Package 'cfDNAPro'

November 11, 2026

Type Package

Title cfDNAPro extracts and Visualises biological features from whole genome sequencing data of cell-free DNA

Version 1.18.0

biocViews Visualization, Sequencing, WholeGenome

Description cfDNA fragments carry important features for building cancer sample classification ML models, such as fragment size, and fragment end motif etc.
Analyzing and visualizing fragment size metrics, as well as other biological features in a curated, standardized, scalable, well-documented, and reproducible way might be time intensive.
This package intends to resolve these problems and simplify the process. It offers two sets of functions for cfDNA feature characterization and visualization.

Depends R (>= 4.1.0), magrittr (>= 1.5.0)

Imports tibble, GenomicAlignments, IRanges, plyranges, GenomeInfoDb, GenomicRanges, BiocGenerics, stats, utils, dplyr (>= 0.8.3), stringr (>= 1.4.0), quantmod (>= 0.4), ggplot2 (>= 3.2.1), Rsamtools (>= 2.4.0), rlang (>= 0.4.0), BSgenome.Hsapiens.UCSC.hg38, BSgenome.Hsapiens.UCSC.hg19, BSgenome.Hsapiens.NCBI.GRCh38

Suggests scales, ggpubr, knitr (>= 1.23), rmarkdown (>= 1.14), devtools (>= 2.3.0), BiocStyle, testthat

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.2

VignetteBuilder knitr

URL https://github.com/hw538/cfDNAPro

BugRePORTS https://github.com/hw538/cfDNAPro/issues

BiocType Software

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

git_branch RELEASE_3_23

git_last_commit 354165a

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Haichao Wang [aut, cre],
Hui Zhao [ctb],
Elkie Chan [ctb],
Christopher Smith [ctb],
Tomer Kaplan [ctb],
Florian Markowetz [ctb],
Nitzan Rosenfeld [ctb]

Maintainer Haichao Wang <hw538@cam.ac.uk>

Contents

callMetrics 2
callMode 3
callPeakDistance 4
callSize 6
callValleyDistance 7
examplePath 8
plotAllToOne 8
plotMetrics 9
plotMode 10
plotModeSummary 11
plotPeakDistance 12
plotSingleGroup 13
plotValleyDistance 14
readBam 15
read_bam_insert_metrics 17

Index **19**

`callMetrics` *Calculate the metrics of insert size*

Description

Calculate the metrics of insert size

Usage

```
callMetrics(
  path = getwd(),
  groups,
  fun = "all",
  outfmt = "df",
```

```
  input_type,
  ...
)
```

Arguments

path	The root folder containing all groups folders, default is the present working folder.
groups	The name of the groups, the input value should be vector, e.g. groups=c('group1','group2'), default is all sub-folders in the 'path'.
fun	String value, the types of metrics to be calculated. Default is 'all', which means both median and mean values will be returned.
outfmt	The output format, a 'list' or 'dataframe' or 'df', default is dataframe.
input_type	Character. The input file format, should be one of these: 'picard', 'bam' or 'cfdnapro'. The bam files has to be marked duplicates.
...	Further arguments passed to or from other methods.

Value

The inter valley distance in list or dataframe format.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the metrics.
df <- callMetrics(path = path)
```

---

callMode *Calculate the mode fragment size of each sample*

Description

Calculate the mode fragment size of each sample

Usage

```
callMode(
  path,
  groups,
  outfmt = "df",
  order = groups,
  summary,
```

```
  mincount,
  input_type,
  ...
)
```

Arguments

path	The root folder containing all groups folders, default is the present working folder.
groups	The name of the groups, the input value should be vector, e.g. groups=c('group1','group2'), default is all folders in the folder path.
outfmt	The output format, 'list' or 'dataframe' or 'df', default is dataframe.
order	The order in the sorted output, default value equals to 'groups' parameter.
summary	Summarize the dataframe result by calculating each mode size and its count number. Default value is False.
mincount	Minimum count number of each mode size in the summarized output. Only significant when 'summary = TRUE'.
input_type	Character. The input file format, should be one of these: 'picard', 'bam', 'cfd-napro'. The bam files has to be marked duplicates.
...	Further arguments passed to or from other methods.

Value

The function returns the inter valley distance in list or dataframe format.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the mode.
df <- callMode(path = path)
```

---

callPeakDistance *Calculate the inter-peak distance of insert size*

Description

Calculate the inter-peak distance of insert size

Usage

```
callPeakDistance(
  path = getwd(),
  groups,
  limit,
  outfmt,
  summary,
  mincount,
  input_type,
  ...
)
```

Arguments

path	The root folder containing all groups folders. Default is the present working folder.
groups	The name of the groups, the input value should be vector, e.g. groups=c('group1','group2'). Default is all folders in the folder path.
limit	The insert size range that will be focused on. Default value is 'limit = c(35,135)'.
outfmt	The output format, a 'list' or 'dataframe'. Default is dataframe.
summary	If TRUE, summarize the output.
mincount	The minimum count value of inter-peak distance in the summary.
input_type	Character. The input file format, should be one of these: 'picard', 'bam', 'cfd-napro'. The bam files has to be marked duplicates.
...	Further arguments passed to or from other methods.

Value

The function returns the inter peak distance in list or dataframe format.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the inter-peak distance.
df <- callPeakDistance(path = path)
```

callSize *Calculate the insert size metrics (i.e. prop, cdf, 1-cdf) or each group*

Description

Calculate the insert size metrics (i.e. prop, cdf, 1-cdf) or each group

Usage

```
callSize(path, groups, outfmt, input_type, ...)
```

Arguments

path	The root folder containing all groups folders, default is the present working folder.
groups	The name of the groups, the input value should be vector, e.g. 'groups=c('group1','group2')', default is all folders in the folder path.
outfmt	The output format, could specify as 'list' or 'dataframe' or 'df', default is dataframe.
input_type	Character. The input file format, should be one of these: 'picard', 'bam', 'cfd-napro'. The bam files has to be marked duplicates.
...	Further arguments passed to or from other methods.

Value

The function returns the insert size metrics of each group in list or dataframe format.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the size.
df <- callSize(path = path)
```

`callValleyDistance` *Calculate the inter-valley distance of insert size*

Description

Calculate the inter-valley distance of insert size

Usage

```
callValleyDistance(
  path = getwd(),
  groups,
  limit,
  outfmt,
  summary,
  mincount,
  input_type,
  ...
)
```

Arguments

`path`	The root folder containing all groups folders, default is the present working folder.
`groups`	The name of the groups, the input value should be vector, e.g. groups = c('group1','group2'), default is all folders in the folder path.
`limit`	The insert size range that will be focused on, default value is 'limit = c(35,135)'.
`outfmt`	The output format, could specify as 'list' or 'dataframe' or 'df', default is dataframe.
`summary`	If TRUE, summarize the output.
`mincount`	The minimum count value of inter-valley distance.
`input_type`	Character. The input file format should be 'picard' or 'bam', or 'cfdnapro'. The bam files has to be marked duplicates.
`...`	Further arguments passed to or from other methods.

Value

The inter-valley distance in a list or dataframe.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the inter-valley distance.
df <- callValleyDistance(path = path)
```

`examplePath` *Get path to cfDNAPro example folder.*

Description

cfDNAPro package has sample files in ‘inst/extdata‘ directory. This function helps get the path to the data.

Usage

```
examplePath(data = NULL)
```

Arguments

`data`	Name of data set. Such as "groups_picard" or "step6". If ‘NULL‘, the path of extdata folder will be returned.

Value

A string. (i.e. the path.)

Examples

```
examplePath()
examplePath("groups_picard")
examplePath("step6")
```

`plotAllToOne` *Plot the raw fragment size metrics (e.g. proportion, cdf and 1-cdf) of all groups with different colors in a single plot*

Description

Plot the raw fragment size metrics (e.g. proportion, cdf and 1-cdf) of all groups with different colors in a single plot

Usage

```
plotAllToOne(x, order, plot, vline, xlim, ylim, ...)
```

Arguments

x	A long-format dataframe contains the metrics of different cohort.
order	The groups show in the final plot, the input value should be vector, e.g. 'groups=c('group1','group2')', default is all folders in the folder path.
plot	The plot type, default is 'all' which means all of proportion, cdf and 1-cdf plots will be shown.
vline	Vertical dashed lines, default value is 'c(81,167)'.
xlim	The x axis range shown in the plot. Default is 'c(0,500)'.
ylim	The y axis range shown in the fraction of fragment size plots. Default is 'c(0,0.035)'.
...	Further arguments passed to or from other methods.

Value

The function returns a list plots.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the sizes.
df <- callSize(path = path)
# Plot all samples from multiple groups into one figure.
plot <- plotAllToOne(df)
```

---

plotMetrics *Plot the fragment size metrics (i.e. proportion, cdf and 1-cdf)*

Description

Plot the fragment size metrics (i.e. proportion, cdf and 1-cdf)

Usage

```
plotMetrics(x, order, plot, vline, xlim, ylim, ...)
```

Arguments

x	A long-format dataframe contains the metrics of different cohort.
order	The groups show in the final plot, the input value should be vector, e.g. 'groups = c('group1','group2')", default is all folders in the folder path
plot	The plot type, default is 'all': both median and mean metrics will be shown. They will include: mean_prop, mean_cdf, mean_1-cdf, median_prop, median_cdf, median_1-cdf. Could also specify as "median" or "mean".
vline	Vertical dashed lines, default value is c(81,167).
xlim	The x axis range shown in the plot. Default is c(0,500).
ylim	The y axis range shown in the fraction of fragment size plots. Default is c(0,0.0125).
...	Further arguments passed to or from other methods.

Value

The function returns a list plots.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the metrics.
df <- callMetrics(path = path)
# Plot metrics.
plot <- plotMetrics(df,
    plot = "median",
    order = c("cohort_1", "cohort_2")
)
```

---

plotMode *Plot mode fragment size*

Description

Plot mode fragment size

Usage

```
plotMode(x, order, type, mincount, hline, ...)
```

Arguments

x	A long-format dataframe contains the interpeak distance, a template please refer to the result of "callPeakdist" function.
order	The groups show in the final plot, the input value should be vector, e.g. ‘groups = c("group1","group2")‘, default is all folders in the folder path.
type	The plot type, could choose "bin" or "stacked" chart. Default is bin plot.
mincount	Minimum count of mode fragment size that will be included. Count number smaller than this value will be removed first, then proportion of each count value will be calculated. Default value is 0.
hline	The horizontal lines added to the bin plot. Default lines will be ‘c(81,112,170)‘.
...	Further arguments passed to or from other methods.

Value

The function returns the plot.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the modes.
df <- callMode(path = path)
# Plot modes.
plot <- plotMode(df, hline = c(80, 111, 170))
```

---

plotModeSummary *Summarize and plot mode fragment size in a stacked bar chart*

Description

Summarize and plot mode fragment size in a stacked bar chart

Usage

```
plotModeSummary(x, order, summarized, mode_partition, ...)
```

Arguments

x	A long-format dataframe contains mode fragment size, a template please refer to the result of 'callMode' function.
order	The groups show in the final plot, the input value should be vector, e.g. 'groups = c('group1','group2')', default is all folders in the folder path.
summarized	Logical value, default is False.
mode_partition	This should be a list. This decides how the modes are partitioned in each stacked bar. Default value is 'list(c(80, 81), c(111, 112), c(167))'. Also this function will automatically calculate an 'Others' group which includes the modes not mentioned by users.
...	Further arguments passed to or from other methods.

Value

The function returns the plot.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")
# Calculate the modes.
df <- callMode(path = path)
# Plot mode summary.
plot <- plotModeSummary(df,
    mode_partition = list(c(80, 81), c(111, 112), c(167))
)
```

---

plotPeakDistance *Plot the inter-peak distance of fragment size distance distribution*

Description

Plot the inter-peak distance of fragment size distance distribution

Usage

```
plotPeakDistance(x, summarized, order, type, mincount, xlim, ...)
```

Arguments

x	A long-format dataframe contains the inter-peak distance, a template please refer to the result of 'callPeakDistance' function.
summarized	Logical value, describe whether the x is summarzied already. summarized means the count and proportion of each interpeak_dist.
order	The groups show in the final plot, the input value should be vector, e.g. 'groups = c('group1','group2')', default is all folders in the folder path.
type	The plot type, default is line plot, now only support line plot. Don't change this parameter in this version, keep it as default.
mincount	Minimum count value of inter peak distance, count number less than this value will be removed first, then proportion of each count value will be calculated. Default value is 0.
xlim	The x axis range shown in the plot. Default is 'c(8,13)'.
...	Further arguments passed to or from other methods.

Value

The function returns the line plot of inter peak distance.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")

# Calculate the inter-peak distance.
df <- callPeakDistance(path = path)

# Plot the inter-peak distance.
plot <- plotPeakDistance(df,
    xlim = c(8, 13),
    mincount = 2
)
```

---

plotSingleGroup *Plot the raw fragment size metrics of single group in a single plot, colored by samples.*

Description

Plot the raw fragment size metrics of single group in a single plot, colored by samples.

Usage

```
plotSingleGroup(x, xlim, ylim, vline, order, plot, ...)
```

Arguments

x	A long-format dataframe contains the metrics of different cohort.
xlim	The x axis range shown in the plot. Default is 'c(0,500)'.
ylim	The y axis range shown in the fraction of fragment size plots. Default is 'c(0,0.035)'.
vline	Vertical dashed lines, default value is 'c(81,167)'.
order	The groups show in the final plot, the input value should be vector, e.g. 'order = c('group1')", default is all groups/cohorts in the folder path.
plot	The plot type, default is 'all' which means both proportion, cdf and 1-cdf plots will be shown.
...	Further arguments passed to or from other methods.

Value

The function returns a list plots.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")

# Calculate the metrics.
df <- callMetrics(path = path)

# Plot the only the group specified..
plot <- plotSingleGroup(x = df, order = c("cohort_1"))
```

---

plotValleyDistance *Plot the inter-valley distance of fragment size distance distribution*

Description

Plot the inter-valley distance of fragment size distance distribution

Usage

```
plotValleyDistance(x, order, type, mincount, xlim, ...)
```

Arguments

x	A long-format dataframe contains the inter-valley distance, a template please refer to the result of 'callValleyDistance' function.
order	The groups show in the final plot, the input value should be vector, e.g. 'groups=c('group1','group2')', default is all folders in the folder path.
type	The plot type, default is line plot, now only support line plot. Don't change this parameter in this version, keep it as default.
mincount	Minimum count value of inter valley distance, count number less than this value will be removed first, then proportion of each count value will be calculated. Default value is 0.
xlim	The x axis range shown in the plot. Default is c(8,13).
...	Further arguments passed to or from other methods.

Value

The function returns the line plot of inter valley distance.

Author(s)

Haichao Wang

Examples

```
# Get the path to example data.
path <- examplePath("groups_picard")

# Calculate the inter-valley distance.
df <- callValleyDistance(path = path)

# Plot the inter-valley distance.
plot <- plotValleyDistance(df,
    xlim = c(8, 13),
    mincount = 2
)
```

---

readBam *Read bam file into a curated GRanges object*

Description

Read bam file into a curated GRanges object

Usage

```
readBam(
  bamfile,
  chromosome_to_keep = paste0("chr", 1:22),
  strand_mode = 1,
  genome_label = "hg19",
  outdir = NA,
  ...
)
```

Arguments

bamfile	The bam file name.
chromosome_to_keep	Should be a character vector containing the seqnames to be kept in the GRanges object. Default is paste0("chr", 1:22).
strand_mode	Usually the strand_mode = 1 means the First read is aligned to positive strand. Details please see GenomicAlignments docs.
genome_label	The Genome you used in the alignment. Should be "hg19" or "hg38" or "hg38-NCBI. Default is "hg19". Note: "hg19" will load BSgenome.Hsapiens.UCSC.hg19 package, which is Full genome sequences for Homo sapiens (Human) as provided by UCSC (hg19, based on GRCh37.p13) and stored in Biostrings objects; "hg38" will load BSgenome.Hsapiens.UCSC.hg38 package, which is Full genome sequences for Homo sapiens (Human) as provided by UCSC (hg38, based on GRCh38.p13) and stored in Biostrings objects. "hg38-NCBI" will load BSgenome.Hsapiens.NCBI.GRCh38 package, which is full genome sequences for Homo sapiens (Human) as provided by NCBI (GRCh38, 2013-12-17) and stored in Biostrings objects.
outdir	The path for saving rds file. Default is NA, i.e. not saving.
...	Further arguments passed to or from other methods.

Value

This function returns curated GRanges object.

Author(s)

Haichao Wang

Examples

```
## Not run:

object <- read_bam(bamfile = "/path/to/bamfile.bam",
                   outdir = "./",
                   chromosome_to_keep = c("chr1", "chr2", "chr3"))

## End(Not run)
```

`read_bam_insert_metrics` *Calculate insert sizes from a curated GRanges object*

Description

Calculate insert sizes from a curated GRanges object

Usage

```
read_bam_insert_metrics(
  bamfile,
  chromosome_to_keep = paste0("chr", 1:22),
  strand_mode = 1,
  genome_label = "hg19",
  outdir = NA,
  isize_min = 1L,
  isize_max = 1000L,
  ...
)
```

Arguments

`bamfile`	The bam file name.
`chromosome_to_keep`	Should be a character vector containing the seqnames to be kept in the GRanges object. Default is paste0("chr", 1:22).
`strand_mode`	Usually the strand_mode = 1 means the First read is aligned to positive strand. Details please see GenomicAlignments docs.
`genome_label`	The Genome you used in the alignment. Should be "hg19" or "hg38" or "hg38-NCBI. Default is "hg19". Note: "hg19" will load BSgenome.Hsapiens.UCSC.hg19 package, which is Full genome sequences for Homo sapiens (Human) as provided by UCSC (hg19, based on GRCh37.p13) and stored in Biostrings objects; "hg38" will load BSgenome.Hsapiens.UCSC.hg38 package, which is Full genome sequences for Homo sapiens (Human) as provided by UCSC (hg38, based on GRCh38.p13) and stored in Biostrings objects. "hg38-NCBI" will load BSgenome.Hsapiens.NCBI.GRCh38 package, which is full genome sequences for Homo sapiens (Human) as provided by NCBI (GRCh38, 2013-12-17) and stored in Biostrings objects.
`outdir`	The path for saving rds file. Default is NA, i.e. not saving.
`isize_min`	min fragment length to keep, default is 1L.
`isize_max`	max fragment length to keep, default is 1000L.
`...`	Further arguments passed to or from other methods.

Value

This function returns a dataframe with two columns: "insert_size" and "All_Reads.fr_count".

Author(s)

Haichao Wang

Examples

```
## Not run:

object <- read_bam_insert_metrics(bamfile = "/path/to/bamfile.bam")

## End(Not run)
```

Index

callMetrics, 2
callMode, 3
callPeakDistance, 4
callSize, 6
callValleyDistance, 7

examplePath, 8

plotAllToOne, 8
plotMetrics, 9
plotMode, 10
plotModeSummary, 11
plotPeakDistance, 12
plotSingleGroup, 13
plotValleyDistance, 14

read_bam_insert_metrics, 17
readBam, 15