Package 'Rcade'

October 16, 2019

Title R-based analysis of ChIP-seq And Differential Expression - a tool for integrating a count-based ChIP-seq analysis with differential expression summary data

Version 1.26.0

Date 2012-01-25

Author Jonathan Cairns

Maintainer Jonathan Cairns <jmcairns200@gmail.com>

Description Rcade (which stands for ``R-based analysis of ChIP-seq And Differential Expression") is a tool for integrating ChIP-seq data with differential expression summary data, through a Bayesian framework. A key application is in identifing the genes targeted by a transcription factor of interest - that is, we collect genes that are associated with a ChIP-seq peak, and differential expression under some perturbation related to that TF.

Depends R (>= 2.14.0), methods, GenomicRanges, Rsamtools, baySeq

Imports utils, grDevices, stats, graphics, rgl, plotrix, S4Vectors, IRanges, GenomeInfoDb, GenomicAlignments

Suggests limma, biomaRt, RUnit, BiocGenerics, BiocStyle

License GPL-2

biocViews DifferentialExpression, GeneExpression, Transcription, ChIPSeq, Sequencing, Genetics

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

git_branch RELEASE_3_9

git_last_commit a918a8d

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

R topics documented:

`constructRcadeTable` *Construct Rcade Table*

Description

Most Rcade users will not need to call this function directly. This function constructs a full Rcade table from ChIP and DE data.

Usage

```
constructRcadeTable(DE, DElookup, chip, annoZone, annoZoneGeneidName, DE.prior=NULL, ChIP.prior=NU
```

Arguments

`DE`	`data.frame` - DE data (see details section, below)
`DElookup`	`list` - a lookup table specifing the columns of interest in the DE argument. FIXME - list mandatory columns
`chip`	`data.frame` - ChIP information as ... Columns correspond to samples, and rows should correspond to bins defined by the `annoZone` arguments's rows.
`annoZone`	`GRanges` - The genomic bins used in the ChIP-seq analysis. FIXME Metadata must be present.
`annoZoneGeneidName`	`character` - The column in the metadata of `annoZone` argument that contains the geneIDs.
`DE.prior`	As per `RcadeAnalysis`
`ChIP.prior`	As per `RcadeAnalysis`
`prior.mode`	As per `RcadeAnalysis`
`prior`	As per `RcadeAnalysis`

Value

`data.frame`

Author(s)

Jonathan Cairns

See Also

`RcadeAnalysis`

Examples

```
data(RcadeSTAT1)

dir <- file.path(system.file("extdata", package="Rcade"), "STAT1")

DE <- getDE(RcadeSTAT1)
DElookup <- list(GeneID="ENSG", logFC="logFC", B="B",
"Genes.Location", "Symbol")

chip <- getChIP(RcadeSTAT1)
annoZone <- getChIP(RcadeSTAT1, what="annoZones")

x <- constructRcadeTable(DE, DElookup, chip, annoZone, annoZoneGeneidName="ENSG", prior.mode="assumeIndepende
```

---

countReads *Count Reads*

Description

Most Rcade users will not need to call this function directly. Given targets information linking to bam files, count the reads that lie in defined bins.

Usage

```
countReads(annoZone, targets, fileDir=NULL, dontCheckTargets=FALSE)
```

Arguments

annoZone	GRanges - The bins to be used when counting reads.
targets	data.frame - Targets file (see vignette)
fileDir	character - The directory in which the raw ChIP-seq data files are kept.
dontCheckTargets	logical - If TRUE, the targets file is not checked for consistency/appropriate field names. This should not be changed for Rcade purposes, but may be useful if you wish to obtain bin counts for some other purpose. Make sure relevant column names are lower case. Use at your own risk!

Value

Matrix of read counts, with columns corresponding to samples and rows corresponding to bins.

Author(s)

Jonathan Cairns

See Also

RcadeAnalysis

Examples

```
dir <- file.path(system.file("extdata", package="Rcade"), "STAT1")

targets <- read.csv(file.path(dir, "targets.csv"), as.is = TRUE)

anno <- read.csv(file.path(dir, "anno.csv"))
anno <- anno[order(anno$chromosome_name),]
colnames(anno) <- c("ENSG","chr","start","end","str")
ChIPannoZones <- defineBins(anno, zone=c(-1500, 1500), geneID="ENSG")

x <- countReads(ChIPannoZones, targets, fileDir = dir)
```

`defineBins` *Define Bins*

Description

Defines bins about the 5' end of certain features of interest - these features are usually transcripts.

Usage

```
defineBins(anno, zone, geneID="ensembl_gene_id", removeDuplicates=TRUE)
```

Arguments

anno	`data.frame` (or, an object that can be coerced to a data.frame, such as a GRanges) - Annotation information, corresponding to features of interest (usually transcripts). Only the 5' end of each object is used. Rcade expects the following column names: `chr,start,end,str`. These correspond to chromosome name, start co-ordinate, end co-ordinate and strand. Additionally, there must be another column specifying a gene ID, specified by the `geneID` argument.
zone	`integer` - must be a length 2 vector of form `c(relative.start,relative.end)`. For example, `zone = c(-10,100)` will produce bins that start 10bp 5' of each transcript's TSS and end 100bp 3' of it.
geneID	`character` or `integer` - The column in `anno` that contains a geneID (or some other feature ID).
removeDuplicates	`logical` - If `TRUE`, then any rows that share the same geneID and genomic location as another row will be removed (even if any of the other columns are different).

Details

The `defineBins` function is useful when ChIP-seq bins are defined about ... . In particular, biomaRt data can be fed into this function directly. FIXME See vignette.

Value

A `GRanges` object, corresponding to genomic bins. This output can be used as the `ChIPannoZones` argument in `RcadeAnalysis`.

Author(s)

Jonathan Cairns

See Also

RcadeAnalysis

Examples

```
## Not run:  ##acquire annotation from biomaRt
library(biomaRt)
anno <- getBM(
attributes= c("ensembl_gene_id", "chromosome_name",
"transcript_start", "transcript_end", "strand"),
mart= useDataset("hsapiens_gene_ensembl", useMart("ensembl"))
)

## End(Not run)
#define bins about the annotation
anno <- anno[order(anno$chromosome_name),]
colnames(anno) <- c("ENSG","chr","start","end","str")
ChIPannoZones <- defineBins(anno, c(-1500, 1500), geneID = "ENSG")
```

---

diffCountsBaySeq *Differential Counts wrapper - BaySeq*

Description

Most Rcade users will not need to call this function directly. A function that provides a wrapper for the methods in the BaySeq package.

Usage

```
diffCountsBaySeq(counts, targets, annoZones, cl = NULL, getLibsizesArgs = list(estimationType = "qu
```

Arguments

counts	Counts from countReads
targets	Data.frame - Information about the ChIP data files. Mandatory column names are: "fileid", "sampleid", "factor", "filepath".
annoZones	GRanges specifying the bins of interest, with a column in the metadata for the geneID.
cl	cluster from makeCluster in the parallel package.
getLibsizesArgs	List - Arguments to be passed to the getLibsizes function. If a libsizes column is present in the targets file, then these arguments are ignored. getLibsizesArgs$cD is always ignored. See getLibsizes for a list of arguments.

`getPriors.NBArgs`
: See `getPriors` for a list of arguments.
`getPriors.NBArgs$cD` and `getPriors.NBArgs$cl` are always ignored.

`getLikelihoods.NBArgs`
: See `getLikelihoods` for a list of arguments.
`getLikelihoods.NBArgs$cD` and `getLikelihoods.NBArgs$cl` are always ignored.

`libsizes`
: Library sizes FIXME

Value

`data.frame` containing differential count information.

Author(s)

Jonathan Cairns

References

Hardcastle, T. J., & Kelly, K. A. (2010). baySeq: Empirical Bayesian methods for identifying differential expression in sequence count data. BMC Bioinformatics, 11, 422.

See Also

`RcadeAnalysis`

Examples

```
dir <- file.path(system.file("extdata", package="Rcade"), "STAT1")

targets <- read.csv(file.path(dir, "targets.csv"), as.is = TRUE)

anno <- read.csv(file.path(dir, "anno.csv"))
anno <- anno[order(anno$chromosome_name),]
colnames(anno) <- c("ENSG","chr","start","end","str")
ChIPannoZones <- defineBins(anno, zone=c(-1500, 1500), geneID="ENSG")

counts <- countReads(ChIPannoZones, targets, fileDir = dir)

x <- diffCountsBaySeq(counts, targets, ChIPannoZones)
```

---

exportRcade-methods *exportRcade and ...*

Description

Methods for exporting Rcade objects, either to disk or in R.

Usage

```
exportRcade(x, directory="RcadeOutput", cutoffMode="top", cutoffArg = 1000, justGeneID=FALSE, remo
```

Arguments

x	An Rcade object.
directory	character - The directory to export output to.
cutoffMode	character - The method to cut off each list (see Details). Must be "all", "top", "B" or "FDR".
cutoffArg	numeric - What cutoff to use (see Details).
justGeneID	logical - if TRUE, export only the geneID column. If FALSE, export all columns.
removeDuplicates	character - Should we remove duplicate GeneIDs and, if so, should we do this before or after applying the cutoff? Must be "beforeCutoff", "afterCutoff" or "none". (If removing duplicates then, for each list, the entry with the highest B value is retained.)

Details

This function exports Rcade output to disk - specifically, it creates the following files:

File:	ChIP:	DE
ChIP.csv	Present (needs log ratio > 0)	Ignored
ChIPonly.csv	Present (needs log ratio > 0)	Absent
DEandChIP.csv	Present (needs log ratio > 0)	Present
DownChIP.csv	Present (needs log ratio > 0)	Present (logFC < 0)
Down.csv	Ignored	Present (logFC < 0)
DownNoChIP.csv	Absent	Present (logFC < 0)
Nothing.csv	Absent	Absent
UpChIP.csv	Present (needs log ratio > 0)	Present (logFC > 0)
Up.csv	Ignored	Present (logFC > 0)
UpNoChIP.csv	Absent	Present (logFC > 0)

Each file contains genes appropriate to its hypothesis, sorted by descending B value (i.e. ranked from most interesting to least interesting). For example, if you wanted the genes that display DE (either up or down) and also have ChIP signal present, you would look at the top rows of DEandChIP.csv. For genes that have a ChIP signal but explicitly show no DE, use ChIPonly.csv.

A cutoff is applied to each list, according to the value of cutoffMode, referring to cutoffArg if necessary:

cutoffMode = "all" cutoff ignored, all results written to disk.

cutoffMode = "top" Take the top N genes, where N is specified by cutoffArg.

cutoffMode = "B" Take all genes with that satisfy B > cutoffArg, where B is the log-odds.

cutoffMode = "FDR" The expected false positive rate, FPR, and the expected false negative rate, FNR, are calculated using B values.

The cutoff chosen is the one that maximizes the value of FPR + cutoffArg*FNR.

Usage

```
exportRcade(x,directory="RcadeOutput",cutoffMode="top",cutoff = 100,justGeneID=FALSE,removeDupli
```

Examples

```
data(RcadeSTAT1)
## Not run: exportRcade(RcadeSTAT1)
```

Rcade-class *Rcade Class*

Description

The main class in Rcade. This class contains data pertaining to any relevant DE experiments, ChIP-seq experiments, and Rcade output from linking the previous two.

Objects of this class are typically created with the `RcadeAnalysis` function.

Plotting methods

`plotPCA(x,...)`: Perform PCA analysis on the ChIP-seq data and plot the results.

`plotMM(x,DE.abs=FALSE,...)`: Plot ChIP log-ratios against DE log-ratios. If `DE.abs=TRUE`, then absolute values of DE log-ratios are plotted. `...` arguments are passed to `plot`.

`plotBB(x,...)`: Plot ChIP log-odds against DE log-odds. `...` arguments are passed to `plot`.

`plotBBB(x,...)`: (NB: Requires the CRAN package **rgl**.) 3D plot comparing log-odds values for ChIP, DE and combined ChIP & DE. `...` arguments are passed to `plot`.

Accessors

`getDE(x,what="summary")`: Get DE information. `what` can be: `"summary"` for the DE analysis, `"prior"` for the prior probability/probabilities of DE presence.

`getChIP(x,what="summary")`: Get ChIP analysis information. `what` can be: `"summary"` for the analysis, `"counts"` for the raw counts, `"annoZones"` for the bins used in the analysis, `"prior"` for the prior probability/probabilities of ChIP signal presence, or `"targets"` for the targets file.

`getRcade(x)`: Get the Rcade table - i.e. combined DE/ChIP information.

Author(s)

Jonathan Cairns

References

NA

See Also

`RcadeAnalysis`

Examples

```
data(RcadeSTAT1)
RcadeSTAT1

x <- getChIP(RcadeSTAT1)
y <- getDE(RcadeSTAT1)
z <- getRcade(RcadeSTAT1)

plotMM(RcadeSTAT1)
plotPCA(RcadeSTAT1)
library(rgl) ##required for plotBBB
plotBBB(RcadeSTAT1)
```

`RcadeAnalysis` *Rcade Analysis*

Description

The main function in Rcade - reads in DE information, processes ChIP data from raw .bam files, and then combines the two to form an Rcade object.

Usage

```
RcadeAnalysis(DE, ChIPannoZones, annoZoneGeneidName, ChIPtargets, ChIPfileDir, cl, DElookup, DE.pr
```

Arguments

`DE`	`Data.frame` - DE summary information for genes of interest. For example, output from limma. EITHER `DE` must have column names "geneID", "logfc" and "B" (case insensitive) OR you should specify DElookup.
`ChIPannoZones`	`GRanges` specifying the bins of interest, with a column in the metadata for the geneID.
`annoZoneGeneidName`	`character` - the name of the column in `ChIPannoZones`'s metadata corresponding to geneID.
`ChIPtargets`	`Data.frame` - Information about the ChIP data files. Mandatory column names are: "fileid", "sampleid", "factor", "filepath".
`ChIPfileDir`	`character` - Directory, within which "filepath" of `ChIPtargets` is evaluated.
`cl`	A `cluster` from `makeCluster` in the parallel package.
`DElookup`	`list` - lookup table of form `list(RcadeField1 = DEcolumn1,RcadeField2 = DEcolumn2,...)`. If you don't specify this argument, then Rcade will try to find the mandatory fields automatically but will not keep any of the other information in its output.
`DE.prior`	`numeric` - The prior probability of DE for each GeneID. Either a scalar, or a vector where the Nth element corresponds to the Nth row of the DE argument. Ignored if `prior.mode = "assumeIndependent"`. For example, if using DE analysis from the limma package (default settings), then set `DE.prior = 0.01`.

`prior.mode` The method used to create prior probabilities in the Rcade table. Current options are:

`assumeIndependent`: Under the prior, ChIP counts and DE log ratios are assumed independent; that is, the prior is of form P(D,C)=P(D)P(C). No need to specify the `prior` argument.

`keepChIP`: The prior is factorized as form P(D,C)=P(D|C)P(C). P(C) is taken from the differential count algorithm used. User must specify the `prior` argument as c(P(D|C), P(D|not C)).

`prior` See `prior.mode`.

`...` Additional arguments.

Details

This is the main analysis function in Rcade. The user should specify information relating to the DE and ChIP data for the experiment in question. Rcade will process these data and rank genes by the combined DE and ChIP strength.

Value

An Rcade object.

Author(s)

Jonathan Cairns

See Also

RcadeAnalysis

Examples

```
dir <- file.path(system.file("extdata", package="Rcade"), "STAT1")

DE <- read.csv(file.path(dir, "DE.csv"))
DElookup <- list(GeneID="ENSG", logFC="logFC", B="B",
"Genes.Location", "Symbol")

targets <- read.csv(file.path(dir, "targets.csv"), as.is = TRUE)

anno <- read.csv(file.path(dir, "anno.csv"))
anno <- anno[order(anno$chromosome_name),]
colnames(anno) <- c("ENSG","chr","start","end","str")
ChIPannoZones <- defineBins(anno, zone=c(-1500, 1500), geneID="ENSG")

Rcade <- RcadeAnalysis(DE, ChIPannoZones, annoZoneGeneidName="ENSG", ChIPtargets=targets, ChIPfileDir = dir,
DElookup=DElookup)
```

RcadeSTAT1 *Rcade object - STAT1 data*

Description

The Rcade object generated in the vignette, `vignette("Rcade")`.

Usage

```
data(RcadeSTAT1)
```

Format

Object of Rcade class.

Source

Differential Expression data from Array Express, http://www.ebi.ac.uk/arrayexpress, under accession number E-GEOD-11299.

STAT1 ChIP-seq data from the Snyder lab, as part of the ENCODE consortium\ Input DCC accession numbers: wgEncodeEH000611 and wgEncodeEH000612\ ChIP DCC accession number: wgEncodeEH000614

Examples

```
data(RcadeSTAT1)
RcadeSTAT1
## maybe str(RcadeSTAT1) ; plot(RcadeSTAT1) ...
```

RcadeTrack-class *RcadeTrack Class*

Description

Class for storing information pertaining to a set of ChIP-seq experiments - in particular, count data and

Details

Most users should not need to interact with this class - please use Rcade-class instead.

Author(s)

Jonathan Cairns

References

NA

See Also

Rcade-class

Index