Package 'Damsel'

November 11, 2026

Type Package

Title Damsel: an end to end analysis of DamID

Version 1.8.0

Description Damsel provides an end to end analysis of DamID data.
Damsel takes bam files from Dam-only control and fusion samples and counts the reads matching to each GATC region. edgeR is utilised to identify regions of enrichment in the fusion relative to the control. Enriched regions are combined into peaks, and are associated with nearby genes.
Damsel allows for IGV style plots to be built as the results build, inspired by ggcoverage, and using the functionality and layering ability of ggplot2.
Damsel also conducts gene ontology testing with bias correction through goseq, and future versions of Damsel will also incorporate motif enrichment analysis.
Overall, Damsel is the first package allowing for an end to end analysis with visual capabilities. The goal of Damsel was to bring all the analysis into one place, and allow for exploratory analysis within R.

License MIT + file LICENSE

Encoding UTF-8

Suggests BiocStyle, biomaRt, biovizBase,
BSgenome.Dmelanogaster.UCSC.dm6, knitr, limma, org.Dm.eg.db,
rmarkdown, testthat (>= 3.0.0),
TxDb.Dmelanogaster.UCSC.dm6.ensGene

Config/testthat/edition 3

Depends R (>= 4.4.0)

RoxygenNote 7.3.1

Imports AnnotationDbi, Biostrings, ComplexHeatmap, dplyr, edgeR,
GenomeInfoDb, GenomicFeatures, GenomicRanges, ggbio, ggplot2,
goseq, magrittr, patchwork, plyranges, reshape2, rlang,
Rsamtools, Rsubread, stats, stringr, tidyr, utils

BugReports https://github.com/Oshlack/Damsel

URL https://github.com/Oshlack/Damsel

biocViews DifferentialMethylation, PeakDetection, GenePrediction,
GeneSetEnrichment

VignetteBuilder knitr

LazyData FALSE

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

git_branch RELEASE_3_23

git_last_commit c148934

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Caitlin Page [aut, cre] (ORCID:
<https://orcid.org/0009-0004-7949-8143>)

Maintainer Caitlin Page <caitlin.page@petermac.org>

Contents

Damsel-package . . . 3
annotatePeaksGenes . . . 3
collateGenes . . . 4
countBamInGATC . . . 5
dros_counts . . . 6
geom_dm . . . 7
geom_gatc . . . 8
geom_genes_tx . . . 9
geom_peak . . . 11
getGatcRegions . . . 12
identifyPeaks . . . 13
makeDGE . . . 14
plotCorrHeatmap . . . 15
plotCounts . . . 16
plotCountsDistribution . . . 17
plotCountsInPeaks . . . 18
plotGeneOntology . . . 19
plotWrap . . . 20
random_counts . . . 22
random_edgeR_results . . . 22
random_regions . . . 23
testDmRegions . . . 24
testGeneOntology . . . 25
%>% . . . 26

Index **27**

`Damsel-package` *Damsel: Damsel: an end to end analysis of DamID*

Description

Damsel provides an end to end analysis of DamID data. Damsel takes bam files from Dam-only control and fusion samples and counts the reads matching to each GATC region. edgeR is utilised to identify regions of enrichment in the fusion relative to the control. Enriched regions are combined into peaks, and are associated with nearby genes. Damsel allows for IGV style plots to be built as the results build, inspired by ggcoverage, and using the functionality and layering ability of ggplot2. Damsel also conducts gene ontology testing with bias correction through goseq, and future versions of Damsel will also incorporate motif enrichment analysis. Overall, Damsel is the first package allowing for an end to end analysis with visual capabilities. The goal of Damsel was to bring all the analysis into one place, and allow for exploratory analysis within R.

Author(s)

Maintainer: Caitlin Page <caitlin.page@petermac.org> (ORCID)

See Also

Useful links:

- https://github.com/Oshlack/Damsel
- Report bugs at https://github.com/Oshlack/Damsel

`annotatePeaksGenes` *Annotation of peaks and genes*

Description

'annotatePeaksGenes' identifies the closest gene(s) for the peaks outputted from 'aggregate_peaks()'.

This distance is relative, as the function will identify the closest genes, even if they are up to a million bp away. The max_distance parameter limits this, with a default setting of 5000 bp. All of the possible pairings are visible with 'max_distance=NULL'. The minimum distance between the peak and gene is calculated, (0 if the peak is within the gene or vice versa) and the relative position of the peak to the gene is also provided (Upstream, Downstream, Overlapping upstream, Contained within etc).

Usage

```
annotatePeaksGenes(peaks, genes, regions, max_distance = 5000)
```

Arguments

peaks	A data.frame of peaks as outputted from [aggregate_peaks()].
genes	A data.frame of genes as outputted from [get_biomart_genes()].
regions	A 'GRanges' object of GATC regions.
max_distance	A number providing the limit for the minimum distance from peak to gene. * Default is 5000. If set to 'NULL', will output all available combinations.

Value

A 'list' of 3 'data.frames': * closest - every peak with it's closest gene * top_5 - every peak with list of 5 closest genes * all - all genes matching to each peak and all information

Examples

```
library(TxDb.Dmelanogaster.UCSC.dm6.ensGene)
library(org.Dm.eg.db)
set.seed(123)
example_regions <- random_regions()
dm_results <- random_edgeR_results()
peaks <- identifyPeaks(dm_results)
txdb <- TxDb.Dmelanogaster.UCSC.dm6.ensGene
genes <- collateGenes(genes = txdb, regions = example_regions, org.Db = org.Dm.eg.db)

annotatePeaksGenes(peaks, genes, example_regions, max_distance = 5000)
# view all combinations
annotatePeaksGenes(peaks, genes, example_regions, max_distance = NULL)
```

---

collateGenes *Get list of genes*

Description

Takes a Txdb object, path to a gff file, or a species (biomaRt) and returns a GRanges of genes.

Usage

```
collateGenes(genes, regions, org.Db = NULL, version = NULL)
```

Arguments

genes	A Txdb object, path to file, or a species for accessing biomaRt.
regions	GATC region file.
org.Db	Required if using a Txdb object so to access gene names.
version	Required for using biomaRt.

Value

A GRanges object of genes and available supplementary information - specifically the TSS, and number of GATC regions overlapping the gene.

References

Carlson M (2019). org.Dm.eg.db: Genome wide annotation for Fly. R package version 3.8.2. Durinck S, Spellman P, Birney E, Huber W (2009). "Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt." Nature Protocols, 4, 1184–1191. Durinck S, Moreau Y, Kasprzyk A, Davis S, De Moor B, Brazma A, Huber W (2005). "BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis." Bioinformatics, 21, 3439–3440. Lee, Stuart, Cook, Dianne, Lawrence, Michael (2019). "plyranges: a grammar of genomic data transformation." Genome Biol., 20(1), 4. http://dx.doi.org/10.1186/s13059-018-1597-8. Team BC, Maintainer BP (2019). TxDb.Dmelanogaster.UCSC.dm6.ensGene: Annotation package for TxDb object(s). R package version 3.4.6.

Examples

```
library(TxDb.Dmelanogaster.UCSC.dm6.ensGene)
library(org.Dm.eg.db)
set.seed(123)
example_regions <- random_regions()
txdb <- TxDb.Dmelanogaster.UCSC.dm6.ensGene
genes <- collateGenes(genes = txdb, regions = example_regions, org.Db = org.Dm.eg.db)
head(genes)
```

---

countBamInGATC *Obtain region counts for BAM files*

Description

'countBamInGATC()' obtains the raw counts for the regions between GATC sites, from indexed BAM files specified in the path.

Usage

```
countBamInGATC(path_to_bams, regions, nthreads = 2, ...)
```

Arguments

path_to_bams	A string identifying the directory containing the BAM files.
regions	A GRanges object of GATC regions. The GATC regions can be made with 'gatc_track()'.
nthreads	The number of computer cores to be used to parallelise the function and decrease it's run time. If not specified, will use default (2 cores). * If computer is being used for multiple tasks at once, we recommend reducing the number of cores - or leave it at the default setting. * The number of available cores can be checked using [parallel::detectCores()]
...	Other arguments passed onto 'Rsubread::featureCounts()'

Value

A ‘data.frame‘ containing the GATC region information in the form in the columns: seqnames (chromosome), start, end, width, and strand. The count information for the BAM files is in the subsequent columns, named by the name of the BAM file. * The ".bam" extension is retained in the sample name as an identifier for the sample columns * If necessary, at this stage please rearrange the BAM file columns so they are ordered in the following way: Dam_1, Fusion_1, Dam_2, Fusion_2 etc * The DamID data captures the ~75bp region extending from each GATC site, so although regions are of differing widths, there is a null to minimal length bias present on the data, and does not require length correction.

References

Liao Y, Smyth GK, Shi W (2019). “The R package Rsubread is easier, faster, cheaper and better for alignment and quantification of RNA sequencing reads.” Nucleic Acids Research, 47, e47. doi:10.1093/nar/gkz114. Morgan M, Pagès H, Obenchain V, Hayden N (2024). Rsamtools: Binary alignment (BAM), FASTA, variant call (BCF), and tabix file import. R package version 2.19.3, https://bioconductor.org/packages/Rsamtools.

Examples

```
path_to_bams <- system.file("extdata", package = "Damsel")
example_regions <- random_regions()
counts.df <- countBamInGATC(path_to_bams,
    regions = example_regions,
    nthreads = 2
)
head(counts.df)
# rearrange columns of bam files so that: Dam_1, Fusion_1, Dam_2, Fusion_2
```

---

dros_counts *Example Drosophila DamID counts*

Description

A subset of data from the DamID experiment in Vissers et al., (2018), GEO accession GSE120731. Shown are the 2 Dam-only controls, and the 2 Scalloped fusion samples. The samples have the following accessions: * Dam_1: SRR7948872 * Sd_1: SRR7948874 * Dam_2: SRR7948876 * Sd_2: SRR7948877

Usage

```
data("dros_counts")
```

Format

‘dros_counts‘ A data frame with 383,654 rows and 10 columns:

Position Chromosome and start position

seqnames Chromosome name

start, end, width Region information

strand DNA strand

dam_1_SRR7948872.BAM, sd_1_SRR7948874.BAM, dam_2_SRR7948876.BAM, sd_2_SRR7948877.BAM Sample counts

Details

Individual samples were downloaded in fastq format from the SRA portal. Instructions for using ‘pre-fetch‘ to download the accessions and ‘fasterq-dump‘ to extract the files can be found here: https://github.com/ncbi/sra-tools/wiki/08.-prefetch-and-fasterq-dump

As per Vissers et. al., (2018), the fastq files were aligned into Bam files using Rsubread with appropriate settings for single and paired-end files. The Bamfiles were sorted and indexed using Samtools. Alignment: Rsubread ‘buildindex(basename = "dros_ref", reference = "path/to/fasta")‘ For the single end ‘align(index = "dros_ref", readfile1 = "path/SRR7948877.fastq")‘ For the paired ‘align(index = "dros_ref", readfile1 = "path/SRR7948872_1.fastq.gz", path/SRR7948872_2.fastq.gz")‘

The Bam files were then sorted with ‘samtools sort file_in.BAM -o file_out.BAM‘ before being indexed with ‘samtools index file_out.BAM -o file_out.BAM.bai‘

The counts file was made by running ‘countBamInGATC()‘ using the above samples, and a GATC region file made from: ‘getGatcRegions(BSgenome.Dmelanogaster.UCSC.dm6)$regions‘

Source

<https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE120731>

---

geom_dm *Plotting results from differential methylation testing*

Description

‘geom_dm.res.lfc‘ is a ggplot2 layer that visualises the dm_results and logFC across a given region.

Usage

```
geom_dm(dm_results.df, plot.space = 0.1, plot.height = 0.1)
```

Arguments

dm_results.df	A data.frame of differential testing results as outputted from ‘testDmRegions()‘.
plot.space	Specify gap to next plot. Recommend leaving to the default: 0.1.
plot.height	Specify overall height of plot. Recommend leaving to the default: 0.1.

Details

* regions are coloured by dm result: 1, 0, NA (grey for NA) * cannot be plotted by itself, must be added to an existing plot - see examples.

Value

A ‘ggplot_add‘ object.

References

ggcoverage - Visualise and annotate omics coverage with ggplot2. https://github.com/showteeth/ggcoverage/tree/main

See Also

[geom_peak()] [plotCounts()] [geom_genes()] [geom_gatc()] [plotWrap()] [ggplot2::ggplot_add()]

Examples

```
set.seed(123)
counts.df <- random_counts()
dm_results <- random_edgeR_results()
plotCounts(counts.df,
    seqnames = "chr2L",
    start_region = 1,
    end_region = 40000,
    log2_scale = FALSE
) +
    geom_dm(dm_results)
```

---

geom_gatc *Plot for a GATC track*

Description

‘geom_gatc‘ is a ggplot2 layer that visualises the positions of GATC sites across a given region. * cannot be plotted by itself, must be added to an existing ggplot2 object - see examples.

Usage

```
geom_gatc(
  gatc_sites.df = NULL,
  gatc.color = "red",
  gatc.size = 5,
  plot.space = 0.2,
  plot.height = 0.05
)
```

Arguments

gatc_sites.df	A data.frame of positions of GATC sites - can be made from ‘gatc_track()$sites‘.
gatc.color	Specify colour of lines. Default is red.
gatc.size	Specify size of the line. Default is 5.
plot.space	Specify gap to next plot. Recommend leaving to the default: 0.2.
plot.height	Specify overall height of the plot. Recommend leaving to the default: 0.05.

Value

A ‘ggplot_add‘ object.

References

ggcoverage - Visualise and annotate omics coverage with ggplot2. https://github.com/showteeth/ggcoverage/tree/main

See Also

[plotCounts()] [geom_peak()] [geom_dm()] [geom_genes.tx()] [plotWrap()] [ggplot2::ggplot_add()]

Examples

```
set.seed(123)
example_regions <- random_regions()
counts.df <- random_counts()
gatc_sites <- dplyr::mutate(example_regions,
    start = start - 3, end = start + 4, width = end - start + 1
)

plotCounts(counts.df,
    seqnames = "chr2L",
    start_region = 1,
    end_region = 40000,
    log2_scale = FALSE
) +
    geom_gatc(gatc_sites)
# The plots can be layered ------------------------------------------------
```

---

geom_genes_tx *Plotting genes*

Description

‘geom_genes_tx‘ is a ggplot2 layer that visualises the positions of genes across a given region. * cannot be plotted by itself, must be added to an existing ggplot object - see examples.

Usage

```
geom_genes_tx(
  genes.df,
  txdb,
  gene_limits = NULL,
  plot.space = 0.1,
  plot.height = 0.3
)
```

Arguments

genes.df	A data.frame of genes as outputted from 'get_biomart_genes'.
txdb	A TxDb object as from a TxDb package.
gene_limits	Set the height of the transcripts generated by 'ggbio::autoplot()'. Default is NULL. * If the gene is disproportionately large for the plot space, we recommend reducing the size with gene_limits = c(0,2). * If there are a large amount of transcripts present, we recommend increasing the overall limit, example: gene_limits = c(0,11).
plot.space	Specify gap to next plot. Recommend leaving to the default: 0.1.
plot.height	Specify overall height of plot. Recommend leaving to the default: 0.3.

Value

A 'ggplot_add' object.

References

ggcoverage - Visualise and annotate omics coverage with ggplot2. https://github.com/showteeth/ggcoverage/tree/main
Yin T, Cook D, Lawrence M (2012). "ggbio: an R package for extending the grammar of graphics for genomic data." Genome Biology, 13(8), R77.

See Also

[geom_peak()] [geom_dm()] [geom_counts()] [geom_gatc()] [plotWrap()] [ggplot2::ggplot_add()]

Examples

```
library(TxDb.Dmelanogaster.UCSC.dm6.ensGene)
library(org.Dm.eg.db)
set.seed(123)
example_regions <- random_regions()
counts.df <- random_counts()

txdb <- TxDb.Dmelanogaster.UCSC.dm6.ensGene
genes <- collateGenes(txdb, example_regions, org.Dm.eg.db)

plotCounts(counts.df,
    seqnames = "chr2L",
    start_region = 1,
```

```
    end_region = 40000,
    log2_scale = FALSE
) +
    geom_genes_tx(genes, txdb)
```

---

geom_peak *Plotting peaks*

Description

'geom_peak' is a ggplot2 layer that visualises the positions of peaks across a given region. * cannot be plotted by itself, must be added to an existing ggplot object - see examples.

Usage

```
geom_peak(
  peaks.df = NULL,
  peak.label = FALSE,
  peak.color = "black",
  peak.size = 5,
  plot.space = 0.1,
  plot.height = 0.05
)
```

Arguments

peaks.df	A data.frame of peaks as outputted from 'identifyPeaks()'.
peak.label	Specify whether peak_id labels should be added to the plot. Default is FALSE.
peak.color	Specify colour of peak. Default is black.
peak.size	Specify size of rectangle. Default is 5.
plot.space	Specify gap to next plot. Recommend leaving to the default: 0.1.
plot.height	Specify overall height of plot. Recommend leaving to the default: 0.05.

Value

A 'ggplot_add' object.

References

ggcoverage - Visualise and annotate omics coverage with ggplot2. https://github.com/showteeth/ggcoverage/tree/main

See Also

[plotCounts()] [geom_dm()] [geom_genes.tx()] [geom_gatc()] [plotWrap()] [ggplot2::ggplot_add()]

Examples

```
set.seed(123)
counts.df <- random_counts()
dm_results <- random_edgeR_results()
peaks <- identifyPeaks(dm_results)
plotCounts(counts.df,
    seqnames = "chr2L",
    start_region = 1,
    end_region = 40000,
    log2_scale = FALSE
) +
    geom_peak(peaks)

plotCounts(counts.df,
    seqnames = "chr2L",
    start_region = 1,
    end_region = 40000,
    log2_scale = FALSE
) +
    geom_peak(peaks, peak.label = TRUE)
# The plots can be layered ------------------------------------------------
```

`getGatcRegions` *Extract GATC regions*

Description

'getGatcRegions' identifies and extracts the GATC sites and regions from a BSgenome object or a fasta file.

Usage

```
getGatcRegions(object)
```

Arguments

`object`	A BSgenome package OR the path to a FASTA file.

Value

A 'GRangesList' object of two GRanges; regions - providing the coordinates between adjacent GATC sites, and sites - providing the coordinates of the GATC sites.

Examples

```
if (require("BSgenome.Dmelanogaster.UCSC.dm6")) {
   gatc <- getGatcRegions(BSgenome.Dmelanogaster.UCSC.dm6::BSgenome.Dmelanogaster.UCSC.dm6)

    head(gatc$regions)
```

```
    head(gatc$sites)
}
```

`identifyPeaks` *Identify peaks from differentially methylated regions*

Description

'identifyPeaks' aggregates neighbouring differentially methylated regions, identifying 'peaks' where the provided transcription factor is believed to have bound to the DNA. These locations can then be used to identify the potential target genes.

Usage

```
identifyPeaks(dm_results, gap_size = 150)
```

Arguments

`dm_results`	The results from differential testing.
`gap_size`	The maximum gap in base pairs between differentially methylated regions to be 'skipped'. Default is 150

Details

Small unmethylated regions are able to be 'skipped' over and included into peaks through the gap_size parameter, whose default is 150bp. This was selected due to the common approach of 75bp sequencing of DamID from the edges of the fragments. The FDR and logFC for each peak is calculated via the theory of [csaw::getBestTest()] where the 'best' (smallest) p-value in the regions that make up the peak is selected as representative of the peak. The logFC is therefore the corresponding logFC from the FDR.

Value

A data.frame of peaks ranked by p-value.

References

Lun ATL, Smyth GK (2016). "csaw: a Bioconductor package for differential binding analysis of ChIP-seq data using sliding windows." Nucleic Acids Res., 44(5), e45. Lun ATL, Smyth GK (2014). "De novo detection of differentially bound regions for ChIP-seq data using peaks and windows: controlling error rates correctly." Nucleic Acids Res., 42(11), e95.

Examples

```
set.seed(123)
counts.df <- random_counts()
dm_results <- random_edgeR_results()
peaks <- identifyPeaks(dm_results)
peaks
```

makeDGE *Create DGE object for differential testing*

Description

'makeDGE()' sets up the edgeR analysis for visualisation of the samples [limma::plotMDS()], and then for identifying differentially methylated regions [edgeR_results()].

Usage

```
makeDGE(
  counts.df,
  max.width = 10000,
  lib.size = NULL,
  min.cpm = 0.5,
  min.samples = 3,
  include_replicates = TRUE,
  group = NULL,
  design = NULL
)
```

Arguments

counts.df	A data.frame generated from [countBamInGATC]. Ensure that the samples are ordered by (Dam_1.bam, Fusion_1.bam, Dam_2.bam, Fusion_2.bam, ...).
max.width	Remove large regions, default is width of 10,000. We recommend this value as the Dam can methylate GATC sites up to 5kb away from the binding site, generating a total width of 10 kb.
lib.size	Library size for each sample is calculated as the sum across all rows for that sample unless otherwise specified.
min.cpm	Filtering parameter, minimum counts per million (cpm) of each sample. Recommend leaving at default of 0.5.
min.samples	Filtering parameter, minimum number of samples to meet the criteria of keep_a in order to retain the region in the downstream analysis. Default is 3 (assuming 6 samples).
include_replicates	Should replicates be incorporated into the design matrix? Assumes pattern of Dam_1, Fusion_2, Dam_2, Fusion_2. Default is 'TRUE'.
group	Optional parameter to provide your own group definitions. Default is 'NULL' and groups Dam and Fusion samples assuming pattern as Dam, Fusion etc.
design	Optional parameter to provide your own design matrix. See the 'limma' documentation for advice on creating a design matrix.

Value

An object of class 'DGEList'. Refer to [edgeR::¿DGEListClass'] for details

References

Robinson MD, McCarthy DJ, Smyth GK (2010). "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data." Bioinformatics, 26(1), 139-140. doi:10.1093/bioinformatics/btp616. McCarthy DJ, Chen Y, Smyth GK (2012). "Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation." Nucleic Acids Research, 40(10), 4288-4297. doi:10.1093/nar/gks042. Chen Y, Lun ATL, Smyth GK (2016). "From reads to genes to pathways: differential expression analysis of RNA-Seq experiments using Rsubread and the edgeR quasi-likelihood pipeline." F1000Research, 5, 1438. doi:10.12688/f1000research.8987.2. Chen Y, Chen L, Lun ATL, Baldoni P, Smyth GK (2024). "edgeR 4.0: powerful differential analysis of sequencing data with expanded functionality and improved support for small counts and larger datasets." bioRxiv. doi:10.1101/2024.01.21.576131.

See Also

[testDmRegions()] [countBamInGATC()]

Examples

```
counts.df <- random_counts()

makeDGE(counts.df)
```

`plotCorrHeatmap` *Plot correlation heatmap*

Description

'plotCorrHeatmap' plots the correlation of all available BAM files Dam and Fusion, to visualise the similarity between files. * uses the non-parametric "spearman's" correlation.

Usage

```
plotCorrHeatmap(df, method = "spearman")
```

Arguments

`df`	A data.frame of GATC region counts as outputted from [countBamInGatc()].
`method`	The correlation method used. If not specified, will use default of non-parametric spearman's. * Non-parametric methods are recommended as data does not reliably meet the requirements for parametric analysis.

Details

The correlation between Dam_1 and Fusion_1 can be expected to reach ~ 0.7, whereas the correlation between Dam_1 & Dam_3 or Fusion_1 & Fusion_2 would be expected to be closer to ~0.9

Value

A heatmap style plot of the samples, coloured by correlation value. * Colour spectrum is determined from the minimum correlation as the lowest correlation, the median correlation as the midpoint colour, and 1 as the top colour.

Examples

```
counts.df <- random_counts()
plotCorrHeatmap(counts.df, method = "spearman")
```

`plotCounts` *Plot for counts for all samples across a given region*

Description

'plotCounts' plots a ggplot2 object visualising the raw counts from the bam files across a given region. * this can be used as the base layer (set n_col = 1) for additional plot layers (geom_peak.new, geom_gatc, geom_de.res.lfc etc)

Usage

```
plotCounts(
  counts.df,
  seqnames,
  start_region = NULL,
  end_region = NULL,
  layout = c("stacked", "spread"),
  log2_scale = FALSE,
  colours = NULL,
  ...
)
```

Arguments

`counts.df`	A data.frame of counts as outputted from [process_bams()].
`seqnames`	A character string of the chromosome of interest.
`start_region`	A number providing the start of region to plot.
`end_region`	A number providing the end of region to plot.
`layout`	Determines the layout of the plot. Default is "stacked" collapsing the Dam samples into one plot, and the Fusion samples into another. Samples can be plotted separately using "spread".
`log2_scale`	Determines whether or not to display the counts on a log2 scale. Default is FALSE.
`colours`	Specify colours for the replicates.
`...`	Arguments passed to ggplot2

Value

A ‘ggplot2‘ object.

References

ggcoverage - Visualise and annotate omics coverage with ggplot2. https://github.com/showteeth/ggcoverage/tree/main

See Also

[geom_peak()] [geom_dm()] [geom_genes.tx()] [geom_gatc()] [plot_wrap()]

Examples

```
set.seed(123)
counts.df <- random_counts()
plotCounts(counts.df,
    seqnames = "chr2L",
    start_region = 1,
    end_region = 40000,
    layout = "stacked",
    log2_scale = FALSE
)
plotCounts(counts.df,
    seqnames = "chr2L",
    start_region = 1,
    end_region = 40000,
    layout = "spread",
    log2_scale = FALSE
)
# Can use this plot to layer other plots ------------------------------
dm_results <- random_edgeR_results()
plotCounts(counts.df,
    seqnames = "chr2L",
    start_region = 1,
    end_region = 40000,
    log2_scale = FALSE
) +
    geom_dm(dm_results)
```

`plotCountsDistribution`

Plot distribution of counts ‘plotCountsDistribution‘ plots the distribution of counts enabling the comparison of different samples. Can highlight the different library sizes of the samples.

Description

Plot distribution of counts ‘plotCountsDistribution‘ plots the distribution of counts enabling the comparison of different samples. Can highlight the different library sizes of the samples.

Usage

```
plotCountsDistribution(counts.df, constant = 1)
```

Arguments

counts.df	A counts data.frame as outputted from ‘countBamInGatc‘
constant	A numerical offset to avoid log2(0) = -Inf. Default is 1

Value

A ggplot2 density plot

Examples

```
set.seed(123)
counts.df <- random_counts()

plotCountsDistribution(counts.df, 1)
```

---

plotCountsInPeaks *Plotting the*

Description

Plotting the

Usage

```
plotCountsInPeaks(
  counts.df,
  dm_results.df,
  peaks.df,
  position = c("stack", "fill")
)
```

Arguments

counts.df	A counts data.frame as outputted from ‘countBamInGatc‘
dm_results.df	A data.frame of differential testing results as outputted from ‘testDmRegions()‘.
peaks.df	A data.frame of peaks as outputted from ‘identifyPeaks()‘.
position	If the bar plots should be stacked (showing the count), or fill (showing proportion). Default is "stack".

Value

A ggplot2 bar plot

Examples

```
set.seed(123)
counts.df <- random_counts()[1:2,]
dm_results <- random_edgeR_results()[1:2,]
peaks <- identifyPeaks(dm_results)

# stacked plot
plotCountsInPeaks(counts.df, dm_results, peaks, position = "stack")
# filled plot
plotCountsInPeaks(counts.df, dm_results, peaks, position = "fill")
```

`plotGeneOntology` *Plot gene ontology results*

Description

'plotGeneOntology()' plots the top 10 GO terms in a ggplot2 style plot.

Usage

```
plotGeneOntology(signif_results, fdr_threshold = 0.05)
```

Arguments

`signif_results`	The results as outputted from goseq_fn()$signif_results. Selects the top 10 GO terms as default.
`fdr_threshold`	The FDR threshold used for significance in the ontology. Default is 0.05

Details

A dot plot with the FDR on the x-axis, the size of the dot being the number of genes in the GO category, and the colour of the dot being the ontology (Biological Process, Cellular Component, and Molecular Function).

Value

A ggplot2 object

Examples

```
library("TxDb.Dmelanogaster.UCSC.dm6.ensGene")
library("org.Dm.eg.db")
set.seed(123)
example_regions <- random_regions()
peaks <- identifyPeaks(random_edgeR_results())
txdb <- TxDb.Dmelanogaster.UCSC.dm6.ensGene
genes <- collateGenes(genes = txdb, regions = example_regions, org.Db = org.Dm.eg.db)
annotation <- annotatePeaksGenes(peaks, genes, example_regions)$all
```

```
ontology <- testGeneOntology(annotation, genes, example_regions)$signif_results
plotGeneOntology(ontology)
```

---

plotWrap *Wrapper function for plotting*

Description

'plot_wrap' plots all the available plots at once

Usage

```
plotWrap(
  id = NULL,
  seqnames = NULL,
  start_region = NULL,
  end_region = NULL,
  counts.df = NULL,
  dm_results.df = NULL,
  peaks.df = NULL,
  genes.df = NULL,
  txdb = NULL,
  gatc_sites.df = NULL,
  extend_by = 250,
  ...
)
```

Arguments

id	A character vector of peak OR gene identifier(s) if wish to plot in peak/gene centric manner. Default is NULL.
seqnames	A chromosome. Default is NULL.
start_region	A number providing the start of region to plot. Default is NULL.
end_region	A number providing the end of region to plot. Default is NULL.
counts.df	A data.frame of counts as from [process_bams()]. Default is NULL.
dm_results.df	A data.frame of dm results as from [edgeR_results()]. Default is NULL.
peaks.df	A data.frame of peaks as from [aggregate_peaks()]. Default is NULL.
genes.df	A data.frame of genes as from [get_biomart_genes()]. Default is NULL.
txdb	A TxDb object as from a TxDb package. Default is NULL.
gatc_sites.df	A data.frame of gatc sites as from [gatc_track()$sites]. Default is NULL.
extend_by	A number to extend the limits of the provided region by. Default is 250 bp.
...	arguments passed to geom_genes.me. Allows for adjusting of the plot appearance via gene_limits and plot.height if necessary. * Default for gene_limits is NULL. If the gene is disproportionately large for the plot space, we recommend reducing the size with gene_limits = c(0,2)

Value

A ‘ggplot2‘ object - or list of plots if provided multiple peaks/genes

See Also

[geom_peak()] [geom_dm()] [geom_genes()] [geom_gatc()] [plotCounts()]

Examples

```
library("TxDb.Dmelanogaster.UCSC.dm6.ensGene")
library("org.Dm.eg.db")
set.seed(123)
example_regions <- random_regions()
gatc_sites <- dplyr::mutate(example_regions,
    start = start - 3, end = start + 4, width = end - start + 1
)
counts.df <- random_counts()
dm_results <- random_edgeR_results()
peaks <- identifyPeaks(dm_results)

txdb <- TxDb.Dmelanogaster.UCSC.dm6.ensGene
genes <- collateGenes(txdb, example_regions, org.Db = org.Dm.eg.db)

## plot using a peak_id
plotWrap(
    id = peaks[1, ]$peak_id,
    counts.df = counts.df,
    dm_results.df = dm_results,
    peaks.df = peaks,
    gatc_sites.df = gatc_sites,
    genes.df = genes, txdb = txdb
)

## plot using a gene id
plotWrap(
    id = genes[1, ]$ensembl_gene_id,
    counts.df = counts.df,
    dm_results.df = dm_results,
    peaks.df = peaks,
    gatc_sites.df = gatc_sites,
    genes.df = genes, txdb = txdb
)

## plot providing a region
plotWrap(
    seqnames = "chr2L", start_region = 1, end_region = 5000,
    counts.df = counts.df,
    dm_results.df = dm_results,
    peaks.df = peaks,
    gatc_sites.df = gatc_sites,
    genes.df = genes, txdb = txdb
)
```

```
## plot multiple peaks or genes by providing a vector of id's
plotWrap(
    id = peaks[1:2, ]$peak_id,
    counts.df = counts.df,
    dm_results.df = dm_results,
    peaks.df = peaks,
    gatc_sites.df = gatc_sites,
    genes.df = genes, txdb = txdb
)
```

`random_counts` *Create example counts*

Description

Create example counts

Usage

```
random_counts(size = 50)
```

Arguments

`size`	number of rows to create

Value

example data.frame of counts similar to ‘process_bams()‘

Examples

```
head(random_counts(size = 50))
```

`random_edgeR_results` *Create example edgeR results*

Description

Create example edgeR results

Usage

```
random_edgeR_results(size = 50)
```

Arguments

size	number of rows to create

Value

example data.frame of edgeR results, output similar to ‘edgeR_results()‘

Examples

```
head(random_edgeR_results(size = 50))
```

---

random_regions *Create example regions*

Description

Create example regions

Usage

```
random_regions(size = 50)
```

Arguments

size	number of rows to create

Value

example data.frame with output similar to ‘gatc_track()$regions‘

Examples

```
head(random_regions(size = 50))
```

`testDmRegions` *Differential testing*

Description

'testDmRegions' calculates the differential methylation results, identifying which GATC regions have been enriched in the Fusion samples relative to the controls. Refer to the following pages for further details: * [edgeR::glmQLFit()] * [edgeR::glmQLFTest()] * [edgeR::decideTestsDGE()]

Usage

```
testDmRegions(dge, regions, p.value = 0.05, lfc = 1, plot = TRUE)
```

Arguments

`dge`	A DGEList object as outputted from [makeDGE()].
`regions`	A data.frame of GATC regions.
`p.value`	A number between 0 and 1 providing the required false discovery rate (FDR). Default is 0.05.
`lfc`	A number giving the minimum absolute log2-fold-change for significant results. Default is 1.
`plot`	An option to plot the results using edgeR::plotSmear. Default is TRUE.

Value

A 'data.frame' of differential methylation results. Columns are: Position (chromosome-start), seqnames, start, end, width, strand, number (region number), dm (edgeR result: 0,1,NA), logFC, adjust.p, meth_status (No_signal, Upreg, Not_included). If plot=TRUE, will also return a [edgeR::plotSmear()] plot of the results.

References

Robinson MD, McCarthy DJ, Smyth GK (2010). "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data." Bioinformatics, 26(1), 139-140. doi:10.1093/bioinformatics/btp616. McCarthy DJ, Chen Y, Smyth GK (2012). "Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation." Nucleic Acids Research, 40(10), 4288-4297. doi:10.1093/nar/gks042. Chen Y, Lun ATL, Smyth GK (2016). "From reads to genes to pathways: differential expression analysis of RNA-Seq experiments using Rsubread and the edgeR quasi-likelihood pipeline." F1000Research, 5, 1438. doi:10.12688/f1000research.8987.2. Chen Y, Chen L, Lun ATL, Baldoni P, Smyth GK (2024). "edgeR 4.0: powerful differential analysis of sequencing data with expanded functionality and improved support for small counts and larger datasets." bioRxiv. doi:10.1101/2024.01.21.576131.

See Also

[makeDGE()]

Examples

```
set.seed(123)
example_regions <- random_regions()
counts.df <- random_counts()
dge <- makeDGE(counts.df)

dm_results <- testDmRegions(dge, regions = example_regions, p.value = 0.05, lfc = 1)
head(dm_results)
```

`testGeneOntology` *Gene ontology analysis*

Description

'testGeneOntology' identifies the over-represented GO terms from the peak data, correcting for the number of GATC regions matching to each gene.

Usage

```
testGeneOntology(
  annotation,
  genes,
  regions,
  extend_by = 2000,
  fdr_threshold = 0.05,
  bias = NULL
)
```

Arguments

`annotation`	A data.frame of annotated genes and peaks as 'annotate_peaks()$all'.
`genes`	A data.frame of gene data as outputted from 'get_biomart_genes()'.
`regions`	A data.frame of GATC regions.
`extend_by`	A number to extend the start and end of the genes. We recommend leaving to the default of 2000 bp. * This is done to incorporate the acceptable distance of a peak to a gene. * This also allows for consistency across significant and non-significant genes
`fdr_threshold`	The FDR threshold used for significance in the ontology. Default is 0.05
`bias`	Alternatively, the bias can be input by itself.

Value

3 objects * Plot of goodness of fit of model * Data frame of significant GO category results * Probability weights for each gene

References

Young MD, Wakefield MJ, Smyth GK, Oshlack A (2010). "Gene ontology analysis for RNA-seq: accounting for selection bias." Genome Biology, 11, R14.

Examples

```
library(TxDb.Dmelanogaster.UCSC.dm6.ensGene)
library(org.Dm.eg.db)
set.seed(123)
example_regions <- random_regions()
peaks <- identifyPeaks(random_edgeR_results())

txdb <- TxDb.Dmelanogaster.UCSC.dm6.ensGene
genes <- collateGenes(genes = txdb, regions = example_regions, org.Db = org.Dm.eg.db)
annotation <- annotatePeaksGenes(peaks, genes, example_regions)$all

ontology <- testGeneOntology(annotation, genes, example_regions)
ontology$signif_results
ontology$prob_weights
```

%>% *Pipe operator*

Description

See `magrittr::%>%` for details.

Usage

```
lhs %>% rhs
```

Arguments

`lhs`	A value or the magrittr placeholder.
`rhs`	A function call using the magrittr semantics.

Value

The result of calling 'rhs(lhs)'.

Examples

```
random_regions()$seqnames %>% unique()
```

Index

∗ **datasets**
dros_counts, 6
∗ **internal**
%>%, 26
Damsel-package, 3
%>%, *26*, 26

annotatePeaksGenes, 3

collateGenes, 4
countBamInGATC, 5

Damsel *(*Damsel-package*)*, 3
Damsel-package, 3
dros_counts, 6

geom_dm, 7
geom_gatc, 8
geom_genes_tx, 9
geom_peak, 11
getGatcRegions, 12

identifyPeaks, 13

makeDGE, 14

plotCorrHeatmap, 15
plotCounts, 16
plotCountsDistribution, 17
plotCountsInPeaks, 18
plotGeneOntology, 19
plotWrap, 20

random_counts, 22
random_edgeR_results, 22
random_regions, 23

testDmRegions, 24
testGeneOntology, 25