Package: seqRdna (via r-universe)

November 3, 2026

Title Tools for Calculating for rDNA in Sequencing Data

Version 0.99.5

Description Functions for working with rDNA data generated from sequencing experiments. The package includes functions to calculate the relative rDNA copy number from coverage data, as well as functions to calculate the absolute rDNA copy number using methylation data.

Depends R (>= 4.5.0)

Imports methylKit, GenomicRanges, IRanges, S4Vectors, methods, glmmTMB

License MIT + file LICENSE

Encoding UTF-8

URL https://github.com/MuscleOmicsLab/seqRdna

BugReports https://github.com/MuscleOmicsLab/seqRdna/issues

Config/roxygen2/version 8.1.1

Suggests BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

biocViews Software, DNAMethylation, Epigenetics, CopyNumberVariation, Sequencing, Coverage

Config/pak/sysreqs
cmake make libbz2-dev libicu-dev liblzma-dev libxml2-dev libssl-dev xz-utils zlib1g-dev

Repository https://bioc.r-universe.dev

Date/Publication 2026-11-02 12:25:32 UTC

RemoteUrl https://github.com/bioc/seqRdna

RemoteRef HEAD

RemoteSha 056ef164e484158823c2bb3b9d44009fb0dbb388

Contents

absolute_rDNA_CN . . . 2
aggregate_numCs_numTs_region . . . 4
analyze_methylation_regions . . . 5
calculate_aggregate_meth . . . 6
coverage_chr . . . 8
diff_meth . . . 9
diff_meth_region . . . 12
diff_meth_site . . . 13
distribution_log_odd . . . 14
extract_site_data . . . 15
get_rDNA_data . . . 16
get_unique_chr_names . . . 17
relative_rDNA_CN . . . 18
shifting . . . 20
sort_by_rDNA_region . . . 21

Index **24**

`absolute_rDNA_CN` *Calculate Absolute rDNA Copy Number*

Description

This function calculates the **absolute rDNA copy number** for each sample in a 'Meth_object' (e.g., a list of methylation data or coverage data). It compares the coverage of the rDNA region to the coverage of either the autosomes (chromosomes 1-19) or the entire genome (excluding rDNA). The absolute copy number is adjusted for ploidy (diploid or haploid).

Usage

```
absolute_rDNA_CN(
  Meth_object,
  num_samples,
  autosomes = TRUE,
  rDNA_chr,
  diploid = TRUE
)
```

Arguments

`Meth_object`	- A named list of 'methylRaw' objects (from 'methylKit') or data frames, where each element represents a sample and contains columns 'chr' (chromosome name) and 'coverage' (coverage depth).
`num_samples`	- An integer specifying the number of samples in 'Meth_object' to process.

autosomes	- A logical (‘TRUE‘/‘FALSE‘) indicating whether to use **autosomes (chromosomes 1-19)** as the reference for genome coverage. If ‘FALSE‘, the reference is the entire genome **excluding rDNA**. Default: ‘TRUE‘.
rDNA_chr	- A character string specifying the name of the rDNA chromosome (e.g., ‘"chrRDNAm"‘).
diploid	- A logical (‘TRUE‘/‘FALSE‘) indicating whether the genome is **diploid** (2 copies of each chromosome). If ‘TRUE‘, the absolute rDNA copy number is multiplied by 2. Default: ‘TRUE‘.

Value

This function **prints** a tab-separated table to the console with the following columns:

Sample ID: The sample identifier (extracted from ‘Meth_object[[i]]@sample.id‘).

Genome Depth: The mean coverage depth for the reference region (autosomes or genome excluding rDNA).

rDNA Depth: The mean coverage depth for the rDNA region.

absolute rDNA CN: The absolute rDNA copy number, calculated as ‘(cov_rDNA / cov_wgenome) * 2‘ (if diploid) or ‘(cov_rDNA / cov_wgenome)‘ (if haploid).

The function does **not** return a value but prints results directly.

Note

- If ‘autosomes = TRUE‘, the reference coverage (‘cov_wgenome‘) is calculated from chromosomes 1-19. Ensure these chromosomes exist in your data. - If ‘autosomes = FALSE‘, the reference coverage is calculated from **all chromosomes except ‘rDNA_chr‘**. - The function assumes ‘Meth_object‘ is a **named list** and that each element has a ‘@sample.id‘ slot (if using ‘methylRaw‘ objects) or can be accessed via ‘names(Meth_object)[i]‘. - If ‘cov_wgenome‘ is ‘0‘, the function will produce ‘Inf‘ or ‘NA‘ for ‘abs_rDNA_CN‘. Add a check for ‘cov_wgenome > 0‘ if this is a concern. - The function uses ‘cat‘ to print tab-separated output. To capture the output, redirect it to a file or use ‘capture.output().

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
)

loaded_data <- readRDS(file_path)

# Define experimental conditions
# 0 = OLD, 1 = YOUNG
treatment <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1)
num_samples <- length(treatment)
```

```
# Construct a methylRawList
loaded_data <- methylKit::methylRawList(
  loaded_data,
  treatment = treatment
)

# Calculate absolute rDNA copy number
abs_out <- absolute_rDNA_CN(
  Meth_object = loaded_data,
  num_samples = num_samples,
  autosomes = TRUE,
  rDNA_chr = "chrRDNAm",
  diploid = TRUE
)

# Inspect the results
print(abs_out)
```

`aggregate_numCs_numTs_region` *Aggregate methylated and unmethylated counts within a genomic region*

Description

Sums numCs and numTs across sites within the specified region for each sample, grouped by condition.

Usage

```
aggregate_numCs_numTs_region(region_res, region, conditions)
```

Arguments

`region_res`	A nested list where each top-level element is a sample (e.g., `region_res[["SRR28431752"]][["18S"]][`
`region`	A string specifying the region of interest (e.g., "18S").
`conditions`	A named list where each name is a condition (e.g., OLD or YOUNG) and each element is a character vector of sample IDs belonging to that condition.

Value

An `S4Vectors::DataFrame` with columns:

Sample_ID The sample identifier.

Condition The condition to which the sample belongs.

aggregate_numCs The sum of methylated counts within the region, excluding missing values.

aggregate_numTs The sum of unmethylated counts within the region, excluding missing values.

analyze_methylation_regions
Analyze Methylation Across Genomic Regions

Description

This function analyzes differential methylation across specified genomic regions for a given set of samples. It uses an external script (‘diff_meth_region.R‘) to compute methylation differences, p-values, and adjusted p-values for each region. Results are combined into a single data frame with multiple-testing correction applied.

Usage

```
analyze_methylation_regions(
  region_res,
  samples,
  model_type = "beta-binomial",
  p_adjust_method = "BH"
)
```

Arguments

region_res	A nested list where each top-level element corresponds to a sample, and each sub-element corresponds to a genomic region (e.g., ‘region_res[[1]]$18S‘). Each region must contain methylation data (e.g., ‘numCs‘, ‘coverage‘).
samples	A vector of sample IDs to include in the analysis.
model_type	A character string specifying the statistical model to use for differential methylation analysis. Default is ‘"beta-binomial"‘.
p_adjust_method	A character string specifying the method for p-value adjustment. Default is ‘"BH"‘ (Benjamini-Hochberg).

Value

A data frame with columns: - ‘region‘: Name of the genomic region. - ‘beta_0‘: Intercept term from the model. - ‘beta_1‘: Coefficient for the methylation difference. - ‘p_value‘: Raw p-value for the methylation difference. - ‘p_adjust‘: Adjusted p-value after multiple-testing correction. - ‘meth_diff‘: Estimated methylation difference between conditions.

`calculate_aggregate_meth`

Computes the aggregate methylation for a specified genomic region across samples, grouped by their respective conditions.

Description

Computes the aggregate methylation for a specified genomic region across samples, grouped by their respective conditions.

Usage

```
calculate_aggregate_meth(region_res, region, conditions, med = FALSE)
```

Arguments

`region_res`	A nested list where each top-level element is a sample (e.g., 'region_res$SRR28431752$18S$numCs').
`region`	A string specifying the region of interest (e.g., '"18S"').
`conditions`	A named list where each name is a condition (e.g., 'CONTROL', 'CONDITION 1') and each element is a vector of sample IDs belonging to that condition.
`med`	A logical value indicating whether to compute the median (if 'TRUE') or the mean (if 'FALSE') for the aggregate methylation values. Default is 'FALSE'.

Value

A dataframe with columns:

sample_ID: The sample identifier.

condition: The condition to which the sample belongs.

aggregate_meth: The aggregate methylation value for the specified region.

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
)

loaded_data <- readRDS(file_path)

# Define the experimental groups
samples <- list(
  OLD = c(
    "SRR28431752", "SRR28431753", "SRR28431754",
```

```
    "SRR28431755", "SRR28431756", "SRR28431757"
  ),
  YOUNG = c(
    "SRR28431748", "SRR28431749",
    "SRR28431750", "SRR28431751"
  )
)

treatment <- c(
  0, 0, 0, 0, 0, 0,
  1, 1, 1, 1
)

num_samples <- length(treatment)

# Construct a methylRawList object
loaded_data <- methods::new(
  "methylRawList",
  loaded_data,
  treatment = treatment
)

# Extract rDNA methylation data
meth_rDNA <- get_rDNA_data(
  loaded_data,
  num_samples,
  rDNA_chr_name = "chrRDNAm"
)

# Adjust genomic coordinates
meth_rDNA_shifted <- shifting(
  meth_rdna_raw = meth_rDNA,
  cutoff = 35000,
  treatment = treatment
)

# Separate methylation data into rDNA regions
region_res <- sort_by_rDNA_region(
  meth_rDNA_shifted,
  num_samples,
  species = "mm39",
  min_cov = 10
)

# Calculate mean methylation for the 18S region
mean_meth <- calculate_aggregate_meth(
  region_res,
  region = "18S",
  conditions = samples,
  med = FALSE
)

# Calculate median methylation for the 18S region
```

```
median_meth <- calculate_aggregate_meth(
  region_res,
  region = "18S",
  conditions = samples,
  med = TRUE
)

# Inspect the results
print(mean_meth)
print(median_meth)
```

`coverage_chr` *Plot Average Coverage per Chromosome*

Description

Calculates the average coverage for each chromosome across all samples and plots it as a bar chart.

Usage

```
coverage_chr(meth_object, num_samples, chromosome_names)
```

Arguments

`meth_object`	A list of data frames or ‘methylRaw‘ objects containing methylation data.
`num_samples`	Integer. Number of samples to process.
`chromosome_names`	Character vector. Names of chromosomes to include in the analysis.

Value

A ‘ggplot‘ bar chart visualizing the average coverage per chromosome.

Note

- The function assumes the input data frames contain a ‘coverage‘ column. - The plot uses ‘ggplot2‘ for visualization.

Examples

```
# Load the example methylation dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
)
```

```
loaded_data <- readRDS(file_path)

# Define the number of samples
treatment <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1)
num_samples <- length(treatment)

# Extract unique chromosome names
chromosome_names <- get_unique_chr_names(
  loaded_data,
  num_samples
)

# Display the chromosome names
print(chromosome_names)
```

`diff_meth` *Differential Methylation Analysis of Ribosomal DNA*

Description

Performs differential methylation analysis of ribosomal DNA (rDNA) at either individual CpG sites or across rDNA regions. The function acts as a wrapper for the site-level and region-level differential methylation functions.

Three regression models are supported: binomial logistic regression, quasi-binomial logistic regression, and beta-binomial regression. The latter two approaches account for overdispersion, which may arise from biological variability and the repetitive nature of rDNA.

Usage

```
diff_meth(
  meth_data,
  samples,
  species,
  num_samples,
  min_cov,
  region,
  model_type,
  p_adjust_method,
  site = TRUE
)
```

Arguments

`meth_data`	A methylKit methylation object containing methylation data for the samples to be analysed.
`samples`	A vector specifying the experimental condition for each sample, in the same order as the samples in ‘meth_data‘.

`species` A character string specifying the reference genome used to identify rDNA regions (e.g., '"mm39"').

`num_samples` An integer specifying the number of samples included in the analysis.

`min_cov` A positive integer specifying the minimum sequencing coverage required for a CpG site to be included in the analysis.

`region` A character string specifying the rDNA region of interest for site-level analysis (e.g., '"18S"', '"5.8S"', '"28S"', or '"45S"').

`model_type` A character string specifying the regression model:

'"binomial"' Binomial logistic regression fitted using 'stats::glm()'.

'"quasi-binomial"' Quasi-binomial logistic regression fitted using 'stats::glm()', allowing for overdispersion.

'"beta-binomial"' Beta-binomial regression fitted using 'glmmTMB::glmmTMB()', accounting for overdispersion through a beta-binomial distribution.

`p_adjust_method`

A character string specifying the method used to adjust p-values for multiple testing. Use '"BH"' for the Benjamini-Hochberg false discovery rate correction.

`site` Logical. If 'TRUE' (default), differential methylation is performed independently at each CpG site within the specified rDNA region. If 'FALSE', region-level differential methylation analysis is performed.

Details

The function first organises the methylation data into rDNA regions using 'sort_by_rDNA_region()', applying the specified minimum coverage threshold.

When 'site = TRUE', methylation counts are extracted for individual CpG sites within the selected region using 'extract_site_data()'. Differential methylation is then calculated using 'diff_meth_site()'.

When 'site = FALSE', differential methylation is calculated across rDNA regions using 'analyze_methylation_regions()'. In this case, the 'region' argument is not passed to the analysis function.

The selected regression model is used to assess differences in methylation between experimental conditions. P-values are adjusted using the specified multiple-testing correction method.

Value

Returns the results of the selected differential methylation analysis. For site-level analysis, the output contains results for individual CpG positions, including regression coefficients, odds ratios, p-values and adjusted p-values, where estimable. For region-level analysis, the output contains the results generated by 'analyze_methylation_regions()'.

See Also

[sort_by_rDNA_region()], [extract_site_data()], [diff_meth_site()], [analyze_methylation_regions()], [stats::glm()], [stats::p.adjust()], [glmmTMB::glmmTMB()]

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
)

loaded_data <- readRDS(file_path)

# Define experimental conditions
# 0 = OLD, 1 = YOUNG
treatment <- c(
  0, 0, 0, 0, 0, 0,
  1, 1, 1, 1
)

num_samples <- length(treatment)

# Define sample groups
samples <- list(
  OLD = c(
    "SRR28431752", "SRR28431753", "SRR28431754",
    "SRR28431755", "SRR28431756", "SRR28431757"
  ),
  YOUNG = c(
    "SRR28431748", "SRR28431749",
    "SRR28431750", "SRR28431751"
  )
)

# Construct a methylRawList object
loaded_data <- methods::new(
  "methylRawList",
  loaded_data,
  treatment = treatment
)

# Extract rDNA methylation data
meth_rDNA <- get_rDNA_data(
  loaded_data,
  num_samples,
  rDNA_chr_name = "chrRDNAm"
)

# Shift rDNA genomic coordinates
meth_rDNA_shifted <- shifting(
  meth_rdna_raw = meth_rDNA,
  cutoff = 35000,
  treatment = treatment
)
```

```
# Site-level differential methylation analysis
site_results <- diff_meth(
  meth_data = meth_rDNA_shifted,
  samples = samples,
  species = "mm39",
  num_samples = num_samples,
  min_cov = 10,
  region = "18S",
  model_type = "binomial",
  p_adjust_method = "BH",
  site = TRUE
)

head(site_results)

# Region-level differential methylation analysis
region_results <- diff_meth(
  meth_data = meth_rDNA_shifted,
  samples = samples,
  species = "mm39",
  num_samples = num_samples,
  min_cov = 10,
  region = "18S",
  model_type = "binomial",
  p_adjust_method = "BH",
  site = FALSE
)

head(region_results)
```

`diff_meth_region` *Perform Differential Methylation Analysis for a Genomic Region*

Description

Aggregates methylation counts (‘numCs‘ and ‘numTs‘) for a specified genomic region, fits a logistic regression model (binomial, quasi-binomial, or beta-binomial), and computes methylation differences, p-values, and predicted probabilities.

Usage

```
diff_meth_region(
  region_res,
  region,
  samples,
  model_type = "beta-binomial",
  p_adjust_method = "BH"
)
```

Arguments

region_res	A nested list where each top-level element corresponds to a sample, and each sub-element corresponds to a genomic region (e.g., 'region_res[[1]]$18S'). Each region must contain methylation data (e.g., 'numCs', 'numTs').
region	A character string specifying the name of the genomic region to analyze.
samples	A named vector of sample IDs, where names correspond to conditions (e.g., 'c(Young = c("sample1", "sample2"), Old = c("sample3", "sample4"))').
model_type	A character string specifying the statistical model to use. Options: '"binomial"', '"quasi-binomial"', or '"beta-binomial"' (default).
p_adjust_method	A character string specifying the method for p-value adjustment. Default is '"BH"' (Benjamini-Hochberg). Note: Adjustment is applied in the wrapper function.

Value

A data frame with columns: - 'beta_0': Intercept term from the model. - 'beta_1': Coefficient for the methylation difference. - 'p_value': Raw p-value for the methylation difference. - 'meth_diff': Estimated methylation difference between conditions (in percentage points).

---

diff_meth_site *Perform Differential Methylation Analysis at Individual Sites*

Description

This function analyzes differential methylation at individual genomic positions. It fits a logistic regression model (binomial, quasi-binomial, or beta-binomial) for each position, computes odds ratios, p-values, and predicted methylation probabilities,and applies multiple-testing correction. Positions with insufficient data are skipped.

Usage

```
diff_meth_site(all_data, model_type = "beta-binomial", p_adjust_method = "BH")
```

Arguments

all_data	A data frame containing methylation data with columns: - 'position': Genomic position. - 'numCs': Number of methylated reads. - 'numTs': Number of unmethylated reads. - 'Group': Binary treatment variable (e.g., 0 for control, 1 for treatment).
model_type	A character string specifying the statistical model to use. Options: '"beta-binomial"' (default), '"quasi-binomial"', or '"binomial"'.
p_adjust_method	A character string specifying the method for p-value adjustment. Default is '"BH"' (Benjamini-Hochberg).

Value

A data frame with columns: - ‘position‘: Genomic position. - ‘beta_0‘: Intercept term from the model. - ‘beta_1‘: Coefficient for the methylation difference. - ‘odds_ratio‘: Odds ratio for the methylation difference. - ‘p_value‘: Raw p-value for the methylation difference. - ‘p_adjust‘: Adjusted p-value after multiple-testing correction. - ‘p_cond_1‘: Predicted methylation probability for the control group. - ‘p_cond_2‘: Predicted methylation probability for the treatment group. - ‘meth_diff‘: Methylation difference between groups (in percentage points). - ‘log10_p_adjust‘: Negative log10 of the adjusted p-value.

`distribution_log_odd` *Calculate Log-Odds of Methylation for a Chromosome*

Description

Computes the log-odds of methylation (log(numCs/numTs)) for each CpG site in a specified chromosome across samples. Useful for assessing the suitability of logistic regression models (e.g., in ‘methylKit‘ or custom analyses).

Usage

```
distribution_log_odd(meth_object, treatment, chromo)
```

Arguments

`meth_object`	A list of ‘methylRaw‘ objects (from ‘methylKit‘), where each object contains methylation data for a sample.
`treatment`	A vector of treatment/condition labels for each sample. Must have the same length as ‘meth_object‘.
`chromo`	A character string specifying the chromosome to analyze.

Value

A data frame with columns: - ‘log_odds‘: Log-odds of methylation (log(numCs/numTs)) for each CpG site. - ‘sample_id‘: Identifier for the sample.

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
)

loaded_data <- readRDS(file_path)
```

```
# Define experimental conditions
# 0 = OLD, 1 = YOUNG
treatment <- c(
  0, 0, 0, 0, 0, 0,
  1, 1, 1, 1
)

# Calculate log-odds for CpG sites on the rDNA chromosome
log_odds <- distribution_log_odd(
  meth_object = loaded_data,
  treatment = treatment,
  chromo = "chrRDNAm"
)

# Display the first few results
head(log_odds)
```

`extract_site_data` *Extract Methylation Site Data for a Genomic Region*

Description

Extracts methylation counts (‘numCs‘, ‘numTs‘) and positions for a specified genomic region across all samples. Assigns a binary treatment variable (‘Group‘) based on the first group in ‘samples‘. Skips samples with mismatched vector lengths and removes zero-coverage sites.

Usage

```
extract_site_data(region_res, samples, region = "18S")
```

Arguments

`region_res`	A nested list where each top-level element corresponds to a sample, and each sub-element corresponds to a genomic region (e.g., ‘region_res[[1]]$18S‘).
`samples`	A named list of sample IDs grouped by condition (e.g., ‘list(OLD = c("sample1", "sample2"), YOUNG = c("sample3"))‘).
`region`	A character string specifying the genomic region to extract. Default is ‘"18S"‘.

Value

A data frame with columns: - ‘sample‘: Sample identifier. - ‘position‘: Genomic position. - ‘numCs‘: Number of methylated reads. - ‘numTs‘: Number of unmethylated reads. - ‘Group‘: Binary treatment variable (0 for control group, 1 otherwise).

get_rDNA_data *Extract rDNA Chromosome Data*

Description

Extracts all data associated with the rDNA chromosome from a ‘meth_object‘.

Usage

```
get_rDNA_data(meth_object, num_samples, rDNA_chr_name = "rDNAm")
```

Arguments

meth_object	A methylation object (e.g., from ‘methylKit‘).
num_samples	Number of samples in ‘meth_object‘ to process.
rDNA_chr_name	Name of the rDNA chromosome (default: ‘"rDNAm"‘).

Value

A subset of ‘meth_object‘ containing only the rDNA chromosome data.

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
)

loaded_data <- readRDS(file_path)

# Define experimental conditions
# 0 = OLD, 1 = YOUNG
treatment <- c(
  0, 0, 0, 0, 0, 0,
  1, 1, 1, 1
)

num_samples <- length(treatment)

# Construct a methylRawList object
loaded_data <- methylKit::methylRawList(
  loaded_data,
  treatment = treatment
)

# Extract methylation data for the rDNA chromosome
```

```
meth_rDNA <- get_rDNA_data(
  meth_object = loaded_data,
  num_samples = num_samples,
  rDNA_chr_name = "chrRDNAm"
)

# Inspect the extracted data
print(meth_rDNA)
```

`get_unique_chr_names` *Get Unique Chromosome Names and Check Consistency*

Description

This function extracts the unique chromosome names from the first sample in a 'Meth_object' (e.g., a list of methylation data) and checks if all samples have the same chromosomes. If inconsistencies are found, it raises an error and lists the problematic samples.

Usage

```
get_unique_chr_names(Meth_object, num_samples)
```

Arguments

`Meth_object`	A named list of data frames or objects (e.g., 'meth_raw'),where each element represents a sample and contains a 'chr' column with chromosome names.
`num_samples`	An integer specifying the number of samples in 'Meth_object' to check.

Value

A character vector of unique chromosome names from the first sample, if all samples are consistent. If inconsistencies are found, the function stops execution and raises an error.

Note

- The function assumes 'Meth_object' is a **named list** (so 'names(Meth_object)[i]' returns the sample name). - If 'Meth_object' is unnamed, replace 'names(meth_raw)[i]' with a custom identifier (e.g., 'paste("Sample", i)'). - The function uses 'setequal()' to ensure exact matching of chromosome sets.

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
```

```
)

loaded_data <- readRDS(file_path)

# Define experimental conditions
# 0 = OLD, 1 = YOUNG
treatment <- c(
  0, 0, 0, 0, 0, 0,
  1, 1, 1, 1
)

num_samples <- length(treatment)

# Extract unique chromosome names
chromosome_names <- get_unique_chr_names(
  loaded_data,
  num_samples
)

# Display the results
print(chromosome_names)
```

`relative_rDNA_CN` *Calculate Relative rDNA Copy Number*

Description

This function calculates the **relative rDNA copy number** for each sample in a ‘Meth_object‘ (e.g., a list of methylation data or coverage data). It compares the **total coverage** of the rDNA region to the **total coverage** of either the autosomes (chromosomes 1-19) or the entire genome. The relative copy number is adjusted for ploidy (diploid or haploid).

Usage

```
relative_rDNA_CN(
  Meth_object,
  num_samples,
  autosomes = TRUE,
  rDNA_chr,
  diploid = TRUE
)
```

Arguments

`Meth_object`	A named list of ‘methylRaw‘ objects (from ‘methylKit‘) or data frames, where each element represents a sample and contains columns ‘chr‘ (chromosome name)‘ and ‘coverage‘ (coverage depth).

num_samples	An integer specifying the number of samples in ‘Meth_object‘ to process.
autosomes	A logical (‘TRUE‘/‘FALSE‘) indicating whether to use **autosomes (chromosomes 1-19)** as the reference for genome coverage. If ‘FALSE‘, the reference is the **entire genome**. Default: ‘TRUE‘.
rDNA_chr	A character string specifying the name of the rDNA chromosome (e.g., ‘"chrRDNAm"‘).
diploid	A logical (‘TRUE‘/‘FALSE‘) indicating whether the genome is **diploid** (2 copies of each chromosome). If ‘TRUE‘, the relative rDNA copy number is multiplied by 2. Default: ‘TRUE‘.

Value

An S4Vectors::DataFrame containing Sample_ID, Total_Number_of_Reads, Number_of_Reads_rDNA, and Relative_rDNA_CN. The read-count columns contain summed coverage across recorded sites.

Note

- If ‘autosomes = TRUE‘, the reference coverage (‘sum_cov_wgenome‘) is calculated from **all chromosomes** in the sample. This is a **deviation from the function name**, as it does not restrict to autosomes (chromosomes 1-19). To fix this, update the filtering logic for ‘chr_data‘ to include only autosomes (e.g., ‘chr_data <- Meth_object[[i]][Meth_object[[i]]$chr‘ ‘ - If ‘autosomes = FALSE‘, the reference coverage is calculated from **all chromosomes** in the sample. - The function assumes ‘Meth_object‘ is a **named list** and that each element has a ‘@sample.id‘ slot (if using ‘methylRaw‘ objects) or can be accessed via ‘names(Meth_object)[i]‘. - If ‘sum_cov_wgenome‘ is ‘0‘, the function will produce ‘Inf‘ or ‘NA‘ for ‘rel_rDNA_CN‘. Add a check for ‘sum_cov_wgenome > 0‘ if this is a concern. - The function uses ‘cat‘ to print tab-separated output. To capture the output, redirect it to a file or use ‘capture.output()‘.

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
)

loaded_data <- readRDS(file_path)

# Define experimental conditions
# 0 = OLD, 1 = YOUNG
treatment <- c(
  0, 0, 0, 0, 0, 0,
  1, 1, 1, 1
)

num_samples <- length(treatment)

# Calculate relative rDNA copy number
```

```
rel_out <- relative_rDNA_CN(
  Meth_object = loaded_data,
  num_samples = num_samples,
  autosomes = TRUE,
  rDNA_chr = "chrRDNAm",
  diploid = TRUE
)

# Display the results
print(rel_out)
```

---

shifting *Shift rDNA Methylation Data*

Description

This function shifts the start and end positions of rDNA methylation data by a calculated shift value, ensuring all coordinates remain non-negative. The shift is computed as the difference between the maximum position in the data and a user-specified cutoff. Invalid coordinates (negative after shifting)are filtered out.

Usage

```
shifting(meth_rdna_raw, cutoff, treatment)
```

Arguments

meth_rdna_raw	A list of 'methylRaw' objects (from 'methylKit'), where each object contains methylation data with 'start' and 'end' positions.
cutoff	A numeric value used to compute the shift. The shift is calculated as 'max_position - cutoff'.
treatment	A character vector or factor specifying the treatment group for each sample. Used to label the output 'methylRawList'.

Value

A 'methylRawList' object containing the shifted methylation data, with invalid coordinates (negative after shifting) removed.

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
```

```
  mustWork = TRUE
)

loaded_data <- readRDS(file_path)

# Define experimental conditions
# 0 = OLD, 1 = YOUNG
treatment <- c(
  0, 0, 0, 0, 0, 0,
  1, 1, 1, 1
)

num_samples <- length(treatment)

# Define sample groups
samples <- list(
  OLD = c(
    "SRR28431752", "SRR28431753", "SRR28431754",
    "SRR28431755", "SRR28431756", "SRR28431757"
  ),
  YOUNG = c(
    "SRR28431748", "SRR28431749",
    "SRR28431750", "SRR28431751"
  )
)

# Construct a methylRawList object
loaded_data <- methods::new(
  "methylRawList",
  loaded_data,
  treatment = treatment
)

# Extract rDNA methylation data
meth_rDNA <- get_rDNA_data(
  loaded_data,
  num_samples,
  rDNA_chr_name = "chrRDNAm"
)

# Shift rDNA genomic coordinates
meth_rDNA_shifted <- shifting(
  meth_rdna_raw = meth_rDNA,
  cutoff = 35000,
  treatment = treatment
)

head(meth_rDNA_shifted)
```

`sort_by_rDNA_region` *Sort Methylation Data by rDNA Regions*

Description

Extracts and organizes methylation data for specified rDNA regions across samples. Supports predefined species (‘mm39‘, ‘rno7‘). Validates sample bounds and required columns.

Usage

```
sort_by_rDNA_region(
  rDNA_meth_object,
  num_samples,
  species = "mm39",
  min_cov = 10
)
```

Arguments

rDNA_meth_object	A list of ‘methylRaw‘ objects (from ‘methylKit‘) or data frames.
num_samples	Integer. Number of samples to process (must be >= 1 and <= ‘length(rDNA_meth_object)‘).
species	Character. Species reference for rDNA regions: ‘"mm39"‘ (default), ‘"rno7"‘, or ‘"hg38"‘ (unsupported).
min_cov	Numeric. Minimum coverage threshold to filter data (default: ‘10‘).

Value

A named list of lists, where each sublist contains: - ‘numCs‘: Number of methylated Cs in the region. - ‘numTs‘: Number of unmethylated Ts in the region. - ‘coverage‘: Coverage depth for the region. - ‘position‘: Start positions of the region. Structured as ‘sample_id -> region -> data‘.

Note

- For ‘species = "hg38"‘, the function stops with a warning (unsupported). - Missing columns in input data will cause an error.

Examples

```
# Load the example mouse WGBS dataset
file_path <- system.file(
  "extdata",
  "filtered_meth_raw_chr1_chrRDNAm_small.rds",
  package = "seqRdna",
  mustWork = TRUE
)

loaded_data <- readRDS(file_path)

# Define experimental conditions
# 0 = OLD, 1 = YOUNG
treatment <- c(
  0, 0, 0, 0, 0, 0,
```

```
  1, 1, 1, 1
)

num_samples <- length(treatment)

# Construct a methylRawList object
loaded_data <- methods::new(
  "methylRawList",
  loaded_data,
  treatment = treatment
)

# Extract methylation data from the rDNA chromosome
meth_rDNA <- get_rDNA_data(
  loaded_data,
  num_samples,
  rDNA_chr_name = "chrRDNAm"
)

# Separate rDNA methylation data into annotated regions
region_res <- sort_by_rDNA_region(
  meth_rDNA,
  num_samples,
  species = "mm39",
  min_cov = 10
)

# Inspect the regions available for the first sample
names(region_res[[1]])
```

Index

absolute_rDNA_CN, 2
aggregate_numCs_numTs_region, 4
analyze_methylation_regions, 5

calculate_aggregate_meth, 6
coverage_chr, 8

diff_meth, 9
diff_meth_region, 12
diff_meth_site, 13
distribution_log_odd, 14

extract_site_data, 15

get_rDNA_data, 16
get_unique_chr_names, 17

relative_rDNA_CN, 18

shifting, 20
sort_by_rDNA_region, 21