- data.frame(
    row.names = colnames(ATAC_normCount),
    type = gsub("_R[0-9]", "", colnames(ATAC_normCount))
  )

  ATAC_normCount_merge <- integrate_replicates(ATAC_normCount, ATAC_colData)
  RNA_colData <- data.frame(
    row.names = colnames(RNA_normCount),
    type = gsub("_R[0-9]", "", colnames(RNA_normCount))
  )

  RNA_normCount_merge <- integrate_replicates(RNA_normCount, RNA_colData)
  mmAnnoCor <- peakGeneCor(
    mmAnno = mmAnno,
    peakScoreMt = ATAC_normCount_merge,
    geneScoreMt = RNA_normCount_merge,
    parallel = FALSE
  )

  mmAnnoCor
}

```

plot_annoDistance *plot_annoDistance*

Description

plot the distance distribution of mmAnno from mm_nearestGene, which helps you decide whether your TF is promoter or enhancer dominant

Usage

```
plot_annoDistance(mmAnno, quantile = c(0.01, 0.99))
```

Arguments

mmAnno the annotated GRange object from mm_nearestGene
quantile the quantile of distanceToTSS you want to show

Value

a ggplot2 object

Examples

```
if (require(Txdb.Athaliana.BioMart.plantsmart28)) {  
  Txdb <- Txdb.Athaliana.BioMart.plantsmart28  
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))  
  
  peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")  
  peak_GR <- loadPeakFile(peak_path)  
  peakAnno <- mm_nearestGene(peak_GR, Txdb)  
  plot_annoDistance(peakAnno)  
  
}
```

plot_peakGeneAlias_summary
 plot_peakGeneAlias_summary

Description

plot_peakGeneAlias_summary

Usage

```
plot_peakGeneAlias_summary(
  mmAnno,
  mmAnno_corFilter = NULL,
  output_type = c("gene_id", "feature_id"),
  fillColor = "#ca6b67"
)
```

Arguments

`mmAnno` the annotated GRange object from `mm_geneScan` or `mm_nearestGene`

`mmAnno_corFilter` the filter `mmAnno` object according to p-value or cor, default is NULL

`output_type` one of 'gene_id' or 'feature_id'

`fillColor` the bar plot color

Value

a ggplot object

Examples

```
if (require(TxDb.Athaliana.BioMart.plantsmart28)) {
  Txdb <- TxDb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))

  peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)
  peakAnno <- mm_nearestGene(peak_GR, Txdb)

  plot_peakGeneAlias_summary(peakAnno)
}
```

`plot_peakGeneCor` *plot_peakGeneCor*

Description

`plot_peakGeneCor`

Usage

```
plot_peakGeneCor(
  mmAnnoCor,
  select_gene,
  addLine = TRUE,
```

```

    addFullInfo = TRUE,
    sigShow = c("pvalue", "padj", "qvalue")
  )

```

Arguments

| | |
|-------------|---|
| mmAnnoCor | the annotated GRange object from peakGeneCor or enhancerPromoterCor |
| select_gene | a gene_id which you want to show |
| addLine | whether add cor line |
| addFullInfo | whether add full feature info on plot |
| sigShow | one of 'pvalue' 'padj' 'qvalue' |

Value

ggplot2 object

Examples

```

if (require(TxDb.Athaliana.BioMart.plantsmart28)) {
  data("RNA_normCount")
  data("ATAC_normCount")
  Txdb <- TxDb.Athaliana.BioMart.plantsmart28
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))
  peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")
  peak_GR <- loadPeakFile(peak_path)[1:100]
  mmAnno <- mm_geneScan(peak_GR, Txdb)

  ATAC_colData <- data.frame(
    row.names = colnames(ATAC_normCount),
    type = gsub("_R[0-9]", "", colnames(ATAC_normCount))
  )

  integrate_replicates(ATAC_normCount, ATAC_colData) -> ATAC_normCount_merge
  RNA_colData <- data.frame(
    row.names = colnames(RNA_normCount),
    type = gsub("_R[0-9]", "", colnames(RNA_normCount))
  )
  integrate_replicates(RNA_normCount, RNA_colData) -> RNA_normCount_merge
  mmAnnoCor <- peakGeneCor(
    mmAnno = mmAnno,
    peakScoreMt = ATAC_normCount_merge,
    geneScoreMt = RNA_normCount_merge,
    parallel = FALSE
  )

  plot_peakGeneCor(mmAnnoCor, select_gene = "AT5G01010")
}

```

RNADiff_LEC2_GR

RNA diff result from LEC2_GR VS LEC2_DMSO

Description

RNA diff result from LEC2_GR VS LEC2_DMSO

Usage

```
data(RNADiff_LEC2_GR)
```

Format

a data frame

Source

<https://doi.org/10.1016/j.devcel.2020.07.003>

RNA_normCount

RNA normCount of E50h-72h in Chr5

Description

RNA normCount of E50h-72h in Chr5

Usage

```
data(RNA_normCount)
```

Format

A matrix

Source

<https://doi.org/10.1016/j.devcel.2020.07.003>

| | |
|-----------------|------------------------|
| test_featureSet | <i>test_featureSet</i> |
|-----------------|------------------------|

Description

test_featureSet

Usage

```
data(test_featureSet)
```

Format

character vector represent your interesting feature_id set

Details

For the detailed progress producing input_feature_id, you can see ?test_geneSet

| | |
|--------------|---------------------|
| test_geneSet | <i>test_geneSet</i> |
|--------------|---------------------|

Description

test_geneSet

Usage

```
data(test_geneSet)
```

Format

character vector represent your interesting gene set

Examples

```
## Not run:  
# source  
if (require(Txdb.Athaliana.BioMart.plantsmart28)) {  
  library(FindIT2)  
  Txdb <- Txdb.Athaliana.BioMart.plantsmart28  
  seqlevels(Txdb) <- paste0("Chr", c(1:5, "M", "C"))  
  ChIP_peak_path <- system.file("extdata", "ChIP.bed.gz", package = "FindIT2")  
  ChIP_peak_GR <- loadPeakFile(ChIP_peak_path)  
  ATAC_peak_path <- system.file("extdata", "ATAC.bed.gz", package = "FindIT2")  
  ATAC_peak_GR <- loadPeakFile(ATAC_peak_path)
```

```

mmAnno_geneScan <- mm_geneScan(
  peak_GR = ChIP_peak_GR,
  Txdb = Txdb,
  upstream = 2e4,
  downstream = 2e4
)

peakRP_gene <- calcRP_TFHit(
  mmAnno = mmAnno_geneScan,
  Txdb = Txdb,
  report_fullInfo = FALSE
)

data("RNADiff_LEC2_GR")
merge_result <- integrate_ChIP_RNA(
  result_geneRP = peakRP_gene,
  result_geneDiff = RNADiff_LEC2_GR
)

target_result <- merge_result$data
test_geneSet <- target_result$gene_id[1:50]

related_peaks <- mm_geneBound(
  peak_GR = ATAC_peak_GR,
  Txdb = Txdb,
  input_genes = test_geneSet
)
test_featureSet <- unique(related_peaks$feature_id)
# save(test_geneSet, file = "data/test_geneSet.rda", version = 2)
# save(test_featureSet, file = "data/test_featureSet.rda", version = 2)
}

## End(Not run)

```

TF_target_database *TF-target database*

Description

TF-target database

Usage

```
data(TF_target_database)
```

Format

a data frame

Source

<http://bioinformatics.psb.ugent.be/webtools/iGRN/pages/download>

Examples

```
## Not run:
# source
library(dplyr)
data <- read.table("~/reference/annoation/Athaliana/TF_target/iGRN_network_full.txt",
                  sep = "\t",
                  stringsAsFactors = FALSE)

data %>%
  rename(TF_id = V1, target_gene = V2) %>%
  select(TF_id, target_gene) %>%
  TF_target_database <- filter(TF_id %in% c("AT1G28300",
    "AT5G63790", "AT5G24110", "AT3G23250")) %>%
  as.data.frame()

save(TF_target_database, file = "inst/extdata/TF_target_database.rda", version = 2,
     compress = "bzip2")

## End(Not run)
```

Index

* datasets

- ATAC_normCount, [3](#)
- RNA_normCount, [28](#)
- RNADiff_LEC2_GR, [28](#)
- test_featureSet, [29](#)
- test_geneSet, [29](#)
- TF_target_database, [30](#)

- RNA_normCount, [28](#)
- RNADiff_LEC2_GR, [28](#)
- test_featureSet, [29](#)
- test_geneSet, [29](#)
- TF_target_database, [30](#)

ATAC_normCount, [3](#)

calcRP_coverage, [3](#)
calcRP_region, [4](#)
calcRP_TFHit, [6](#)

enhancerPromoterCor, [7](#)

findIT_enrichFisher, [8](#)
findIT_enrichWilcox, [9](#)
findIT_MARA, [10](#)
findIT_regionRP, [11](#)
findIT_TFHit, [13](#)
findIT_TTPair, [14](#)

getAssocPairNumber, [15](#)

integrate_ChIP_RNA, [16](#)
integrate_replicates, [17](#)

jaccard_findIT_enrichFisher, [18](#)
jaccard_findIT_TTpair, [19](#)

loadPeakFile, [20](#)

mm_geneBound, [21](#)
mm_geneScan, [22](#)
mm_nearestGene, [23](#)

peakGeneCor, [23](#)
plot_annoDistance, [25](#)
plot_peakGeneAlias_summary, [25](#)
plot_peakGeneCor, [26](#)