

# Package ‘ChIPseeker’

January 16, 2025

**Type** Package

**Title** ChIPseeker for ChIP peak Annotation, Comparison, and Visualization

**Version** 1.43.0

**Maintainer** Guangchuang Yu <guangchuangyu@gmail.com>

**Description** This package implements functions to retrieve the nearest genes around the peak, annotate genomic region of the peak, statistical methods for estimate the significance of overlap among ChIP peak data sets, and incorporate GEO database for user to compare the own dataset with those deposited in database. The comparison can be used to infer cooperative regulation and thus can be used to generate hypotheses. Several visualization functions are implemented to summarize the coverage of the peak experiment, average profile and heatmap of peaks binding to TSS regions, genomic annotation, distance to TSS, and overlap of peaks or genes.

**Depends** R (>= 3.5.0)

**Imports** AnnotationDbi, aplot, BiocGenerics, boot, dplyr, enrichplot, IRanges, GenomeInfoDb, GenomicRanges, GenomicFeatures, ggplot2, gplots, graphics, grDevices, gtools, magrittr, methods, plotrix, parallel, RColorBrewer, rlang, rtracklayer, S4Vectors, scales, stats, tibble, TxDb.Hsapiens.UCSC.hg19.knownGene, utils, yulab.utils (>= 0.1.5)

**Suggests** clusterProfiler, ggimage, ggplotify, ggupset, ggVennDiagram, knitr, org.Hs.eg.db, prettydoc, ReactomePA, rmarkdown, testthat

**Remotes** GuangchuangYu/enrichplot

**URL** <https://yulab-smu.top/contribution-knowledge-mining/>

**BugReports** <https://github.com/YuLab-SMU/ChIPseeker/issues>

**Encoding** UTF-8

**VignetteBuilder** knitr

**ByteCompile** true

**License** Artistic-2.0

**biocViews** Annotation, ChIPSeq, Software, Visualization,  
MultipleComparison

**RoxygenNote** 7.3.2

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

**git\_branch** devel

**git\_last\_commit** 9e54d40

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

**Repository** Bioconductor 3.21

**Date/Publication** 2025-01-16

**Author** Guangchuang Yu [aut, cre] (ORCID:

<<https://orcid.org/0000-0002-6485-8781>>),

Ming Li [ctb],

Qianwen Wang [ctb],

Yun Yan [ctb],

Hervé Pagès [ctb],

Michael Kluge [ctb],

Thomas Schwarzl [ctb],

Zhougeng Xu [ctb],

Chun-Hui Gao [ctb] (ORCID: <<https://orcid.org/0000-0002-1445-7939>>)

## Contents

ChIPseeker-package . . . . .	4
. . . . .	5
annotatePeak . . . . .	5
as.data.frame.csAnno . . . . .	7
as.GRanges . . . . .	8
check_upstream_and_downstream . . . . .	8
combine_csAnno . . . . .	9
covplot . . . . .	9
csAnno-class . . . . .	10
downloadGEObedFiles . . . . .	11
downloadGSMbedFiles . . . . .	11
dropAnno . . . . .	12
enrichAnnoOverlap . . . . .	12
enrichPeakOverlap . . . . .	13
getAnnoStat . . . . .	14
getBioRegion . . . . .	14
getGeneAnno . . . . .	15
getGenomicAnnotation . . . . .	16
getGEOgenomeVersion . . . . .	17
getGEOInfo . . . . .	17
getGEOspecies . . . . .	18
getNearestFeatureIndicesAndDistances . . . . .	18
getPromoters . . . . .	19

getSampleFiles . . . . .	20
getTagMatrix . . . . .	20
getTagMatrix.binning.internal . . . . .	22
getTagMatrix.internal . . . . .	23
getTagMatrix2 . . . . .	23
getTagMatrix2.binning.internal . . . . .	24
getTagMatrix2.internal . . . . .	25
info . . . . .	25
makeBioRegionFromGranges . . . . .	26
overlap . . . . .	27
peakHeatmap . . . . .	27
peakHeatmap_multiple_Sets . . . . .	29
peak_Profile_Heatmap . . . . .	30
plotAnnoBar . . . . .	31
plotAnnoBar.data.frame . . . . .	32
plotAnnoPie . . . . .	33
plotAnnoPie.csAnno . . . . .	34
plotAvgProf . . . . .	36
plotAvgProf.binning . . . . .	37
plotAvgProf2 . . . . .	38
plotDistToTSS . . . . .	39
plotDistToTSS.data.frame . . . . .	40
plotMultiProf . . . . .	41
plotMultiProf.binning . . . . .	42
plotMultiProf.binning.internal . . . . .	43
plotMultiProf.normal . . . . .	44
plotMultiProf.normal.internal . . . . .	45
plotPeakProf . . . . .	45
plotPeakProf2 . . . . .	48
plotPeakProf_MultiWindows . . . . .	50
readPeakFile . . . . .	52
reexports . . . . .	52
seq2gene . . . . .	53
show . . . . .	54
shuffle . . . . .	54
tagHeatmap . . . . .	55
upsetplot . . . . .	56
vennpie . . . . .	56
vennplot . . . . .	57
vennplot.peakfile . . . . .	58

ChIPseeker-package      *ChIPseeker: ChIPseeker for ChIP peak Annotation, Comparison, and Visualization*

---

## Description

This package implements functions to retrieve the nearest genes around the peak, annotate genomic region of the peak, statistical methods for estimate the significance of overlap among ChIP peak data sets, and incorporate GEO database for user to compare the own dataset with those deposited in database. The comparison can be used to infer cooperative regulation and thus can be used to generate hypotheses. Several visualization functions are implemented to summarize the coverage of the peak experiment, average profile and heatmap of peaks binding to TSS regions, genomic annotation, distance to TSS, and overlap of peaks or genes.

## Author(s)

**Maintainer:** Guangchuang Yu <guangchuangyu@gmail.com> ([ORCID](#))

Other contributors:

- Ming Li <limiang929@gmail.com> [contributor]
- Qianwen Wang <treywea@gmail.com> [contributor]
- Yun Yan <youryanyun@gmail.com> [contributor]
- Hervé Pagès <hpages.on.github@gmail.com> [contributor]
- Michael Kluge <michael.kluge@bio.ifi.lmu.de> [contributor]
- Thomas Schwarzl <schwarzl@embl.de> [contributor]
- Zhougeng Xu <xuzhougeng@163.com> [contributor]
- Chun-Hui Gao <gaospecial@gmail.com> ([ORCID](#)) [contributor]

## See Also

Useful links:

- <https://yulab-smu.top/contribution-knowledge-mining/>
- Report bugs at <https://github.com/YuLab-SMU/ChIPseeker/issues>

---

---

### Description

capture name of variable

### Usage

```
.(..., .env = parent.frame())
```

### Arguments

...	expression
.env	environment

### Value

expression

### Examples

```
x <- 1  
eval(.(x)[[1]])
```

---

annotatePeak	<i>annotatePeak</i>
--------------	---------------------

---

### Description

Annotate peaks

### Usage

```
annotatePeak(  
  peak,  
  tssRegion = c(-3000, 3000),  
  TxDb = NULL,  
  level = "transcript",  
  assignGenomicAnnotation = TRUE,  
  genomicAnnotationPriority = c("Promoter", "5UTR", "3UTR", "Exon", "Intron",  
    "Downstream", "Intergenic"),  
  annoDb = NULL,  
  addFlankGeneInfo = FALSE,  
  flankDistance = 5000,  
  sameStrand = FALSE,
```

```

ignoreOverlap = FALSE,
ignoreUpstream = FALSE,
ignoreDownstream = FALSE,
overlap = "TSS",
verbose = TRUE,
columns = c("ENTREZID", "ENSEMBL", "SYMBOL", "GENENAME")
)

```

### Arguments

peak	peak file or GRanges object
tssRegion	Region Range of TSS
TxDb	TxDb or EnsDb annotation object
level	one of transcript and gene
assignGenomicAnnotation	logical, assign peak genomic annotation or not
genomicAnnotationPriority	genomic annotation priority
annoDb	annotation package
addFlankGeneInfo	logical, add flanking gene information from the peaks
flankDistance	distance of flanking sequence
sameStrand	logical, whether find nearest/overlap gene in the same strand
ignoreOverlap	logical, whether ignore overlap of TSS with peak
ignoreUpstream	logical, if True only annotate gene at the 3' of the peak.
ignoreDownstream	logical, if True only annotate gene at the 5' of the peak.
overlap	one of 'TSS' or 'all', if overlap="all", then gene overlap with peak will be reported as nearest gene, no matter the overlap is at TSS region or not.
verbose	print message or not
columns	names of columns to be obtained from database

### Value

data.frame or GRanges object with columns of:

all columns provided by input.

annotation: genomic feature of the peak, for instance if the peak is located in 5'UTR, it will annotated by 5'UTR. Possible annotation is Promoter-TSS, Exon, 5' UTR, 3' UTR, Intron, and Inter-genic.

geneChr: Chromosome of the nearest gene

geneStart: gene start

geneEnd: gene end

geneLength: gene length

geneStrand: gene strand  
 geneId: entrezgene ID  
 distanceToTSS: distance from peak to gene TSS  
 if annoDb is provided, extra column will be included:  
 ENSEMBL: ensembl ID of the nearest gene  
 SYMBOL: gene symbol  
 GENENAME: full gene name

**Author(s)**

G Yu

**See Also**

[plotAnnoBar](#) [plotAnnoPie](#) [plotDistToTSS](#)

**Examples**

```
## Not run:
require(TxDb.Hsapiens.UCSC.hg19.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
peakfile <- system.file("extdata", "sample_peaks.txt", package="ChIPseeker")
peakAnno <- annotatePeak(peakfile, tssRegion=c(-3000, 3000), TxDb=txdb)
peakAnno

## End(Not run)
```

---

as.data.frame.csAnno    *as.data.frame.csAnno*

---

**Description**

convert csAnno object to data.frame

**Usage**

```
## S3 method for class 'csAnno'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

**Arguments**

x	csAnno object
row.names	row names
optional	should be omitted.
...	additional parameters

**Value**

data.frame

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

---

as.GRanges

*as.GRanges*

---

**Description**

convert csAnno object to GRanges

**Usage**

```
as.GRanges(x)
```

**Arguments**

x                   csAnno object

**Value**

GRanges object

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

---

check\_upstream\_and\_downstream

*check upstream and downstream parameter*

---

**Description**

check\_upstream\_and\_downstream

**Usage**

```
check_upstream_and_downstream(upstream, downstream)
```

**Arguments**

upstream           upstream

downstream        downstream



---

combine_csAnno	<i>combine_csAnno</i>
----------------	-----------------------

---

**Description**

Combine csAnno Object

**Usage**

```
combine_csAnno(x, ...)
```

**Arguments**

x	csAnno object
...	csAnno objects

**Details**

<https://github.com/YuLab-SMU/ChIPseeker/issues/157>

**Value**

csAnno object

---

covplot	<i>covplot</i>
---------	----------------

---

**Description**

plot peak coverage

**Usage**

```
covplot(  
  peak,  
  weightCol = NULL,  
  xlab = "Chromosome Size (bp)",  
  ylab = "",  
  title = "ChIP Peaks over Chromosomes",  
  chrs = NULL,  
  xlim = NULL,  
  lower = 1,  
  fill_color = "black"  
)
```

**Arguments**

peak	peak file or GRanges object
weightCol	weight column of peak
xlab	xlab
ylab	ylab
title	title
chrs	selected chromosomes to plot, all chromosomes by default
xlim	ranges to plot, default is whole chromosome
lower	lower cutoff of coverage signal
fill_color	specify the color/palette for the plot. Order matters

**Value**

ggplot2 object

**Author(s)**

G Yu

---

csAnno-class	<i>Class "csAnno" This class represents the output of ChIPseeker Annotation</i>
--------------	---

---

**Description**

Class "csAnno" This class represents the output of ChIPseeker Annotation

**Slots**

anno annotation  
 tssRegion TSS region  
 level transcript or gene  
 hasGenomicAnnotation logical  
 detailGenomicAnnotation Genomic Annotation in detail  
 annoStat annotation statistics  
 peakNum number of peaks

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

**See Also**

[annotatePeak](#)

---

downloadGEObedFiles    *downloadGEObedFiles*

---

**Description**

download all BED files of a particular genome version

**Usage**

```
downloadGEObedFiles(genome, destDir = getwd())
```

**Arguments**

genome	genome version
destDir	destination folder

**Author(s)**

G Yu

---

downloadGSMbedFiles    *downloadGSMbedFiles*

---

**Description**

download BED supplementary files of a list of GSM accession numbers

**Usage**

```
downloadGSMbedFiles(GSM, destDir = getwd())
```

**Arguments**

GSM	GSM accession numbers
destDir	destination folder

**Author(s)**

G Yu

dropAnno                    *dropAnno*

---

**Description**

dropAnno

**Usage**

```
dropAnno(csAnno, distanceToTSS_cutoff = 10000)
```

**Arguments**

csAnno                    output of annotatePeak  
distanceToTSS\_cutoff    distance to TSS cutoff

**Details**

drop annotation exceeding distanceToTSS\_cutoff

**Value**

csAnno object

**Author(s)**

Guangchuang Yu

---

enrichAnnoOverlap        *enrichAnnoOverlap*

---

**Description**

calculate overlap significant of CHIP experiments based on their nearest gene annotation

**Usage**

```
enrichAnnoOverlap(  
  queryPeak,  
  targetPeak,  
  TxDb = NULL,  
  pAdjustMethod = "BH",  
  chainFile = NULL,  
  distanceToTSS_cutoff = NULL  
)
```

**Arguments**

queryPeak	query bed file
targetPeak	target bed file(s) or folder containing bed files
TxDb	TxDb
pAdjustMethod	pvalue adjustment method
chainFile	chain file for liftOver
distanceToTSS_cutoff	restrict nearest gene annotation by distance cutoff

**Value**

data.frame

**Author(s)**

G Yu

---

enrichPeakOverlap      *enrichPeakOverlap*

---

**Description**

calculate overlap significant of ChIP experiments based on the genome coordinations

**Usage**

```
enrichPeakOverlap(
  queryPeak,
  targetPeak,
  TxDb = NULL,
  pAdjustMethod = "BH",
  nShuffle = 1000,
  chainFile = NULL,
  pool = TRUE,
  mc.cores = detectCores() - 1,
  verbose = TRUE
)
```

**Arguments**

queryPeak	query bed file or GRanges object
targetPeak	target bed file(s) or folder that containing bed files or a list of GRanges objects
TxDb	TxDb
pAdjustMethod	pvalue adjustment method

nShuffle	shuffle numbers
chainFile	chain file for liftOver
pool	logical, whether pool target peaks
mc.cores	number of cores, see <a href="#">mclapply</a>
verbose	logical

**Value**

data.frame

**Author(s)**

G Yu

---

getAnnoStat	<i>getAnnoStat</i>
-------------	--------------------

---

**Description**

getting status of annotation

**Usage**

```
getAnnoStat(x)
```

**Arguments**

x                   csAnno object

---

getBioRegion	<i>getBioRegion</i>
--------------	---------------------

---

**Description**

prepare a bioregion of selected feature

**Usage**

```
getBioRegion(
  TxDb = NULL,
  upstream = 1000,
  downstream = 1000,
  by = "gene",
  type = "start_site"
)
```

**Arguments**

TxDB	TxDB
upstream	upstream from start site or end site
downstream	downstream from start site or end site
by	one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', 'UTR'
type	one of "start_site", "end_site", "body"

**Details**

this function combined previous functions getPromoters(), getBioRegion() and getGeneBody() in order to solve the following issues.

(1) <https://github.com/GuangchuangYu/ChIPseeker/issues/16>

(2) <https://github.com/GuangchuangYu/ChIPseeker/issues/87>

The getBioRegion() function can provide a region of interest from txdb object. There are three kinds of regions, start\_site, end\_site and body.

We take transcript region to explain the differences of these three regions. tx: chr1 1000 1400.

body region refers to the 1000-1400bp.

start\_site region with upstream = 100, downstream = 100 refers to 900-1100bp.

end\_site region with upstream = 100, downstream = 100 refers to 1300-1500bp.

**Value**

GRanges object

**Author(s)**

Guangchuang Yu, Ming L

---

getGeneAnno

*getGeneAnno*

---

**Description**

get gene annotation, symbol, gene name etc.

**Usage**

```
getGeneAnno(annoDb, geneID, type, columns)
```

**Arguments**

annoDb	annotation package
geneID	query geneID
type	gene ID type
columns	names of columns to be obtained from database

**Value**

data.frame

**Author(s)**

G Yu

---

getGenomicAnnotation *getGenomicAnnotation*

---

**Description**

get Genomic Annotation of peaks

**Usage**

```
getGenomicAnnotation(  
  peaks,  
  distance,  
  tssRegion = c(-3000, 3000),  
  TxDb,  
  level,  
  genomicAnnotationPriority,  
  sameStrand = FALSE  
)
```

**Arguments**

peaks	peaks in GRanges object
distance	distance of peak to TSS
tssRegion	tssRegion, default is -3kb to +3kb
TxDb	TxDb object
level	one of gene or transcript
genomicAnnotationPriority	genomic Annotation Priority
sameStrand	whether annotate gene in same strand

**Value**

character vector

**Author(s)**

G Yu



---

`getGEOgenomeVersion`     *getGEOgenomeVersion*

---

**Description**

get genome version statistics collecting from GEO ChIPseq data

**Usage**

```
getGEOgenomeVersion()
```

**Value**

data.frame

**Author(s)**

G Yu

---

`getGEOInfo`                     *getGEOInfo*

---

**Description**

get subset of GEO information by genome version keyword

**Usage**

```
getGEOInfo(genome, simplify = TRUE)
```

**Arguments**

<code>genome</code>	genome version
<code>simplify</code>	simplify result or not

**Value**

data.frame

**Author(s)**

G Yu

---

<code>getGEOspecies</code>	<i>getGEOspecies</i>
----------------------------	----------------------

---

**Description**

accessing species statistics collecting from GEO database

**Usage**

```
getGEOspecies()
```

**Value**

data.frame

**Author(s)**

G Yu

---

<code>getNearestFeatureIndicesAndDistances</code>	<i>getNearestFeatureIndicesAndDistances</i>
---	---

---

**Description**

get index of features that closest to peak and calculate distance

**Usage**

```
getNearestFeatureIndicesAndDistances(  
  peaks,  
  features,  
  sameStrand = FALSE,  
  ignoreOverlap = FALSE,  
  ignoreUpstream = FALSE,  
  ignoreDownstream = FALSE,  
  overlap = "TSS"  
)
```

**Arguments**

peaks	peak in GRanges
features	features in GRanges
sameStrand	logical, whether find nearest gene in the same strand
ignoreOverlap	logical, whether ignore overlap of TSS with peak
ignoreUpstream	logical, if True only annotate gene at the 3' of the peak.
ignoreDownstream	logical, if True only annotate gene at the 5' of the peak.
overlap	one of "TSS" or "all"

**Value**

list

**Author(s)**

G Yu

---

getPromoters	<i>getPromoters</i>
--------------	---------------------

---

**Description**

prepare the promoter regions

**Usage**

```
getPromoters(TxDb = NULL, upstream = 1000, downstream = 1000, by = "gene")
```

**Arguments**

TxDb	TxDb
upstream	upstream from TSS site
downstream	downstream from TSS site
by	one of gene or transcript

**Value**

GRanges object

---

getSampleFiles      *getSampleFiles*

---

**Description**

get filenames of sample files

**Usage**

```
getSampleFiles()
```

**Value**

list of file names

**Author(s)**

G Yu

---

getTagMatrix      *getTagMatrix*

---

**Description**

calculate the tag matrix

**Usage**

```
getTagMatrix(  
  peak,  
  upstream,  
  downstream,  
  windows,  
  type,  
  by,  
  TxDb = NULL,  
  weightCol = NULL,  
  nbin = NULL,  
  verbose = TRUE,  
  ignore_strand = FALSE  
)
```

**Arguments**

peak	peak peak file or GRanges object
upstream	the distance of upstream extension
downstream	the distance of downstream extension
windows	a collection of region
type	one of "start_site", "end_site", "body"
by	one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', or specified by users
TxDb	TxDb or self-made granges object, served as txdb
weightCol	column name of weight, default is NULL
nbin	the amount of nbins
verbose	print message or not
ignore_strand	ignore the strand information or not

**Details**

getTagMatrix() function can produce the matrix for visualization. peak stands for the peak file. window stands for a collection of regions that users want to look into. Users can use window to capture the peak of interest. There are two ways to input window.

The first way is that users can use getPromoters()/getBioRegion()/makeBioRegionFromGranges() to get window and put it into getTagMatrix().

The second way is that users can use getTagMatrix() to call getPromoters()/getBioRegion()/makeBioRegionFromGranges(). In this way users do not need to input window parameter but they need to input txdb.

txdb is a set of packages contained annotation of regions of different genomes. Users can get the regions of interest through specific functions. These specific functions are built in getPromoters()/getBioRegion(). Many regions can not be gain through txdb, like insulator and enhancer regions. Users can provide these regions in the form of granges object. These self-made granges object will be passed to TxDb parameter and they will be passed to makeBioRegionFromGranges() to produce the window. In a word, TxDb parameter is a reference information. Users can pass txdb object or self-made granges into it.

Details see [getPromoters](#), [getBioRegion](#) and [makeBioRegionFromGranges](#)

upstream and downstream parameter have different usages:

(1) window parameter is provided,

if type == 'body', upstream and downstream can use to extend the flank of body region.

if type == 'start\_site'/'end\_site', upstream and downstream do not play a role in getTagMatrix() function.

(2) window parameter is missing,

if type == 'body', upstream and downstream can use to extend the flank of body region.

if type == 'start\_site'/'end\_site', upstream and downstream refer to the upstream and downstream of the start\_site or the end\_site.

weightCol refers to column in peak file. This column acts as a weight vaule. Details see <https://github.com/YuLab-SMU/ChIPseeker/issues/15>

nbin refers to the number of bins. getTagMatrix() provide a binning method to get the tag matrix.

**Value**

tagMatrix

---

```
getTagMatrix.binning.internal
    getTagMatrix.binning.internal
```

---

**Description**

calculate the tagMatrix by binning the idea was derived from the function of deeptools <https://deeptools.readthedocs.io/en/dev>

**Usage**

```
getTagMatrix.binning.internal(
    peak,
    weightCol = NULL,
    windows,
    nbin = 800,
    upstream = NULL,
    downstream = NULL,
    ignore_strand = FALSE
)
```

**Arguments**

peak	peak peak file or GRanges object
weightCol	weightCol column name of weight, default is NULL
windows	windows a collection of region with equal or not equal size, eg. promoter region, gene region.
nbin	the amount of nbines needed to be splited and it should not be more than min_body_length
upstream	rel object, NULL or actual number
downstream	rel object, NULL or actual number
ignore_strand	ignore the strand information or not

**Value**

tagMatrix

---

`getTagMatrix.internal` *getTagMatrix.internal*

---

**Description**

calculate the tag matrix

**Usage**

```
getTagMatrix.internal(peak, weightCol = NULL, windows, ignore_strand = FALSE)
```

**Arguments**

peak	peak file or GRanges object
weightCol	column name of weight, default is NULL
windows	a collection of region with equal size, eg. promoter region.
ignore_strand	ignore the strand information or not

**Value**

tagMatrix

**Author(s)**

G Yu

---

`getTagMatrix2` *getTagMatrix2*

---

**Description**

Nested function for `getTagMatrix()` to deal with multiple windows

**Usage**

```
getTagMatrix2(  
  peak,  
  upstream,  
  downstream,  
  windows_name,  
  type,  
  by,  
  TxDb = NULL,  
  weightCol = NULL,  
  nbin = NULL,
```

```

    verbose = TRUE,
    ignore_strand = FALSE
  )

```

### Arguments

peak	peak peak file or GRanges object
upstream	the distance of upstream extension
downstream	the distance of downstream extension
windows_name	the names of windows
type	one of "start_site", "end_site", "body"
by	one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', or specified by users
TxDb	TxDb or self-made granges object, served as txdb
weightCol	column name of weight, default is NULL
nbin	the amount of nbines
verbose	print message or not
ignore_strand	ignore the strand information or not

### Details

This is an internal function.

### Value

tagMatrix

---

```

getTagMatrix2.binning.internal
      internal function

```

---

### Description

internal function

### Usage

```

getTagMatrix2.binning.internal(
  peak,
  weightCol = NULL,
  windows,
  windows_name,
  nbin = 800,
  upstream = NULL,
  downstream = NULL,
  ignore_strand = FALSE
)

```



**Arguments**

peak	peak peak file or GRanges object
weightCol	column name of weight, default is NULL
windows	a collection of region
windows_name	the name of windows
nbin	the amount of nbins
upstream	the distance of upstream extension
downstream	the distance of downstream extension
ignore_strand	ignore the strand information or not

---

getTagMatrix2.internal

*getTagMatrix2.internal*

---

**Description**

getTagMatrix2.internal

**Usage**

```
getTagMatrix2.internal(
  peak,
  weightCol = NULL,
  windows,
  windows_name,
  ignore_strand = FALSE
)
```

**Arguments**

peak	peak peak file or GRanges object
weightCol	column name of weight, default is NULL
windows	a collection of region
windows_name	the name of windows
ignore_strand	ignore the strand information or not

---

info

*Information Datasets*

---

**Description**

ucsc genome version, precalcuated data and gsm information

---

`makeBioRegionFromGranges`*makeBioRegionFromGranges*

---

## Description

make windows from granges object

## Usage

```
makeBioRegionFromGranges(gr, by, type, upstream = 1000, downstream = 1000)
```

## Arguments

<code>gr</code>	a grange object contain region of interest
<code>by</code>	specify be users, e.g. gene, insulator, enhancer
<code>type</code>	one of "start_site", "end_site", "body"
<code>upstream</code>	upstream from start site or end site, can be NULL if the type == 'body'
<code>downstream</code>	downstream from start site or end site, can be NULL if the type == 'body'

## Details

`makeBioRegionFromGranges()` function can make bioregion from granges object.

The differences between `makeBioRegionFromGranges()` and `getBioRegion()` is that `getBioRegion()` get the region object from `txdb` object but `makeBioRegionFromGranges()` get the region from the granges object provided by users. For example, `txdb` object do not contain insulator or enhancer regions. Users can provide these regions through self-made granges object

There are three kinds of regions, `start_site`, `end_site` and `body`.

We take enhancer region to explain the differences of these three regions. enhancer: chr1 1000 1400.

body region refers to the 1000-1400bp.

`start_site` region with `upstream = 100`, `downstream = 100` refers to 900-1100bp.

`end_site` region with `upstream = 100`, `downstream = 100` refers to 1300-1500bp.

In `makeBioRegionFromGranges()`, `upstream` and `downstream` can be NULL if the type == 'body'. `by` should be specified by users and can not be omitted. `by` parameter will be used to made labels. `type` should also be specified.

<https://github.com/YuLab-SMU/ChIPseeker/issues/189>

## Value

GRanges object

---

overlap	<i>overlap</i>
---------	----------------

---

**Description**

calculate the overlap matrix, which is useful for vennplot

**Usage**

```
overlap(Sets)
```

**Arguments**

Sets            a list of objects

**Value**

data.frame

**Author(s)**

G Yu

---

peakHeatmap	<i>peakHeatmap</i>
-------------	--------------------

---

**Description**

plot the heatmap of peaks

**Usage**

```
peakHeatmap(  
  peak,  
  weightCol = NULL,  
  TxDb = NULL,  
  upstream = 1000,  
  downstream = 1000,  
  xlab = "",  
  ylab = "",  
  title = NULL,  
  palette = NULL,  
  verbose = TRUE,  
  by = "gene",  
  type = "start_site",  
  nbin = NULL,  
)
```

```

    ignore_strand = FALSE,
    windows,
    ncol = NULL,
    nrow = NULL
)

```

### Arguments

peak	peak file or GRanges object
weightCol	column name of weight
TxDb	TxDb object
upstream	upstream position
downstream	downstream position
xlab	xlab
ylab	ylab
title	title
palette	palette to be filled in,details see <a href="#">scale_colour_brewer</a>
verbose	print message or not
by	one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', 'UTR'
type	one of "start_site", "end_site", "body"
nbin	the amount of nbines
ignore_strand	ignore the strand information or not
windows	a collection of region
ncol	the ncol of plotting a list of peak
nrow	the nrow of plotting a list of peak

### Value

figure

### Author(s)

G Yu

---

```
peakHeatmap_multiple_Sets
      peakHeatmap
```

---

## Description

plot the heatmap of peaks align to a sets of regions

## Usage

```
peakHeatmap_multiple_Sets(
  peak,
  weightCol = NULL,
  TxDb = NULL,
  upstream = 1000,
  downstream = 1000,
  xlab = "",
  ylab = "",
  title = NULL,
  palette = NULL,
  verbose = TRUE,
  by = "gene",
  type = "start_site",
  nbin = NULL,
  ignore_strand = FALSE,
  windows_name = NULL,
  ncol = NULL,
  nrow = NULL,
  facet_label_text_size = 12
)
```

## Arguments

peak	peak file or GRanges object
weightCol	column name of weight
TxDb	TxDb object
upstream	upstream position
downstream	downstream position
xlab	xlab
ylab	ylab
title	title
palette	palette to be filled in,details see <a href="#">scale_colour_brewer</a>
verbose	print message or not
by	one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', 'UTR'

type	one of "start_site", "end_site", "body"
nbin	the amount of nbines
ignore_strand	ignore the strand information or not
windows_name	the name for each window, which will also be showed in the picture as labels
ncol	the ncol of plotting a list of peak
nrow	the nrow of plotting a list of peak
facet_label_text_size	the size of facet label text

**Value**

figure

---

peak\_Profile\_Heatmap *peak\_Profile\_Heatmap*

---

**Description**

plot peak heatmap and profile in a picture

**Usage**

```
peak_Profile_Heatmap(
  peak,
  weightCol = NULL,
  TxDb = NULL,
  upstream = 1000,
  downstream = 1000,
  xlab = "",
  ylab = "",
  title = NULL,
  palette = NULL,
  verbose = TRUE,
  by = "gene",
  type = "start_site",
  nbin = NULL,
  ignore_strand = FALSE,
  windows_name = NULL,
  ncol = NULL,
  nrow = NULL,
  facet_label_text_size = 12,
  conf,
  facet = "row",
  free_y = TRUE,
  height_proportion = 4
)
```

**Arguments**

peak	peak file or GRanges object
weightCol	column name of weight
TxDb	TxDb object
upstream	upstream position
downstream	downstream position
xlab	xlab
ylab	ylab
title	title
palette	palette to be filled in,details see <a href="#">scale_colour_brewer</a>
verbose	print message or not
by	one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', 'UTR'
type	one of "start_site", "end_site", "body"
nbin	the amount of nbines
ignore_strand	ignore the strand information or not
windows_name	the name for each window, which will also be showed in the picture as labels
ncol	the ncol of plotting a list of peak
nrow	the nrow of plotting a list of peak
facet_label_text_size	the size of facet label text
conf	confidence interval
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
height_proportion	the proportion of profiling picture and heatmap

---

plotAnnoBar

*plotAnnoBar method generics*


---

**Description**

plotAnnoBar method for csAnno instance

**Usage**

```

plotAnnoBar(
  x,
  xlab = "",
  ylab = "Percentage(%)",
  title = "Feature Distribution",
  ...
)

## S4 method for signature 'list'
plotAnnoBar(
  x,
  xlab = "",
  ylab = "Percentage(%)",
  title = "Feature Distribution",
  ...
)

plotAnnoBar(x, xlab="", ylab='Percentage(%)',title="Feature Distribution", ...)

```

**Arguments**

<code>x</code>	csAnno instance
<code>xlab</code>	xlab
<code>ylab</code>	ylab
<code>title</code>	title
<code>...</code>	additional paramter

**Value**

plot

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

---

plotAnnoBar.data.frame

*plotAnnoBar.data.frame*

---

**Description**

plot feature distribution based on their chromosome region



**Usage**

```
plotAnnoBar.data.frame(  
  anno.df,  
  xlab = "",  
  ylab = "Percentage%",  
  title = "Feature Distribution",  
  categoryColumn  
)
```

**Arguments**

anno.df	annotation stats
xlab	xlab
ylab	ylab
title	plot title
categoryColumn	category column

**Details**

plot chromosome region features

**Value**

bar plot that summarize genomic features of peaks

**Author(s)**

Guangchuang Yu <https://yulab-smu.top>

**See Also**

[annotatePeak](#) [plotAnnoPie](#)

---

plotAnnoPie

*plotAnnoPie method generics*

---

**Description**

plotAnnoPie method for csAnno instance

**Usage**

```
plotAnnoPie(  
  x,  
  ndigit = 2,  
  cex = 0.9,  
  col = NA,  
  legend.position = "rightside",  
  pie3D = FALSE,  
  radius = 0.8,  
  ...  
)
```

```
plotAnnoPie(x,ndigit=2,cex=0.9,col=NA,legend.position="rightside",pie3D=FALSE,radius=0.8,...)
```

**Arguments**

x	csAnno instance
ndigit	number of digit to round
cex	label cex
col	color
legend.position	topright or other.
pie3D	plot in 3D or not
radius	radius of the pie
...	extra parameter

**Value**

plot

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

---

plotAnnoPie.csAnno     *plotAnnoPie*

---

**Description**

pieplot from peak genomic annotation

**Usage**

```
plotAnnoPie.csAnno(  
  x,  
  ndigit = 2,  
  cex = 0.8,  
  col = NA,  
  legend.position = "rightside",  
  pie3D = FALSE,  
  radius = 0.8,  
  ...  
)
```

**Arguments**

x	csAnno object
ndigit	number of digit to round
cex	label cex
col	color
legend.position	topright or other.
pie3D	plot in 3D or not
radius	radius of Pie
...	extra parameter

**Value**

pie plot of peak genomic feature annotation

**Author(s)**

Guangchuang Yu <https://yulab-smu.top>

**See Also**

[annotatePeak](#) [plotAnnoBar](#)

**Examples**

```
## Not run:  
require(TxDb.Hsapiens.UCSC.hg19.knownGene)  
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene  
peakfile <- system.file("extdata", "sample_peaks.txt", package="chipseeker")  
peakAnno <- annotatePeak(peakfile, TxDb=txdb)  
plotAnnoPie(peakAnno)  
  
## End(Not run)
```

---

`plotAvgProf`*plotAvgProf*

---

**Description**

plot the profile of peaks

**Usage**

```
plotAvgProf(  
  tagMatrix,  
  xlim,  
  xlab = "Genomic Region (5'→3')",  
  ylab = "Peak Count Frequency",  
  conf,  
  facet = "none",  
  free_y = TRUE,  
  origin_label = "TSS",  
  verbose = TRUE,  
  ...  
)
```

**Arguments**

<code>tagMatrix</code>	tagMatrix or a list of tagMatrix
<code>xlim</code>	xlim
<code>xlab</code>	x label
<code>ylab</code>	y label
<code>conf</code>	confidence interval
<code>facet</code>	one of 'none', 'row' and 'column'
<code>free_y</code>	if TRUE, y will be scaled by AvgProf
<code>origin_label</code>	label of the center
<code>verbose</code>	print message or not
<code>...</code>	additional parameter

**Value**

ggplot object

**Author(s)**

G Yu; Y Yan

---

plotAvgProf.binning    *plotAvgProf.binning*

---

## Description

plot the profile of peaks by binning

## Usage

```
plotAvgProf.binning(  
  tagMatrix,  
  xlab = "Genomic Region (5'->3')",  
  ylab = "Peak Count Frequency",  
  conf,  
  facet = "none",  
  free_y = TRUE,  
  upstream = NULL,  
  downstream = NULL,  
  label,  
  ...  
)
```

## Arguments

tagMatrix	tagMatrix or a list of tagMatrix
xlab	x label
ylab	y label
conf	confidence interval
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled
upstream	rel object reflects the percentage of flank extension, e.g rel(0.2) integer reflects the actual length of flank extension or TSS region NULL reflects the gene body with no extension
downstream	rel object reflects the percentage of flank extension, e.g rel(0.2) integer reflects the actual length of flank extension or TSS region NULL reflects the gene body with no extension
label	label
...	additional parameter

## Value

ggplot object

---

plotAvgProf2

*plotAvgProf*


---

### Description

plot the profile of peaks that align to flank sequences of TSS

### Usage

```
plotAvgProf2(
  peak,
  weightCol = NULL,
  TxDb = NULL,
  upstream = 1000,
  downstream = 1000,
  xlab = "Genomic Region (5'->3')",
  ylab = "Peak Count Frequency",
  conf,
  facet = "none",
  free_y = TRUE,
  verbose = TRUE,
  ignore_strand = FALSE,
  ...
)
```

### Arguments

peak	peak file or GRanges object
weightCol	column name of weight
TxDb	TxDb object
upstream	upstream position
downstream	downstream position
xlab	xlab
ylab	ylab
conf	confidence interval
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
verbose	print message or not
ignore_strand	ignore the strand information or not
...	additional parameter

### Details

This function is the old function of plotPeakProf2. It can only plot the start site region of gene.

**Value**

ggplot object

**Author(s)**

G Yu, Ming L

plotDistToTSS

*plotDistToTSS method generics***Description**

plotDistToTSS method for csAnno instance

**Usage**

```

plotDistToTSS(
  x,
  distanceColumn = "distanceToTSS",
  xlab = "",
  ylab = "Binding sites (%) (5'→3')",
  title = "Distribution of transcription factor-binding loci relative to TSS",
  ...
)

## S4 method for signature 'list'
plotDistToTSS(
  x,
  distanceColumn = "distanceToTSS",
  xlab = "",
  ylab = "Binding sites (%) (5'→3')",
  title = "Distribution of transcription factor-binding loci relative to TSS",
  distanceBreaks = c(0, 1000, 3000, 5000, 10000, 1e+05),
  palette = NULL,
  ...
)

plotDistToTSS(x,distanceColumn="distanceToTSS", xlab="",
ylab="Binding sites (%) (5'→3')",
title="Distribution of transcription factor-binding loci relative to TSS",...)

```

**Arguments**

x	csAnno instance
distanceColumn	distance column name
xlab	xlab

<code>ylab</code>	<code>ylab</code>
<code>title</code>	<code>title</code>
<code>...</code>	additional parameter
<code>distanceBreaks</code>	breaks of distance, default is <code>'c(0, 1000, 3000, 5000, 10000, 100000)'</code>
<code>palette</code>	palette name for coloring different distances. Run <code>'RColorBrewer::display.brewer.all()'</code> to see all applicable values.

**Value**

plot

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

---

`plotDistToTSS.data.frame`

*plotDistToTSS.data.frame*

---

**Description**

plot feature distribution based on the distances to the TSS

**Usage**

```
plotDistToTSS.data.frame(
  peakDist,
  distanceColumn = "distanceToTSS",
  distanceBreaks = c(0, 1000, 3000, 5000, 10000, 1e+05),
  palette = NULL,
  xlab = "",
  ylab = "Binding sites (%) (5'->3')",
  title = "Distribution of transcription factor-binding loci relative to TSS",
  categoryColumn = ".id"
)
```

**Arguments**

<code>peakDist</code>	peak annotation
<code>distanceColumn</code>	column name of the distance from peak to nearest gene
<code>distanceBreaks</code>	default is <code>'c(0, 1000, 3000, 5000, 10000, 100000)'</code>
<code>palette</code>	palette name for coloring different distances. Run <code>'RColorBrewer::display.brewer.all()'</code> to see all applicable values.
<code>xlab</code>	x label
<code>ylab</code>	y lable
<code>title</code>	figure title
<code>categoryColumn</code>	category column, default is <code>".id"</code>



**Value**

bar plot that summarize distance from peak to TSS of the nearest gene.

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

**See Also**

[annotatePeak](#)

**Examples**

```
## Not run:
require(TxDb.Hsapiens.UCSC.hg19.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
peakfile <- system.file("extdata", "sample_peaks.txt", package="ChIPseeker")
peakAnno <- annotatePeak(peakfile, TxDb=txdb)
plotDistToTSS(peakAnno)

## End(Not run)
```

---

plotMultiProf

*internal function for plotPeakProf\_MultiWindows*

---

**Description**

internal function for plotPeakProf\_MultiWindows

**Usage**

```
plotMultiProf(
  tagMatrix,
  conf,
  xlab = "Genomic Region (5'→3')",
  ylab = "Peak Count Frequency",
  facet = "none",
  free_y = TRUE,
  ...
)
```

**Arguments**

tagMatrix	tagMatrix
conf	confidence interval
xlab	xlab
ylab	ylab

facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
...	additional parameter

---

plotMultiProf.binning *internal function*

---

### Description

internal function

### Usage

```
plotMultiProf.binning(
  tagMatrix,
  xlab = "Genomic Region (5'->3')",
  ylab = "Peak Count Frequency",
  conf,
  facet = "none",
  free_y = TRUE,
  upstream = NULL,
  downstream = NULL,
  label,
  ...
)
```

### Arguments

tagMatrix	tagMatrix
xlab	xlab
ylab	ylab
conf	confidence interval
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
upstream	the upstream extension
downstream	the downstream extension
label	the label of the center
...	additional parameter

---

plotMultiProf.binning.internal  
*internal function*

---

## Description

internal function

## Usage

```
plotMultiProf.binning.internal(  
  tagMatrix,  
  conf,  
  xlab = "Genomic Region (5'->3')",  
  ylab = "Peak Count Frequency",  
  facet = "none",  
  free_y = TRUE,  
  upstream = NULL,  
  downstream = NULL,  
  label,  
  ...  
)
```

## Arguments

tagMatrix	tagMatrix
conf	confidence interval
xlab	xlab
ylab	ylab
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
upstream	the upstream extension
downstream	the downstream extension
label	the label of the center
...	additional parameter

---

plotMultiProf.normal *internal function*

---

## Description

internal function

## Usage

```
plotMultiProf.normal(  
  tagMatrix,  
  xlim,  
  xlab = "Genomic Region (5'->3')",  
  ylab = "Peak Count Frequency",  
  conf,  
  facet = "none",  
  free_y = TRUE,  
  origin_label = "TSS",  
  verbose = TRUE,  
  ...  
)
```

## Arguments

tagMatrix	tagMatrix
xlim	xlim
xlab	xlab
ylab	ylab
conf	confidence interval
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
origin_label	the label of the center
verbose	print message or not
...	additional parameter

---

```
plotMultiProf.normal.internal
      internal function
```

---

**Description**

internal function

**Usage**

```
plotMultiProf.normal.internal(
  tagMatrix,
  conf,
  xlim = c(-3000, 3000),
  xlab = "Genomic Region (5'→3')",
  ylab = "Peak Count Frequency",
  facet = "row",
  free_y = TRUE,
  origin_label,
  ...
)
```

**Arguments**

tagMatrix	tagMatrix
conf	confidence interval
xlim	xlim
xlab	xlab
ylab	ylab
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
origin_label	the label of the center
...	additional parameter

---

```
plotPeakProf      plotPeakProf_MultiWindows
```

---

**Description**

plot the profile of peaks 'plotPeakProf\_MultiWindows()' is almost the same as plotPeakProf2(), having the main difference of accepting two or more granges objects. Accepting more granges objects can help compare the same peaks in different windows.

**Usage**

```

plotPeakProf(
  tagMatrix = NULL,
  peak,
  upstream,
  downstream,
  conf,
  by,
  type,
  windows_name = NULL,
  weightCol = NULL,
  TxDb = NULL,
  xlab = "Genomic Region (5'→3')",
  ylab = "Peak Count Frequency",
  facet = "row",
  free_y = TRUE,
  verbose = TRUE,
  nbin = NULL,
  ignore_strand = FALSE,
  ...
)

```

**Arguments**

tagMatrix	tagMatrix or a list of tagMatrix
peak	peak file or GRanges object
upstream	upstream position
downstream	downstream position
conf	confidence interval
by	feature of interest
type	one of "start_site", "end_site", "body"
windows_name	the name for each window, which will also be showed in the picture as labels
weightCol	column name of weight
TxDb	TxDb object or self-made granges objects
xlab	xlab
ylab	ylab
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
verbose	print message or not
nbin	the amount of bins
ignore_strand	ignore the strand information or not
...	additional parameter

## Details

TxDB parameter can accept txdb object. But many regions can not be obtained by txdb object. In this case, Users can provide self-made granges served the same role as txdb object and pass to TxDb object.

by the features of interest.

(1) if users use txdb, by can be one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', 'UTR'. These features can be obtained by functions from txdb object.

(2) if users use self-made granges object, by can be everything. Because this by will not pass to functions to get features, which is different from the case of using txdb object. This by is only used to made labels showed in picture.

type means the property of the region. one of the "start site", "end site" and "body".

upstream and downstream parameter have different usages:

(1) if type == 'body', upstream and downstream can use to extend the flank of body region.

(2) if type == 'start\_site'/'end\_site', upstream and downstream refer to the upstream and downstream of the start\_site or the end\_site.

weightCol refers to column in peak file. This column acts as a weight value. Details see <https://github.com/YuLab-SMU/ChIPseeker/issues/15>

nbin refers to the number of bins. getTagMatrix() provide a binning method to get the tag matrix.

There are two ways input a list of window.

(1) Users can input a list of self-made granges objects

(2) Users can input a list of by and only one type. In this way, plotPeakProf\_MultiWindows() can made a list of window from txdb object based on by and type.

Warning:

(1) All of these window should be the same type. It means users can only compare a list of "start site"/"end site"/"body region" with the same upstream and downstream.

(2) So it will be only one type and several by.

(3) Users can make window by txdb object or self-made granges object. Users can only choose one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR' or 'UTR' in the way of using txdb object. User can input any by in the way of using self-made granges object.

(4) Users can mingle the by designed for the two ways. plotPeakProf\_MultiWindows can accept the hybrid by. But the above rules should be followed.

<https://github.com/YuLab-SMU/ChIPseeker/issues/189>

## Value

ggplot object

---

plotPeakProf2

*plotPeakProf2*


---

## Description

plot the profile of peaks automatically

## Usage

```
plotPeakProf2(
  peak,
  upstream,
  downstream,
  conf,
  by,
  type,
  weightCol = NULL,
  TxDb = NULL,
  xlab = "Genomic Region (5'->3')",
  ylab = "Peak Count Frequency",
  facet = "none",
  free_y = TRUE,
  verbose = TRUE,
  nbin = NULL,
  ignore_strand = FALSE,
  ...
)
```

## Arguments

peak	peak file or GRanges object
upstream	upstream position
downstream	downstream position
conf	confidence interval
by	e.g. 'gene', 'transcript', 'exon' or features of interest(e.g. "enhancer")
type	one of "start_site", "end_site", "body"
weightCol	column name of weight
TxDb	TxDb object, or self-made granges object
xlab	xlab
ylab	ylab
facet	one of 'none', 'row' and 'column'
free_y	if TRUE, y will be scaled by AvgProf
verbose	print message or not



nbin	the amount of nbines
ignore_strand	ignore the strand information or not
...	additional parameter

## Details

peak stands for the peak file.

by the features of interest.

(1) if users use txdb, by can be one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', 'UTR'. These features can be obtained by functions from txdb object.

(2) if users use self-made granges object, by can be everything. Because this by will not pass to functions to get features, which is different from the case of using txdb object. This by is only used to made labels showed in picture.

type means the property of the region. one of the "start site", "end site" and "body".

upstream and downstream parameter have different usages:

(1) if type == 'body', upstream and downstream can use to extend the flank of body region.

(2) if type == 'start\_site'/'end\_site', upstream and downstream refer to the upstream and downstream of the start\_site or the end\_site.

weightCol refers to column in peak file. This column acts as a weight vaule. Details see <https://github.com/YuLab-SMU/ChIPseeker/issues/15>

nbin refers to the number of bins, providing a binning method to get the tag matrix.

TxDB parameter can accept txdb object. But many regions can not be obtained by txdb object. In this case, Users can provide self-made granges served the same role as txdb object and pass to TxDb object.

plotPeakProf2() is different from the plotPeakProf(). plotPeakProf2() do not need to provide window parameter, which means plotPeakProf2() will call relevent functions to make window automatically.

## Value

ggplot object

## Author(s)

G Yu, Ming Li

---

```
plotPeakProf_MultiWindows
      plotPeakProf_MultiWindows
```

---

### Description

plot the profile of peaks in two or more windows

### Usage

```
plotPeakProf_MultiWindows(
  peak,
  upstream,
  downstream,
  conf,
  by,
  type,
  windows_name = NULL,
  weightCol = NULL,
  TxDb = NULL,
  xlab = "Genomic Region (5'→3')",
  ylab = "Peak Count Frequency",
  facet = "row",
  free_y = TRUE,
  verbose = TRUE,
  nbin = NULL,
  ignore_strand = FALSE,
  ...
)
```

### Arguments

peak	peak file or GRanges object
upstream	upstream position
downstream	downstream position
conf	confidence interval
by	feature of interest
type	one of "start_site", "end_site", "body"
windows_name	the name for each window, which will also be showed in the picture as labels
weightCol	column name of weight
TxDb	TxDb object or self-made granges objects
xlab	xlab
ylab	ylab
facet	one of 'none', 'row' and 'column'

free_y	if TRUE, y will be scaled by AvgProf
verbose	print message or not
nbin	the amount of bins
ignore_strand	ignore the strand information or not
...	additional parameter

## Details

This function comes from <https://github.com/YuLab-SMU/ChIPseeker/issues/189> 'plotPeakProf\_MultiWindows()' is almost the same as plotPeakProf2(), having the main difference of accepting two or more granges objects. Accepting more granges objects can help compare the same peaks in different windows.

Txdb parameter can accept txdb object. But many regions can not be obtained by txdb object. In this case, Users can provide self-made granges served the same role as txdb object and pass to TxDb object.

by the features of interest.

(1) if users use txdb, by can be one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR', 'UTR'. These features can be obtained by functions from txdb object.

(2) if users use self-made granges object, by can be everything. Because this by will not pass to functions to get features, which is different from the case of using txdb object. This by is only used to made labels showed in picture.

type means the property of the region. one of the "start site", "end site" and "body".

upstream and downstream parameter have different usages:

(1) if type == 'body', upstream and downstream can use to extend the flank of body region.

(2) if type == 'start\_site'/'end\_site', upstream and downstream refer to the upstream and downstream of the start\_site or the end\_site.

weightCol refers to column in peak file. This column acts as a weight value. Details see <https://github.com/YuLab-SMU/ChIPseeker/issues/15>

nbin refers to the number of bins. getTagMatrix() provide a binning method to get the tag matrix.

There are two ways input a list of window.

(1) Users can input a list of self-made granges objects

(2) Users can input a list of by and only one type. In this way, plotPeakProf\_MultiWindows() can made a list of window from txdb object based on by and type.

Warning:

(1) All of these window should be the same type. It means users can only compare a list of "start site"/"end site"/"body region" with the same upstream and downstream.

(2) So it will be only one type and several by.

(3) Users can make window by txdb object or self-made granges object. Users can only choose one of 'gene', 'transcript', 'exon', 'intron', '3UTR', '5UTR' or 'UTR' in the way of using txdb object. User can input any by in the way of using self-made granges object.

(4) Users can mingle the by designed for the two ways. plotPeakProf\_MultiWindows can accept the hybrid by. But the above rules should be followed.

**Value**

ggplot object

---

readPeakFile	<i>readPeakFile</i>
--------------	---------------------

---

**Description**

read peak file and store in data.frame or GRanges object

**Usage**

```
readPeakFile(peakfile, as = "GRanges", ...)
```

**Arguments**

peakfile	peak file
as	output format, one of GRanges or data.frame
...	additional parameter (pass to 'utils::read.delim()')

**Value**

peak information, in GRanges or data.frame object

**Author(s)**

G Yu

**Examples**

```
peakfile <- system.file("extdata", "sample_peaks.txt", package="ChIPseeker")
peak.gr <- readPeakFile(peakfile, as="GRanges")
peak.gr
```

---

reexports	<i>Objects exported from other packages</i>
-----------	---

---

**Description**

These objects are imported from other packages. Follow the links below to see their documentation.

**GenomicRanges** [GRangesList](#)

**ggplot2** [rel](#)

---

seq2gene	<i>seq2gene</i>
----------	-----------------

---

## Description

annotate genomic regions to genes in many-to-many mapping

## Usage

```
seq2gene(seq, tssRegion, flankDistance, TxDb, sameStrand = FALSE)
```

## Arguments

seq	genomic regions in GRanges object
tssRegion	TSS region
flankDistance	flanking search radius
TxDb	TranscriptDb object
sameStrand	logical whether find nearest/overlap gene in the same strand

## Details

This function associates genomic regions with coding genes in a many-to-many mapping. It first maps genomic regions to host genes (either located in exon or intron), proximal genes (located in promoter regions) and flanking genes (located in upstream and downstream within user specify distance).

## Value

gene vector

## Author(s)

Guangchuang Yu

## Examples

```
## Not run:
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
TxDb <- TxDb.Hsapiens.UCSC.hg19.knownGene
file <- getSampleFiles()[[1]] # a bed file
gr <- readPeakFile(file)
genes <- seq2gene(gr, tssRegion=c(-1000, 1000), flankDistance = 3000, TxDb)

## End(Not run)
```

show *show method*

---

**Description**

show method for csAnno instance

**Usage**

show(object)

**Arguments**

object            A csAnno instance

**Value**

message

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

---

shuffle *shuffle*

---

**Description**

shuffle the position of peak

**Usage**

shuffle(peak.gr, TxDb)

**Arguments**

peak.gr            GRanges object  
TxDb                TxDb

**Value**

GRanges object

**Author(s)**

G Yu

---

tagHeatmap	<i>tagHeatmap</i>
------------	-------------------

---

**Description**

plot the heatmap of tagMatrix

**Usage**

```
tagHeatmap(  
  tagMatrix,  
  xlab = "",  
  ylab = "",  
  title = NULL,  
  palette = "RdBu",  
  nrow = NULL,  
  ncol = NULL  
)
```

**Arguments**

tagMatrix	tagMatrix or a list of tagMatrix
xlab	xlab
ylab	ylab
title	title
palette	palette to be filled in,details see <a href="#">scale_colour_brewer</a>
nrow	the nrow of plotting a list of peak
ncol	the ncol of plotting a list of peak

**Value**

figure

**Author(s)**

G Yu

upsetplot

*upsetplot method*

---

**Description**

upsetplot method generics

**Usage**

upsetplot(x, ...)

**Arguments**

x	A csAnno instance
...	additional parameter

**Value**

plot

**Author(s)**Guangchuang Yu <https://guangchuangyu.github.io>

---

vennpie

*vennpie method generics*

---

**Description**

vennpie method generics

**Usage**

vennpie(x, r = 0.2, cex = 1.2, ...)

vennpie(x, r = 0.2, cex=1.2, ...)

**Arguments**

x	A csAnno instance
r	initial radius
cex	value to adjust legend
...	additional parameter



**Value**

plot

**Author(s)**

Guangchuang Yu <https://guangchuangyu.github.io>

---

vennplot

*vennplot*

---

**Description**

plot the overlap of a list of object

**Usage**

```
vennplot(Sets, by = "gplots", ...)
```

**Arguments**

Sets	a list of object, can be vector or GRanges object
by	one of gplots, ggVennDiagram or Vennerable
...	extra parameters using ggVennDiagram. Details see <a href="#">ggVennDiagram</a>

**Details**

There are two ways to plot, which users can specify through 'by'.

The first way is to use 'gplots' packages, by setting 'by = gplots'. This method is default method. The venn plot produced through this way has no color.

The second way is to use 'ggVennDiagram' packages, by setting 'by = ggVennDiagram'. The venn plot produced through this way has colors which can be defined by users using ggplot2 grammar e.g.(scale\_fill\_distiller()). And users can specify any details, like digital number, text size and showing percentage or not, by inputting '...' extra parameters.

**Value**

venn plot that summarize the overlap of peaks from different experiments or gene annotation from different peak files.

**Author(s)**

G Yu

**Examples**

```
## example not run
## require(TxDb.Hsapiens.UCSC.hg19.knownGene)
## txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
## peakfiles <- getSampleFiles()
## peakAnnoList <- lapply(peakfiles, annotatePeak)
## names(peakAnnoList) <- names(peakfiles)
## genes= lapply(peakAnnoList, function(i) as.data.frame(i)$geneId)
## vennplot(genes)
```

---

vennplot.peakfile      *vennplot.peakfile*

---

**Description**

vennplot for peak files

**Usage**

```
vennplot.peakfile(files, labels = NULL)
```

**Arguments**

files	peak files
labels	labels for peak files

**Value**

figure

**Author(s)**

G Yu

# Index

- \* **classes**
  - csAnno-class, 10
- \* **datasets**
  - info, 25
- \* **internal**
  - ChIPseeker-package, 4
  - reexports, 52
- ., 5
- annotatePeak, 5, 10, 33, 35, 41
- as.data.frame.csAnno, 7
- as.GRanges, 8
  
- check\_upstream\_and\_downstream, 8
- ChIPseeker (ChIPseeker-package), 4
- ChIPseeker-package, 4
- combine\_csAnno, 9
- covplot, 9
- csAnno-class, 10
  
- downloadGEObedFiles, 11
- downloadGSMbedFiles, 11
- dropAnno, 12
  
- enrichAnnoOverlap, 12
- enrichPeakOverlap, 13
  
- getAnnoStat, 14
- getBioRegion, 14, 21
- getGeneAnno, 15
- getGenomicAnnotation, 16
- getGEOgenomeVersion, 17
- getGEOInfo, 17
- getGEOspecies, 18
- getNearestFeatureIndicesAndDistances, 18
- getPromoters, 19, 21
- getSampleFiles, 20
- getTagMatrix, 20
- getTagMatrix.binning.internal, 22
- getTagMatrix.internal, 23
  
- getTagMatrix2, 23
- getTagMatrix2.binning.internal, 24
- getTagMatrix2.internal, 25
- ggVennDiagram, 57
- GRangesList, 52
- GRangesList (reexports), 52
- gsminfo (info), 25
  
- info, 25
  
- makeBioRegionFromGranges, 21, 26
- mclapply, 14
  
- overlap, 27
  
- peak\_Profile\_Heatmap, 30
- peakHeatmap, 27
- peakHeatmap\_multiple\_Sets, 29
- plotAnnoBar, 7, 31, 35
- plotAnnoBar, csAnno, ANY-method (plotAnnoBar), 31
- plotAnnoBar, csAnno-method (csAnno-class), 10
- plotAnnoBar, list-method (plotAnnoBar), 31
- plotAnnoBar.data.frame, 32
- plotAnnoPie, 7, 33, 33
- plotAnnoPie, csAnno, ANY-method (plotAnnoPie), 33
- plotAnnoPie, csAnno-method (csAnno-class), 10
- plotAnnoPie.csAnno, 34
- plotAvgProf, 36
- plotAvgProf.binning, 37
- plotAvgProf2, 38
- plotDistToTSS, 7, 39
- plotDistToTSS, csAnno, ANY-method (plotDistToTSS), 39
- plotDistToTSS, csAnno-method (csAnno-class), 10

plotDistToTSS, list-method  
    (plotDistToTSS), 39  
plotDistToTSS.data.frame, 40  
plotMultiProf, 41  
plotMultiProf.binning, 42  
plotMultiProf.binning.internal, 43  
plotMultiProf.normal, 44  
plotMultiProf.normal.internal, 45  
plotPeakProf, 45  
plotPeakProf2, 48  
plotPeakProf\_MultiWindows, 50  
  
readPeakFile, 52  
reexports, 52  
rel, 52  
rel (reexports), 52  
  
scale\_colour\_brewer, 28, 29, 31, 55  
seq2gene, 53  
show, 54  
show, csAnno, ANY-method (show), 54  
show, csAnno-method (csAnno-class), 10  
shuffle, 54  
subset, csAnno-method (csAnno-class), 10  
  
tagHeatmap, 55  
tagMatrixList (info), 25  
  
ucsc\_release (info), 25  
upsetplot, 56  
upsetplot, csAnno-method (csAnno-class),  
    10  
  
vennpie, 56  
vennpie, csAnno-method (csAnno-class), 10  
vennplot, 57  
vennplot.peakfile, 58