Package 'PCBS'

May 7, 2026

Type Package

Title Principal Component BiSulfite

Version 0.1.1

Description A system for fast, accurate, and flexible whole genome bisulfite sequencing (WGBS) data analysis of two-condition comparisons. Principal Component BiSulfite, 'PCBS', assigns methylated loci eigenvector values from the treatment-delineating principal component in lieu of running millions of pairwise statistical tests, which dramatically increases analysis flexibility and reduces computational requirements. Methods: <https://katlande.github.io/PCBS/articles/Differential_Methylation.html>.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 2.10)

URL https://github.com/katlande/PCBS

BugReports https://github.com/katlande/PCBS/issues

Imports ggplot2, tibble, ggrepel, dplyr, data.table

NeedsCompilation no

Author Kathryn Lande [aut, cre, cph]

Maintainer Kathryn Lande <kathryn.lande@mail.mcgill.ca>

Repository CRAN

Date/Publication 2024-08-27 18:00:02 UTC

Contents

---

addRanks *Add ranks to eigenvector scores.*

Description

Defines the best principle component to use for downstream analysis.

Usage

```
addRanks(ranks)
```

Arguments

ranks	getPCRanks output data frame.

Value

The input `data.frame` with rank order and absolute rank order columns.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"))
ranks <- addRanks(ranks)
```

CheckOvercompression *Check if DMR calling seed number is ovcompressed.*

Description

Identifies if seed number to use for DMR calling causes overcompression.

Usage

```
CheckOvercompression(ranks, CpG_cutoff, values, max.dmr.size, return.plot)
```

Arguments

ranks	Rank data frame from getPCRanks.
CpG_cutoff	NULL or numeric. If NULL, seed numbers tested will be input of the values argument. If numeric, seed numbers tested will be CpG_cutoff*values argument. Recommended to us rankDist estimate if not null
values	Numeric vector, either seed numbers to test if CpG_cutoff=NULL or multipliers if CpG_cutoff is numeric
max.dmr.size	Automatic=5000. Maximum DMR expansion size in downstream analysis. Note: pipeline is optimized for 5000bp max DMR size, it is not recommended to play with this value.
return.plot	T/F, whether to return a plot or a numeric representing the best seed number for downstream analysis

Value

If return.plot=T, a grob plotting input seed number vs. compressed seed number is returned. Otherwise, a numeric is returned containing the largest tested input value without detectable overcompression.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
CheckOvercompression(ranks, 980)
```

checkRank *Check rank cut-off values manually.*

Description

Plots a score vs. rank plot with a manually chosen rank cut-off for manual k selection.

Usage

```
checkRank(ranks, cutoff)
```

Arguments

ranks	getPCRanks output data frame
cutoff	integer, rank value to check

Value

Returns a grob plotting the input cutoff on a plot of absolute eigenvector score vs. absolute rank order.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
test_50 <- checkRank(ranks, 50) # set cut-off to 50
test_500 <- checkRank(ranks, 500) # set cut-off to 500
```

chromDict *Convert a rank object into a chromDict.*

Description

Internal to many functions; creates a chromDict for faster computing times. chromDict can be run separately to speed up functions run iteratively. A chromDict is a list of chromosome-specific data.tables generated from ranks.

Usage

```
chromDict(ranks)
```

Arguments

ranks	getPCRanks output data frame

Value

Returns a list of `data.tables` for each chromosome, for faster analysis. Used internall by many PCBS functions.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
chromDictObj <- chromDict(ranks)
```

`chromDictMeth` *Create a chromDict of percent methylation difference at all sites.*

Description

chromDicts are lists of keyed data.tables that enable very fast computing times. Much like the primary PCBS function chromDict() which makes a chromDict of all locus ranks, the chromDictMeth() function makes is a list of chromosome-specific data.tables containing percent methylation differences at all loci..

Usage

```
chromDictMeth(mat, IDs, filter_thresh)
```

Arguments

`mat`	`data.frame` object containing percent methylation and locus information for all sites, in the format of `eigen`
`IDs`	`character` vector of IDs containing the common names for compared conditions. E.g., for samples trt1 & trt2 vs. ctl1 & ctl2, IDs=c("trt1", "ctl")
`filter_thresh`	`Integer`, a coverage threshold for filtering, where CpG coverage of all samples must be larger than this value. Auto=50

Value

Returns a list of `data.tables` for each chromosome, for faster analysis.

Examples

```
chromDictMethylDiff <- chromDictMeth(eigen, c("trt", "ctl"))
```

DefineBestPC *Identify your best principle component.*

Description

Defines the best principle component to use for downstream analysis.

Usage

```
DefineBestPC(mat, IDs, filter_thresh, return.plot)
```

Arguments

mat	Bismark2Matrix.R output file, or data frame object
IDs	A character vector of IDs containing the common names for compared conditions. E.g., for samples trt1, trt2 vs. ctl1, ctl2, IDs=c("trt", "ctl")
filter_thresh	A coverage threshold for filtering, where CpG coverage of all samples must be larger than this value
return.plot	T/F, whether to return a PCA plot or a numeric representing the best principle component for downstream analysis

Value

If `return.plot=T`, a `grob` plotting a PCA of percent methylation of all samples is returned. Otherwise, a `numeric` representing the best principal component to use for PCBS analysis is returned.

Examples

```
DefineBestPC(eigen, IDs = c("trt", "ctl"))
```

eigen *Simulated WGBS data for PCBS vignettes*

Description

simulated WGBS data with added DMRs and DMLs

Usage

```
eigen
```

Format

A data frame with 50000 observations on the following 13 variables.

`cpgID` a character vector, chrom:locus

`trt1_PercMeth` a numeric vector, percent methylated of sample trt1

`trt1_nCpG` a numeric vector, depth of sample trt1

`trt2_PercMeth` a numeric vector, percent methylated of sample trt2

`trt2_nCpG` a numeric vector, depth of sample trt2

`trt3_PercMeth` a numeric vector, percent methylated of sample trt3

`trt3_nCpG` a numeric vector, depth of sample trt3

`ctl1_PercMeth` a numeric vector, percent methylated of sample ctl1

`ctl1_nCpG` a numeric vector, depth of sample ctl1

`ctl2_PercMeth` a numeric vector, percent methylated of sample ctl2

`ctl2_nCpG` a numeric vector, depth of sample ctl2

`ctl3_PercMeth` a numeric vector, percent methylated of sample ctl3

`ctl3_nCpG` a numeric vector, depth of sample ctl3

Source

generated through simulation

`getPCRanks` *Get CpG eigenvector scores from a principle component.*

Description

Returns eigenvector scores for input CpG sites.

Usage

```
getPCRanks(mat, IDs, PC, filter_thresh)
```

Arguments

`mat`	Bismark2Matrix.R output file, or data frame object
`IDs`	A character vector of IDs containing the common names for compared conditions. E.g., for samples trt1 & trt2 vs. ctl1 & ctl2, IDs=c("trt1", "ctl")
`PC`	Integer, which principle component to use. Use to DefineBestPC if unsure.
`filter_thresh`	Integer, a coverage threshold for filtering, where CpG coverage of all samples must be larger than this value.

Value

Returns a `data.frame` of eigenvector scores for all loci.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
```

`getRegionScores` *Calculated methylation significance in a set of regions.*

Description

Returns p-values and Z-scores for CpGs in a set of regions, compared to a local null background distribution.

Usage

```
getRegionScores(ranks, regions,  chromDictObj)
```

Arguments

`ranks`	getPCRanks output data.frame, only necessary if chromDictObj=NULL
`regions`	A three-column dataframe containing a set of regions to test. Columns = chrom, start, end.
`chromDictObj`	chromDict() output object, recommended input instead of ranks.

Value

Returns a `data.frame` with significance scores for all input regions.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
DMLs <- addRanks(ranks)

# data.frame of regions to test:
regions <- data.frame(chr=c("chr3", "chr3", "chr1"),
                      s=c(4920450, 3961576, 300000),
                      e=c(4923267, 3963805, 302900),
                      ID=c("Hypo-DMR", "partial Hyper-DMR", "random"))

getRegionScores(DMLs, regions)
```

get_all_DMRs *Nested DMR calling function within Get_Novel_DMRs()*

Description

Expands DMRs from collapsed seeds iteratively.

Usage

```
get_all_DMRs(chromDictObj, seeds, res=40, max.dmr.size=3000, min.dmr.cpgs=10,
min.absZscore, null)
```

Arguments

chromDictObj	chromDict() output.
seeds	compressed seeds
res	Get_Novel_DMRs arg DMR_resolution
max.dmr.size	Get_Novel_DMRs arg QueryLimit
min.dmr.cpgs	Get_Novel_DMRs arg minCpGs
min.absZscore	Get_Novel_DMRs arg minZ
null	null distribution

Value

Returns a `list` with two indicies, representing intermediate DMR calls within the `Get_Novel_DMRs()` function and a `list` of background regions..

Get_Novel_DMRs *Call DMRs from WGBS data.*

Description

DMR Calling.

Usage

```
Get_Novel_DMRs(ranks, nSeeds, chromDictObj, DMR_resolution,
QueryLimit, minCpGs, minZ, perms)
```

Arguments

ranks	Rank data frame from getPCRanks.
nSeeds	Integer, number of input seeds for DMR expansion.
chromDictObj	chromDict() output. If null, chromDict() is run internally.
DMR_resolution	Automatic=NULL. Integer, number of bases to increase the DMR by with each expansion. If NULL, QueryLimit/25.
QueryLimit	Automatic=5000. Maximum DMR expansion size (bp)
minCpGs	Automatic=15. Minimum CpGs in a DMR region, regions with fewer CpGs will be discarded.
minZ	Automatic=1. Absolute Z score threshold for DMR calling; internal value. Not recommended to play with this setting.
perms	Automatic=1000. Number of permutations to use when defining the null distribution. Increasing this value largely influences computational time with minimal return

Value

Returns a `data.frame` of all novel DMRs.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
DMRs <- Get_Novel_DMRs(ranks, 2940, minCpGs=10)
```

---

lin *Find y value of linear regression given x.*

Description

Find y value of linear regression given x.

Usage

```
lin(x, l)
```

Arguments

x	x coordinate
l	linear model of lm()

Value

Returns a column `numeric` representing the y coordinate at the input `x` of the linear model `l`.

lmIntx *PC-Intersect nested function.*

Description

Finds the intersection point of two linear regression models, lm().

Usage

```
lmIntx(fit1, fit2, rnd=2)
```

Arguments

fit1	lm() model 1
fit2	lm() model 2
rnd	number of significant figures

Value

Returns a 2 column data.frame of one row, containing the x and y coordinates of the intersection point of the input models.

MethyDiff_Set *Add a mean methylation column to a data.frame of regions.*

Description

Using a chromDictMeth() output object, quickly calculate the mean methylation difference across set of regions.

Usage

```
MethyDiff_Set(chromDictMeth, regions)
```

Arguments

chromDictMeth	chromDictMeth() output object
regions	data.frame of regions, where column 1 = chromosome, column 2 = region start, and column 3 = region end

Value

Returns the regions object with a mean percent methylation column.

Examples

```
chromDictMethylDiff <- chromDictMeth(eigen, c("trt", "ctl"))
regions <- data.frame(chr=c("chr3", "chr3", "chr1"),
                      s=c(4920450, 3961576, 300000),
                      e=c(4923267, 3963805, 302900),
                      ID=c("Hypo-DMR", "partial Hyper-DMR", "random"))

MethyDiff_Set(chromDictMethylDiff, regions)
```

MethylDiff *Get the mean methylation difference across a specified region.*

Description

Using a chromDictMeth() output object, quickly calculate the mean methylation difference across a user-specified region.

Usage

```
MethylDiff(chromDictMeth, chrom, start, end)
```

Arguments

chromDictMeth	chromDictMeth() output object
chrom	character, chromosome
start	integer, region start
end	integer, region end

Value

Returns a list of data.tables for each chromosome, for faster analysis.

Examples

```
chromDictMethylDiff <- chromDictMeth(eigen, c("trt", "ctl"))
MethylDiff(chromDictMethylDiff, "chr3", 4920450, 4923267)
```

methylDiff_metagene *Make a metagene from mean percent methylation differences.*

Description

Uses mean binned percent methylation differences across a set of regions to draw a metagene.

Usage

```
methylDiff_metagene(chromDictMethObj, regions, bin, title,
xaxis, yaxis, return.data, linecol, value)
```

Arguments

chromDictMethObj	chromDictMeth() output object
regions	A three-column data.frame containing a set of regions to test. Columns = chrom, start, end.
bin	integer, number of bins to use in metagenes. Default=100.
title	Output plot title
xaxis	Output plot x-axis title
yaxis	Output plot y-axis title
return.data	T/F, whether to return a plot, or data that can be run with plot_metagene() or multiple_metagenes().
linecol	Colour for line, auto="red"
value	Name of the plotted metric in chromDictMethObj. Only needs to be set explicity for abnormal use cases where chromDictMethObj contains a non-rank value output by chromDict().

Value

If `return.data=F`, returns a `grob` containing a metagene plot. Otherwise, returns a `list` of two `data.frames` containing metagene and metagene standard error plotting information.

Examples

```
chromDictMethylDiff <- chromDictMeth(eigen, c("trt", "ctl"))
regions <- data.frame(chr=c("chr3", "chr3", "chr1"),
                      s=c(4920450, 3961576, 300000),
                      e=c(4923267, 3963805, 302900),
                      ID=c("Hypo-DMR", "partial Hyper-DMR", "random"))

methylDiff_metagene(chromDictMethylDiff, regions)
```

`multiple_metagenes` *Plot multiple metagene data objects on a single plot.*

Description

Plots multiple metagene object using the raw data generated by score_metagene().

Usage

```
multiple_metagenes(data_list, set_names, title, xaxis, yaxis, legend.title, col, se_alpha)
```

Arguments

`data_list`	List of score_metagene() raw data output
`set_names`	Character vector of names for score_metagene() object
`title`	Output plot title
`xaxis`	Output plot x-axis title
`yaxis`	Output plot y-axis title
`legend.title`	T/F, whether to show legend title
`col`	Vector of colours to use for lines
`se_alpha`	0-1, alpha value for standard error shading

Value

Returns a `grob` containing a plot of the input metagene data.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
DMRs <- Get_Novel_DMRs(ranks, 2940, minCpGs=10)

# Select all significantly hypomethylated DMRs:
hypo_DMRs <- DMRs[DMRs$FDR <= 0.05 & DMRs$DMR_Zscore < 0,]
# Select all significantly hypermethylated DMRs:
hyper_DMRs <- DMRs[DMRs$FDR <= 0.05 & DMRs$DMR_Zscore > 0,]

# select chrom, start, and end of all hyper DMRs
regions_hypo <- hypo_DMRs[c(1:3)]
regions_hyper <- hyper_DMRs[c(1:3)]

# return.data = T returns raw data instead of a plot:
hyper_metagene <- score_metagene(ranks, regions_hyper, return.data = TRUE)
hypo_metagene <- score_metagene(ranks, regions_hypo, return.data = TRUE)

# The multiple_metagenes function plots multiple metagenes
# using a list of raw data objects from score_metagene().
multiple_metagenes(data_list = list(hyper_metagene, hypo_metagene),
```

```
set_names = c("Hyper DMRs", "Hypo DMRs"),
title="Metagenes of DMR Regions", legend.title = FALSE)
```

`Ol_Reliable` *PCBS ggplot theme.*

Description

Custom theme for ggplot used by all PCBS output figures.

Usage

```
Ol_Reliable()
```

Value

Theme for ggplot objects used by PCBS.

Examples

```
df <- data.frame(A=c(1,2,3), B=c(1,2,3))

ggplot2::ggplot(df, ggplot2::aes(x=A, y=B))+
ggplot2::geom_point()+
Ol_Reliable()
```

`oneSeed` *Nested DMR calling function within within get_all_DMRs(), Get_Novel_DMRs()*

Description

Expands one DMR from a single point.

Usage

```
oneSeed(chroms, seed, resolution, max.size, mincpgs, null_list, Zlim=1)
```

Arguments

`chroms`	chromDict() output.
`seed`	seed value input
`resolution`	Get_Novel_DMRs arg DMR_resolution
`max.size`	Get_Novel_DMRs arg QueryLimit
`mincpgs`	Get_Novel_DMRs arg minCpGs
`null_list`	get_all_DMRs arg null
`Zlim`	Get_Novel_DMRs arg minZ

Value

returns a `list` of two indices, containing a `data.frame` with the output from a single compressed seed expansion and a `data.frame` of locus information around the seed expansion, intended for use within the `Get_Novel_DMRs()` function.

`plot_metagene` *Generate a metagene plot from raw metagene data.*

Description

Plots a metagene object using the raw data generated by score_metagene().

Usage

```
plot_metagene(data, title, xaxis, yaxis, linecol)
```

Arguments

`data`	list, score_metagene() raw data output
`title`	Output plot title
`xaxis`	Output plot x-axis title
`yaxis`	Output plot y-axis title
`linecol`	Colour for line, auto="red"

Value

Returns a `grob` containing a plot of the input metagene data.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
DMRs <- Get_Novel_DMRs(ranks, 2940, minCpGs=10)

# Select all significantly hypomethylated DMRs:
hypo_DMRs <- DMRs[DMRs$FDR <= 0.05 & DMRs$DMR_Zscore < 0,]

# select chrom, start, and end of all hyper DMRs
regions_hypo <- hypo_DMRs[c(1:3)]

# return.data = T returns raw data instead of a plot:
hypo_metagene <- score_metagene(ranks, regions_hypo, return.data = TRUE)

plot_metagene(hypo_metagene)
```

rankDist *Identify the best rank cut-off for significant CpGs.*

Description

Automated rank cut-off estimator for input CpGs.

Usage

```
rankDist(ranks, draw_intersects, noise_perc, mode, return.plot)
```

Arguments

ranks	getPCRanks output data frame.
draw_intersects	T/F whether to draw intersect lines if return.plot=T
noise_perc	Automatic=0.5, numeric between 0 and 1. Fraction of ranks to use to model the background noise. Not recommended to play with this value. Increasing/decreasing returns a looser/stricter threshold, respectively.
mode	"intersect" or "strict", determine cut-off with "intersect" or "strict" method. "Strict" is recommended for sets with lower variability
return.plot	T/F, whether to return a plot or a numeric

Value

If `return.plot=T`, a `grob` plotting the estimated `cutoff` on a plot of absolute eigenvector score vs. absolute rank order is returned. Otherwise, a `numeric` of the estimated cut-off is returned.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
rankDist(ranks, mode="intersect")
```

score_metagene *Make a metagene from PC Scores.*

Description

Uses mean binned PC scores across a set of regions to draw a metagene.

Usage

```
score_metagene(ranks, regions, bin, title, xaxis, yaxis,
chromDictObj, return.data, linecol)
```

Arguments

ranks	getPCRanks output data.frame
regions	A three-column data.frame containing a set of regions to test. Columns = chrom, start, end.
bin	integer, number of bins to use in metagenes. Default=100.
title	Output plot title
xaxis	Output plot x-axis title
yaxis	Output plot y-axis title
chromDictObj	Optional chromDictObject made from chromDict(), runs internally if set to NULL (default). Scripts that run this function multiple times will be sped up by setting this option.
return.data	T/F, whether to return a plot, or data that can be run with plot_metagene() or multiple_metagenes().
linecol	Colour for line, auto="red"

Value

If `return.data=F`, returns a `grob` containing a metagene plot. Otherwise, returns a `list` of two `data.frames` containing metagene and metagene standard error plotting information.

Examples

```
ranks <- getPCRanks(eigen, IDs = c("trt", "ctl"), PC = 1)
DMRs <- Get_Novel_DMRs(ranks, 2940, minCpGs=10)

# select chrom, start, and end of all hyper DMRs:
hyper_DMRs <- DMRs[DMRs$FDR <= 0.05 & DMRs$DMR_Zscore > 0,]
regions_hyper <- hyper_DMRs[c(1:3)]
score_metagene(ranks, regions_hyper)
```

---

se *Standard error of a vector.*

Description

Takes the standard error of a vector.

Usage

```
se(x)
```

Arguments

x	numeric vector.

Value

Returns a numeric, representing the standard error of the input vector.

Examples

```
x <- sample(1:100, 20)
se(x)
```

`tilt` *Component of PCBS ggplot theme.*

Description

Wrapper to title x-axis text in ggplot objects.

Usage

```
tilt()
```

Value

Theme for ggplot objects that cleanly rotates x-axis text.

`trimDMR` *Nested DMR calling function within within Get_Novel_DMRs()*

Description

Trims the edges off of DMR expansions.

Usage

```
trimDMR(df, region, min.dmr.cpgs, max.dmr.size, null_summary, null_values)
```

Arguments

df	DMR expansion output dataframe.
region	get_all_DMRs() region output
min.dmr.cpgs	Get_Novel_DMRs arg minCpGs
max.dmr.size	Get_Novel_DMRs arg QueryLimit
null_summary	null distribution conrtainer
null_values	null distribution conrtainer

Value

Returns a data.frame of all trimmed DMRs for use within the Get_Novel_DMRs() function.

Index