Package ‘MDTS’

July 10, 2025

Title Detection of de novo deletion in targeted sequencing trios

Version 1.28.0

Description A package for the detection of de novo copy number deletions in targeted sequencing of trios with high sensitivity and positive predictive value.

Depends R (>= 3.5.0)

Imports GenomicAlignments, GenomicRanges, IRanges, Biostrings, DNAcopy, Rsamtools, parallel, stringr

Suggests testthat, knitr

VignetteBuilder knitr

License Artistic-2.0

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

biocViews StatisticalMethod, Technology, Sequencing, TargetedResequencing, Coverage, DataImport

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

git_branch RELEASE_3_21

git_last_commit 2078f3e

git_last_commit_date 2025-04-15

Repository Bioconductor 3.21

Date/Publication 2025-07-09

Author Jack M.. Fu [aut, cre]

Maintainer Jack M.. Fu `<jmfu@jhsph.edu>`

Contents

`MDTS-package` *The MDTS package for Detection of Denovo Deletions from Targeted Sequencing Data Using Minimum-Distance*

Description

The MDTS package for Detection of Denovo Deletions from Targeted Sequencing Data Using Minimum-Distance

---

`calcBins` *Sample driven calculation of bins for counting*

Description

This function will randomly select a sample of bam files to calculate dynamic MDTS bins for subsequent read-depth analysis.

Usage

```
calcBins(metaData, n, readLength, medianCoverage, minimumCoverage, genome,
  mappabilityFile, seed = 1337)
```

Arguments

`metaData`	A table in the format of the output of getMetaData().
`n`	The number of subsamples to use.
`readLength`	The read length of the experiment.
`medianCoverage`	The median number of reads across sub-samples to reach before creating a new bin.
`minimumCoverage`	The miminum number of coverage across all sub-samples required to create the proto-region.
`genome`	The BSGenome object that assists in calculations of the GC content of the bins.
`mappabilityFile`	A path to the bigwig file of 100mer mappability of the corresponding genome.
`seed`	Sets the seed so results are reproducible. Defaults to 1337.

Value

Returns a `GRanges` object depicting the dynamic bins that MDTS calculates.

Examples

```
load(system.file("extdata", 'bins.RData', package = "MDTS"))
bins
```

`calcCounts` *Creating the raw count matrix*

Description

This function will return a matrix of read counts where ecah column is a sample, and each row is a bin.

Usage

```
calcCounts(metaData, bins, rl, mc.cores = 1)
```

Arguments

`metaData`	A table in the format of the output of getMetaData().
`bins`	The set of bins determined by calcBins().
`rl`	The read length of the experiment.
`mc.cores`	The number of cores to use for multi-threaded analysis. Defaults to 1.

Value

A `data.frame` that contains the counts for each sample in the `metaData` input that fall into each segment of `bins`.

Examples

```
## Not run:
pD <- getMetaData(
'https://raw.githubusercontent.com/JMF47/MDTSData/master/data/pD.ped')
genome = BSgenome.Hsapiens.UCSC.hg19
map_file <-
"https://raw.githubusercontent.com/JMF47/MDTSData/master/data/chr1.map.bw"
bins = calcBins(pD, n=5, rl=100, med=150, min=5, genome, map_file)

## End(Not run)
load(system.file("extdata", 'bins.RData', package = "MDTS"))
load(system.file("extdata", 'counts.RData', package = "MDTS"))
counts
```

`calcMD` *Calculating the Minimum Distance matrix*

Description

This function will return a matrix of minimum distances where ecah column is a family, and each row is a bin.

Usage

```
calcMD(mCounts, metaData)
```

Arguments

`mCounts`	A matrix of normalized coverage output by normalizedCounts().
`metaData`	A table in the format of the output of metaData().

Value

A `data.frame` of minimum distances. Each column is a trio, while each row is an entry in `bins`

Examples

```
load(system.file("extdata", 'bins.RData', package = "MDTS"))
load(system.file("extdata", 'counts.RData', package = "MDTS"))
load(system.file("extdata", 'pD.RData', package = "MDTS"))
mCounts <- normalizeCounts(counts, bins)
md <- calcMD(mCounts, pD)
```

`denovoDeletions` *Denovo Deletion Calling*

Description

This function will return a single GRanges object containing all denovo deletions that passed filtering from a Circular Binary Segmentation object with supplementary information.

Usage

```
denovoDeletions(cbs, mCounts, bins)
```

Arguments

`cbs`	The output from segmentMD().
`mCounts`	The normalized counts matrix output by normalizeCounts().
`bins`	The set of bins determined by calcBins().

Value

A `GRanges` object that reports all detected denovo deletions passing requite filters.

Examples

```
load(system.file("extdata", 'bins.RData', package = "MDTS"))
load(system.file("extdata", 'counts.RData', package = "MDTS"))
load(system.file("extdata", 'pD.RData', package = "MDTS"))
mCounts = normalizeCounts(counts, bins)
md = calcMD(mCounts, pD)
cbs = segmentMD(md, bins)
denovo = denovoDeletions(cbs, mCounts, bins)
```

`getMetaData` *Constructor for metadata*

Description

This function allows constructor of phenotype information necessary for downstream analysis. See format of required fields. Function will also rearrange the rows such that trios are grouped together - with proband first, mother second, and father third.

Usage

```
getMetaData(path, id = "subj_id", familyId = "family_id",
  fatherId = "father_id", motherId = "mother_id", bamPath = "bam_path")
```

Arguments

`path`	The path pointing to the file that contains information on each subject in the dataset.
`id`	The column name that identifies each sample. Defaults to ’subj_id’.
`familyId`	The column name that identifies which family the sample belongs to. Defaults to ’family_id’.
`fatherId`	The column name that identifies the id of the father. Defaults to ’father_id’.
`motherId`	The column name that identifies the id of the mother. Defaults to ’mother_id’.
`bamPath`	The column name that identifies where to find the bam file for each subject. Defaults to ’bam_path’.

Value

Returns a `data.frame` of required sample information for running MDTS.

Examples

```
meta <- getMetaData(
'https://raw.githubusercontent.com/JMF47/MDTSData/master/data/pD.ped')
```

`normalizeCounts` *Calculating the normalized M scores*

Description

This function will return a matrix of normalized M scores where ecah column is a sample, and each row is a bin.

Usage

```
normalizeCounts(counts, bins, GC = TRUE, map = TRUE, mc.cores = 1)
```

Arguments

`counts`	A matrix of raw coverage output by calcCounts().
`bins`	The set of bins determined by calcBins().
`GC`	Whether to loess adjust for GC. Defaults to TRUE.
`map`	Whether to loess adjust for mappability. Defaults to TRUE. Defaults to 1.
`mc.cores`	The number of cores to use for multi-threaded analysis.

Value

A `data.frame` of normalized counts. Each column is a sample, and each row is a entry of `bins`.

Examples

```
load(system.file("extdata", 'bins.RData', package = "MDTS"))
load(system.file("extdata", 'counts.RData', package = "MDTS"))
load(system.file("extdata", 'pD.RData', package = "MDTS"))
mCounts <- normalizeCounts(counts, bins)
```

`segmentMD` *Circular Binary Segmentation on Minimum Distances*

Description

This function will return a GRanges object containing the copy number segments of all families in the input minimum distance matrix. It calls segment() from DNAcopy (alpha=0.001, undo.splits="sdundo", undo.SD=4).

Usage

```
segmentMD(md, bins, alpha = 0.001, undo.splits = "sdundo", undo.SD = 4,
  mc.cores = 1)
```

Arguments

`md`	The minimum distance matrix produced by calcMD.
`bins`	The set of bins determined by calcBins.
`alpha`	Controls the alpha option in calling DNAcopy::segment()
`undo.splits`	Controls the undo.splits option in DNAcopy::segment()
`undo.SD`	Controls the undo.SD option in calling DNAcopy::segment()
`mc.cores`	The number of cores to use for multi-threaded analysis. Defaults to 1.

Value

A `data.frame` containing the segmented regions based to be parsed by denovoDeletions() minimum distance.

Examples

```
load(system.file("extdata", 'bins.RData', package = "MDTS"))
load(system.file("extdata", 'counts.RData', package = "MDTS"))
load(system.file("extdata", 'pD.RData', package = "MDTS"))
mCounts <- normalizeCounts(counts, bins)
md <- calcMD(mCounts, pD)
cbs <- segmentMD(md, bins)
```

`visualizeDeletion` *Visualization for deletions*

Description

This function plots the raw read information from the location of interest for a trio.

Usage

```
visualizeDeletion(deletion, bins, metaData, mCounts, md, save = FALSE)
```

Arguments

`deletion`	A GRanges object in the format of the output of denovoDeletions().
`bins`	The set of bins determined by calcBins().
`metaData`	A table in the format of the output of getMetaData().
`mCounts`	A matrix of normalized coverage output by normalizedCounts().
`md`	The minimum distance matrix output by calcMD()
`save`	If TRUE will save plot to current working directory instead of rendering.

Value

The file name if the plot was saved.

Examples

```
## Not run:
load(system.file("extdata", 'bins.RData', package = "MDTS"))
load(system.file("extdata", 'counts.RData', package = "MDTS"))
load(system.file("extdata", 'pD.RData', package = "MDTS"))
mCounts <- normalizeCounts(counts, bins)
md <- calcMD(mCounts, pD)
cbs <- segmentMD(md, bins)
denovo <- denovoDeletions(cbs, mCounts, bins)
visualizeDeletion(denovo[1], bins, pD, mCounts, md)

## End(Not run)
```

Index