Package ‘MassArray’

December 21, 2016

Type Package
Title Analytical Tools for MassArray Data
Version 1.26.0
Date 2009-05-19
Author Reid F. Thompson <rthompso@aecom.yu.edu>, John M. Greally <jgreally@aecom.yu.edu>
Maintainer Reid F. Thompson <rthompso@aecom.yu.edu>
Imports graphics, grDevices, methods, stats, utils
Depends R (>= 2.10.0), methods
Description This package is designed for the import, quality control, analysis, and visualization of methylation data generated using Sequenom’s MassArray platform. The tools herein contain a highly detailed amplicon prediction for optimal assay design. Also included are quality control measures of data, such as primer dimer and bisulfite conversion efficiency estimation. Methylation data are calculated using the same algorithms contained in the EpiTyper software package. Additionally, automatic SNP-detection can be used to flag potentially confounded data from specific CG sites. Visualization includes barplots of methylation data as well as UCSC Genome Browser-compatible BED tracks. Multiple assays can be positionally combined for integrated analysis.
License GPL (>=2)
biocViews DNAMethylation, SNP, MassSpectrometry, Genetics, DataImport, Visualization
LazyData yes
LazyLoad yes
NeedsCompilation no

R topics documented:

ampliconPrediction .................................................. 2
analyzeCpGs .......................................................... 4
bisConvert ............................................................ 5
calcMeth .............................................................. 5
calcMW ................................................................. 6
calcPercentAdduct .................................................. 7
calcPercentConversion ............................................. 8
combine ............................................................... 9
ampliconPrediction

ampliconPrediction

Amplicon prediction

ampliconPrediction

Description

Function to predict amplicon fragmentation pattern and details for T&C reactions on the plus and minus strands

Usage

ampliconPrediction(sequence, lower.threshold = 1500, upper.threshold = 7000, fwd.tag = "AGGAAGAGAG", rev.tag = "AGCCTTCTCCC", plot = TRUE, table = TRUE, lwd = 1, cex = 1, multiple.conversion = FALSE)

Arguments

sequence Nucleotide sequence input as a character string
lower.threshold Lower limit (in Da) of usable mass window (default: ‘1500’)
upper.threshold Upper limit (in Da) of usable mass window (default: ‘7000’)
ampliconPrediction

fwd.tag          Nucleotide tag sequence 5’ of the forward primer
rev.tag          T7-containing nucleotide tag sequence 5’ of the reverse primer
plot             Logical specifying whether or not to display graphical representation of fragmentation profiles (default is TRUE)
table            Logical specifying whether or not to return tabular representation of fragmentation profiles (default is TRUE)
lwd              The line width used for fragmentation display, a positive number, defaulting to 1
cex              A numerical value (defaulting to 1) giving the amount by which plotting text and symbols should be magnified relative to the default
multiple.conversion Logical value specifying whether or not to include multiple CGs on the same conversion control fragment where possible (default is FALSE).

Details
Plotted fragmentation patterns contain a number of detailed features including: CG positions, molecular weight overlaps, conversion controls, fragment assayability, and more.

Note that the graphical output does not contain a built-in legend at this time, but the plot may be interpreted as follows: Putative fragmentation patterns are shown for T and C-cleavage reactions on both the plus and minus strands of an input amplicon, with the T-forward, T-reverse, C-forward, and C-reverse shown in descending order. CG dinucleotides (filled circles) are numbered and colored in blue. Other fragments are colored according to their ability to be assayed: fragment molecular weight outside the testable mass window (gray), fragment molecular weight overlapping with another fragment (red), fragment containing a potential conversion control (green), or fragment uniquely assayable but containing no CGs (black). Linked arrowheads denote molecular weight overlaps between multiple CG-containing fragments. Yellow highlights represent tagged or primer sequences, while lavender highlights denote user-specified “required” sites.

Value
If table is TRUE, returns a list containing the following items:

summary          A summary matrix of logical values specifying whether or not each CG is assayable by a given combination of cleavage reaction and DNA strand
counts           A numerical tally of the quantity of CGs that are assayable by each assay

Author(s)
Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples
ampliconPrediction("TGGAAACCCAGCAAAGATCAAGCAGAAAAGGGCGCAGCAGCCCTCGTGGCTAACCTCCTCTCACCCTGGTACCCCAAGGCACCGG")
analyzeCpGs

**analyzeCpGs**

Analyze CG methylation

**Description**

Function to determine percent methylation for all CGs from input fragmentation

**Usage**

```r
analyzeCpGs(fragments, peaks, method = c("weighted", "proportion"))
```

**Arguments**

- `fragments`: List of MassArrayFragment objects
- `peaks`: List of MassArrayPeak objects comprising spectral data for a given assay
- `method`: Specifies which algorithm to use when calculating percent methylation (either "weighted" or "proportion")

**Details**

Wrapper function for `calcMeth()`, takes fragmentation pattern and spectral data as input and applies percent methylation calculation for all CG-containing, non conversion control fragments

**Value**

Returns a list of numerical values corresponding to percent methylation for each CG dinucleotide, with 0

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**See Also**

See Also `calcMeth`

**Examples**

```r
data(MassArray.example.data)
cpg.data <- analyzeCpGs(MassArray.example.data$fragments.T, MassArray.example.data$samples[[1]]$peaks, method="weighted")
barplot(cpg.data, xlab="CpG (Number)", ylim=c(0,1), ylab="Methylation (Percent)")
```
**bisConvert**

**Bisulphite conversion**

**Description**
Bisulphite convert nucleotide sequence input

**Usage**
`bisConvert(sequence)`

**Arguments**
- `sequence` Nucleotide sequence in the form of a character string

**Value**
Returns a character value corresponding to the bisulphite converted input sequence.

**Author(s)**
Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**Examples**
`bisConvert("AAATTCGGAACCC")`

---

**calcMeth**

**Calculate percent methylation**

**Description**
Function to calculate percent methylation from a collection of peaks corresponding to a single fragment.

**Usage**
`calcMeth(SNR.list, fragments = rep(1, length(SNR.list)), non.cg.fragments = numeric(0), method = c("weighted", "proportion"), prune.non.cg.peaks = TRUE, na.rm = FALSE)`

**Arguments**
- `SNR.list` List of signal-to-noise ratios, sorted from low to high MWs, corresponding to the unmethylated and methylated peaks for a given set of fragments
- `fragments` List of all fragments contributing to each of the input peaks, automatically defaulting to a single fragment
- `non.cg.fragments` List of fragments (without CGs) contributing to any of the input peaks, automatically defaulting to `numeric(0)`
- `method` Specifies which algorithm to use when calculating percent methylation (either "weighted" or "proportion")
Function to calculate molecular weight of a fragment generated by the MassCLEAVE assay for either the T or C cleavage reactions

**Details**

Note that the current release of this function performs as expected for the large majority of cases. However, certain complex combinations of peak overlaps are not handled at this time. This may affect data for a minority of points, particularly those containing multiple overlaps with alternative fragments. Please ensure more in-depth review of such loci.

**Value**

Returns a numerical value corresponding to percent methylation, with 0

**Author(s)**

Reid F. Thompson (<rthompson@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**References**


**See Also**

See Also `MassArrayPeak`

**Examples**

```r
data(MassArray.example.data)
frags <- MassArray.example.data$fragments.T[[6]]$"MW"
peaks <- findPeaks(frags, unlist(lapply(MassArray.example.data$samples[[1]]$peaks, slot, "MW.actual")))
SNRs <- unlist(lapply(MassArray.example.data$samples[[1]]$peaks[peaks], slot, "SNR"))
frag.list <- list(1:3, 1:3, 1:3, 1:3)
calcMeth(SNRs, fragments=frag.list, method="weighted")
calcMeth(SNRs, fragments=frag.list, method="proportion")
```
calcPercentAdduct

Usage

calcPercentAdduct(peaks)

Arguments

peaks List of MassArrayPeak objects comprising complete spectral data

Details

Salt adducts (either Na or K) are identified and compared to each of their unmodified reference peaks

Value

Returns a list of numerical values corresponding to the ratios of salt adduct peak heights to their unmodified reference peaks

Description

Function to calculate ratio of salt adduct peak heights to reference/unmodified peaks

calcPercentAdduct 

Calculate percent salt adducts

Usage

calcPercentAdduct(peaks)

Arguments

peaks List of MassArrayPeak objects comprising complete spectral data

Details

Salt adducts (either Na or K) are identified and compared to each of their unmodified reference peaks

Value

Returns a list of numerical values corresponding to the ratios of salt adduct peak heights to their unmodified reference peaks

calcMW

Usage

calcMW(sequence, extra = c("5OH", "5PPP-3P", "5PPP-3OH"), adduct = c("", "Na", "K"), rxn = c("T", "C"))

Arguments

sequence Nucleotide sequence input
extra One of "5OH" (default), "5PPP-3P", or "5PPP-3OH" describing 5' and/or 3' modifications of the fragment
adduct One of 'Na', 'K', or " (default) specifying whether or not the molecular weight should be calculated for a salt adduct
rxn One of 'T' or 'C' indicating which cleavage reaction is employed to generate the fragment

Value

Returns a numerical output corresponding to the molecular weight (in Da) of sequence input. Note that the output may actually represent multiple molecular weights if/whenever the input sequence contains one or more degenerate bases (e.g. R or Y).

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

calcMW("AAATCCC")
calcMW("AARTYCC")
Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

data(MassArray.example.data)
adduct.ratios <- calcPercentAdduct(MassArray.example.data$samples[[1]]$peaks)
median(adduct.ratios)

calcPercentConversion  Calculate percent conversion

Description

Function to calculate percent methylation (wrapper for calcMeth() function) for each identified conversion control

Usage

calcPercentConversion(fragments, peaks)

Arguments

   fragments List of MassArrayFragment objects
peaks      List of MassArrayPeak objects comprising spectral data to be used for conversion control calculations

Details

This function serves as a wrapper function for calcMeth(), such that percent methylation is calculated for all conversion controls within the input list of fragments.

Value

Returns a list of numerical values (from 0 to 1) corresponding to percent methylation for each conversion control, with 0 Note that each element within the returned list will represent conversion control(s) for a single sample, while each element may contain multiple values with each value corresponding to data obtained from a single conversion control fragment.

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also

See Also calcMeth, convControl
Examples

data(MassArray.example.data)
conversion.data <- calcPercentConversion(MassArray.example.data$fragments.T, MassArray.example.data$samples[1])
mean(conversion.data)

# NOTE: conversion control data may already be contained within a MassArrayData object; these data can be accessed and visualized by the following (or alternative) commands
conversion.data <- unlist(lapply(lapply(MassArray.example.data$samples, slot, "quality.conversion"), median, na.rm=TRUE))
barplot(conversion.data)

combine

Combine MassArrayData objects

Description

Function to join two MassArrayData objects by sequence positions and samples

Usage

combine(x, y, ...)

Arguments

x MassArrayData object
y MassArrayData object
...
Other arguments passed to combine not supported at this time.

Value

Returns a single MassArrayData object that contains a union of samples and amplicons and spectral data from both MassArrayData objects in input

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also

See Also MassArrayData

Examples

data(MassArray.example.data)
samples(MassArray.example.data)
combined.data <- combine(MassArray.example.data[2,], MassArray.example.data[1,])
samples(combined.data)
Description

Methods for joining two MassArrayData objects by sequence positions and samples, or simply operating on a single MassArrayData object to combine samples, depending on input.

Methods

\[ x = \text{"MassArrayData"}, y = \text{"MassArrayData"} \] Combine two MassArrayData objects by position and then by sample.

\[ x = \text{"MassArrayData"}, y = \text{"missing"} \] Combine duplicate samples within the same MassArrayData object.

See Also

See Also `combine`.

convControl

Conversion control

Description

Function to identify all potential conversion controls in a given input sequence, for a given list of fragments.

Usage

convControl(sequence, fragmentation, multiple = FALSE)

Arguments

sequence Nucleotide sequence input as a character string.
fragmentation List of MassArrayFragment objects corresponding to the fragmentation pattern of the sequence input.
multiple Logical value specifying whether or not to include multiple CGs on the same conversion control fragment where possible (default is FALSE).

Details

Potential conversion controls are identified from the nucleotide sequence input through pattern recognition of fragments that contain non-CG cytosines. Any conversion controls that contain CG dinucleotides or have a molecular weight outside of the usable mass window are screened out. Additionally, conversion controls that are located in identified primer sequence or have molecular weight overlap with other fragments are removed from consideration. Lastly, if the consideration of the fragment as a conversion control will cause new molecular weight overlap(s) with one or more other fragments, the control is also removed from consideration.
countCGs

Value

Returns a list of MassArrayFragment objects identical to the input, with the exception that conversion controls are labeled and updated accordingly.

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also

See Also as MassArrayFragment

Examples

data(MassArray.example.data)
MassArray.example.data$fragments.T[[54]]
conversion.data <- convControl(MassArray.example.data$sequence, MassArray.example.data$fragments.T)
conversion.data[[54]]

countCGs

Count number of CGs

Description

Function to count the number of CG dinucleotides in a given sequence (can include special characters for degenerate bases - i.e. 'Y' or 'R')

Usage

countCGs(sequence)

Arguments

sequence Nucleotide sequence input as a character string

Value

Returns a numerical count of the number of CG dinucleotides in a given sequence, 'NA' if sequence input is 'NA'

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

countCGs("AAACGCGAAAAAYGAAA")
**createWiggle**

**Create wiggle track**

**Description**

Function to create and write a wiggle track (UCSC Genome Browser format) to flat file from methylation data contained in a MassArrayData object.

**Usage**

```r
createWiggle(x, file = "", append = FALSE, colors = NULL, na.rm = FALSE, sep = " ")
```

**Arguments**

- `x` MassArrayData object containing methylation data for at least one sample.
- `file` location of file to write wiggle track information; if "", wiggle track prints to the standard output connection: see `cat`.
- `append` logical; if `TRUE`, the output is appended to an existent wiggle track file. If `FALSE` (default), a new file with a new header is created and any existing file of the same name is destroyed.
- `colors` vector of colors, indicates which colors to use for which wiggle track
- `na.rm` logical; if `TRUE` (default), missing values are removed from data. If `FALSE` any missing values cause an error.
- `sep` a string used to separate columns. Using `sep = "\t"` (default) gives tab-delimited output.

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**References**


**See Also**

`write`, `cat`

**Examples**

```r
data(MassArray.example.data)
createWiggle(MassArray.example.data)
```
estimatePrimerDimer  Estimate Primer Dimers

Description

Function to estimate level of signal due to primer dimers in a given spectrum

Usage

estimatePrimerDimer(fragments, peaks, method = c("ratio", "mann-whitney"))

Arguments

fragments List of MassArrayFragment objects corresponding to the sample
peaks List of MassArrayPeak objects comprising spectral data for a complete assay
method Specifies which algorithm to use when estimating primer dimer levels (either "ratio" or "mann-whitney")

Details

Primer dimers are calculated by: 1) identifying fragments that occur within the expected primer sequence, 2) identifying which of these fragments is assayable, and 3) comparing the overall signal for primer peaks and peaks from the rest of the amplicon.

Value

Returns a list containing primer dimer ratios or significance estimates (i.e. p-values) depending on the analytical method specified ("ratio" or "mann-whitney", respectively). Returns "NA" in cases where insufficient data is present to calculate primer dimer levels.

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also

See Also MassArrayData

Examples

data(MassArray.example.data)
primer.data <- estimatePrimerDimer(MassArray.example.data$fragments.T, MassArray.example.data$samples[[1]])
mean(primer.data)
primer.data <- estimatePrimerDimer(MassArray.example.data$fragments.T, MassArray.example.data$samples[[1]])
mean(primer.data)
evaluateSNPs

Description

Function to analyze a MassArrayData object for all potential single nucleotide polymorphisms (SNPs) indicated by new and/or missing peaks in the spectral data for one or more samples.

Usage

evaluateSNPs(x, verbose = TRUE, plot = TRUE)

Arguments

x MassArrayData object containing spectral data for one or more samples
verbose Logical specifying whether or not to display descriptive progress updates as SNPs are analyzed
plot Logical specifying whether or not to display graphical representation of fragmentation profiles (default is TRUE)

Details

This function performs an exhaustive search for all potential SNPs (single base pair substitutions or deletions) that may give rise to new and/or missing peaks. Graphical output is displayed by default, and extensive data describing putative SNPs is also returned.

Note that the graphical output does not contain a built-in legend at this time, but the plot may be interpreted as follows: In the uppermost panel the T-cleavage fragmentation profile is shown for a given amplicon (C-cleavage reactions occupy a split screen whenever relevant). CG dinucleotides (filled circles) are numbered and colored in blue. Other fragments are colored according to their ability to be assayed: fragment molecular weight outside the testable mass window (gray), fragment molecular weight overlapping with another fragment (red), fragment containing a potential conversion control (green), or fragment uniquely assayable but containing no CGs (black). Putative SNPs are shown directly below their location within the amplicon fragmentation profile. Each row represents analysis from a single sample. Small, gray symbols represent potential SNPs that do not have sufficient evidence (presence of a new peak with corresponding absence of an expected peak). Larger black symbols indicate a potential SNP with both new peaks and missing expected peaks. Triangles indicate base pair substitution, while circles indicate single base pair deletion.

Value

Returns a list of potential SNPs for each identified new peak in the spectral data. Note that each new peak may be explained by any number of potential SNPs; the list returned only includes the most reliable hits, but the redundant nature of the data necessitates returning a nested list, such that each new peak is associated with the following list elements:

- **SNP**: Contains a list of SNPs, each of which takes the form "position:base" where position is the base pair location within the amplicon sequence, and base is the mutated character
- **SNR**: Contains a numerical list of signal-to-noise ratios corresponding to the expected original peak for the fragment mapping to the identified SNP position
**expandSequence**

Contains a numerical list of fragment IDs which map the SNP position to a specific fragment.

Contains a numerical list (values ranging from 0 to 2, with 0 being a highly unlikely SNP and 2 being a SNP with increased likelihood. This number is calculated as a function of new peak SNR and expected peak SNR.

Contains a list of samples whose spectral data contained the given new peak.

Specifies the number of unique SNP and sample pairs, exactly equivalent to the length of SNP, SNR, fragment, SNP.quality, or samples.

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**See Also**

See Also **identifySNPs**

**Examples**

```r
data(MassArray.example.data)
SNP.data <- evaluateSNPs(MassArray.example.data[2,])
```

**Description**

Function to process shorthand form of a nucleotide sequence, where a given base pair followed by a number specifies a run of the indicated nucleotide for the specified length (ex: "A6TTCGA4")

**Usage**

`expandSequence(sequence)`

**Arguments**

- `sequence` Nucleotide sequence input as a character string

**Value**

Returns an expanded nucleotide sequence as a character string

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**Examples**

```r
expandSequence("A6TTCGA4")
expandSequence("C10C10")
expandSequence("AT1CG")
```
findCollisions

Find peak collisions

Description

Function to determine which subset(s) of peaks collide with each other (i.e. molecular weights are indistinguishable given the specified resolution)

Usage

findCollisions(peaks, resolution = 0.5)

Arguments

- peaks: Numerical list of molecular weights (in Da) corresponding to a set of peaks
- resolution: Resolution (in Da), used to specify the ability to distinguish two different molecular weights. For a resolution of ’0.5’ (default), two molecular weights are considered identical if they are less than ’0.5’ Da apart.

Value

Returns a list of peak collisions for each peak in the original list, thus the data object returned is in the form of a nested list.

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

findCollisions(1:5, 1.5)

findFragments

Find fragments

Description

Function to identify which fragment(s) in a list of fragments match a given molecular weight

Usage

findFragments(MW, fragments, resolution = 1)

Arguments

- MW: Molecular weight target (in Da)
- fragments: List of molecular weights corresponding to unique fragments
- resolution: Resolution (in Da), used to specify the ability to distinguish two different molecular weights. For a resolution of ‘1’ (default), two molecular weights are considered identical if they are less than ‘1’ Da apart.
findPeaks

Value

Returns the index or indices of fragment(s) within the input list that have a molecular weight which matches that specified as input

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

data(MassArray.example.data)
findFragments(3913, MassArray.example.data$fragments.T, resolution=0.1)
findFragments(3913, MassArray.example.data$fragments.T, resolution=0.5)

findPeaks  Find peaks

Description

Function to determine which peak(s) in a list of peaks match a given molecular weight.

Usage

findPeaks(MW, peaks, resolution = 1)

Arguments

MW  Molecular weight target (in Da)
peaks  List of molecular weights corresponding to unique peaks
resolution  Resolution (in Da), used to specify the ability to distinguish two different molecular weights. For a resolution of ‘1’ (default), two molecular weights are considered identical if they are less than ‘1’ Da apart.

Value

Returns the index or indices of peak(s) within the input list that have a molecular weight which matches that specified as input

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

findPeaks(3.1, 6:1, res=0)
findPeaks(3.1, 6:1, res=0.2)
identifySNPs

**Description**

Function to identify potential single nucleotide polymorphisms (SNPs) which allow mapping of a novel peak sequence to the expected amplicon sequence

**Usage**

```r
identifySNPs(peak.sequence, sequence, rxn = c("T", "C"))
```

**Arguments**

- `peak.sequence` Nucleotide sequence (can also be base composition - ex: "A6G2C1T3") as a character string
- `sequence` Nucleotide sequence for wildtype/expected amplicon as a character string
- `rxn` One of "T" or "C" specifying which cleavage reaction to use for SNP analysis

**Details**

The algorithm steps through the `sequence`, substituting one nucleotide at a time with the other three base pairs or a blank character (deletion), in order to determine a base compositional match to the input `peak.sequence` which represents a peak not found in the native `sequence`.

**Value**

Returns a list of potential SNP matches for the input `peak.sequence`. Each element of the list contains multiple items as follows:

- `sequence` corresponds to `peak.sequence`
- `position` corresponds to the matched position within `sequence`
- `base` corresponds to the altered nucleotide (i.e. "A", "T", "C", "G", or ")"
- `type` corresponds to the class of SNP (i.e. "substitution" or "deletion")

**Author(s)**

Reid F. Thompson (<rthompson@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**Examples**

```r
## SINGLE SUBSTITUTION
identifySNPs("AAGT","AATTTT")
## MULTIPLE SUBSTITUTION POSSIBILITIES
identifySNPs("A1G1T1","AATTTT")
## DELETION
identifySNPs("AAT","AGATTTT")
```
importEpiTyperData

Import EpiTyper data (v.1.0)

Description
Function to read and import an EpiTyper datafile (v.1.0) and store it as a MassArraySpectrum objects

Usage
importEpiTyperData(data, MassArrayObject, verbose = TRUE)

Arguments
data  location of EpiTyper datafile as a character string
MassArrayObject  Pre-existent MassArrayData object in which store relevant sample and spectral information from datafile
verbose  Logical specifying whether or not to display descriptive progress updates as datafile is processed

Details
EpiTyper v.1.0 datafiles must only contain a single amplicon, thus the user must export peak data for one amplicon at a time.

Value
Returns a list of MassArraySpectrum objects each populated by spectral data

Author(s)
Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also
See Also MassArraySpectrum

importEpiTyperData.new

Import EpiTyper data (v.1.0.5)

Description
Function to read and import an EpiTyper datafile (v.1.0.5) and store it as a list of MassArraySpectrum objects

Usage
importEpiTyperData.new(data, MassArrayObject, verbose = TRUE)
inSilicoFragmentation

Arguments

- **data**: location of EpiTyper datafile as a character string
- **MassArrayObject**: Pre-existent MassArrayData object in which store relevant sample and spectral information from datafile
- **verbose**: Logical specifying whether or not to display descriptive progress updates as datafile is processed

Details

EpiTyper v.1.0.5 datafiles must only contain a single amplicon, thus the user must export peak data for one amplicon at a time.

Value

Returns a list of MassArraySpectrum objects each populated by spectral data

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also

See Also MassArraySpectrum

---

**Description**

Function to perform a complete in silico fragmentation of input sequence. Provides wrapper to a number of different functions, each of which determines additional information about each fragment.

**Usage**

```r
inSilicoFragmentation(sequence, fwd.tag = "", rev.tag = "", type = c("T", "C"), lower.threshold = 1500, upper.threshold = 7000, fwd.primer = 0, rev.primer = 0, multiple.conversion = FALSE)
```

**Arguments**

- **sequence**: Nucleotide sequence input as a character string
- **fwd.tag**: Nucleotide tag sequence 5’ of the forward primer
- **rev.tag**: T7-containing nucleotide tag sequence 5’ of the reverse primer
- **type**: One of ‘T’ or ‘C’ indicating which cleavage reaction to use
- **lower.threshold**: Lower limit (in Da) of usable mass window (default: ‘1500’)
- **upper.threshold**: Upper limit (in Da) of usable mass window (default: ‘7000’)
- **fwd.primer**: Length (in bp) of forward primer
### isAssayable

**Description**

Function to determine whether or not a given molecular weight is assayable (i.e. within the usable mass window specified)

**Usage**

```r
isAssayable(MW, lower.threshold = 1500, upper.threshold = 7000)
```

**Arguments**

- **MW**
  
  Numerical input corresponding to molecular weight

- **lower.threshold**
  
  Lower limit (in Da) of usable mass window (default: ‘1500’)

- **upper.threshold**
  
  Upper limit (in Da) of usable mass window (default: ‘7000’)

**Value**

Returns a logical corresponding to whether or not the molecular weight input falls within the usable mass window specified

---

**Details**

In silico fragmentation analysis includes RNase A digestion, peak mapping and overlap detection, CG detection, assayability and conversion controls.

**Value**

Returns a list of MassArrayFragment objects, each with extensive contextual and other information

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**See Also**

See Also as MassArrayFragment

**Examples**

```r
inSilicoFragmentation("GGTGTAGTCC")
```
MassArray.example.data

Author(s)
Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

isAssayable(5000)

isAssayable(1200)

MassArray.example.data

MassArray Data object

Description

This data contains MassArray spectral information for two samples.

Usage

MassArray.example.data

Format

The format is: Formal class 'MassArrayData' [package "MassArray"] with 17 slots ..@ sequence : chr "CCAGGTCCAAAGGTTCAGACCAGTCTGAACCTGTCCAGGGGCACTCCATATTTTCCTACCTGTCCTTCTTGTGTTGAAAACAAATATACAGGGATCCCAGCAACTTCGGGGCATGTGTGTAACT" | __truncated__ ..@ chr : chr(0) .@ start : int(0) .@ end : int(0) .@ strand : chr "+" .@ fwd.tag : chr "AGGAAGAGAG" .@ rev.tag : chr "AGCCTTCCTCCC" .@ fwd.primer : num 29 .@ rev.primer : num 27 .@ lower.threshold : num 1500 .@ upper.threshold : num 9000 .@ fragments.T : List of 89 ..@ Formal class 'MassArrayFragment' [package "MassArray"] with 21 slots ..@ ID : int 1 ..@ assay.name : chr "" ..@ name : chr "" ..@ sequence : chr "GGGAGAAGGCT" ..@ position : int 385 ..@ length : int 11 ..@ CpGs : int 0 ..@ MW : num 3913 ..@ collisions : int 0 ..@ collision.IDs : List of 1 ..@ MW : num 3913 ..@ collisions : int 0 ..@ CG.collisions : int 0 ..@ CG.collision.IDs : list() ..@ type : chr "T" ..@ direction : chr "+" ..@ extra : chr "5PPP-3P" ..@ bisulfite.converted : logi TRUE ..@ assayable : logi TRUE ..@ conversion.control : logi FALSE ..@ required : logi FALSE ..@ ignored : logi FALSE ..@ primer : logi TRUE ..@ samples : List of 2 ..@ : Formal class 'MassArraySpectrum' [package "MassArray"] with 9 slots ..@ sample : chr "A" ..@ rxn : chr "T" ..@ strand : chr "+" ..@ peaks : List of 184 ..@ : Formal class 'MassArrayPeak' [package "MassArray"] with 16 slots ..@ ID : int 1 ..@ MW.theoretical : num 1111 ..@ MW.actual : num NA ..@ probability : num 0 ..@ SNR : num 0 ..@ height : num NA ..@ sample.intensity : num NA ..@ ref.intensity : num 0.1 ..@ ... components : int 0 ..@ missing : logi TRUE ..@ new : logi FALSE ..@ quality: conversion : num [1:4] 0.0529 0.0 0 0 ...@ quality.spectra : num NA ..@ quality.primerdimer : num [1:7] 8.51 15.83 3.28 1.04 0 ..@ quality.contaminant : num NA ..@ quality addscts : num [1:4] 1.0 0.97 0.231 0.412 ..@ groups : chr(0) ..@ CpG.data : num [1:2, 1:18] 0.0322 0.0449 0.1468 0.3641 0.1468 ... ..@ CpG.data.combined : num [1:2, 1:18] 0.0322 0.0449 0.1468 0.3641 0.1468 ...
MassArrayData-class

Source

Thompson et al. 2009

Examples

data(MassArray.example.data)

Description

A data structure containing MassArray data and associated information for a single amplicon

Objects from the Class

Objects can be created by calls of the form `new("MassArrayData", sequence, file, verbose, fwd.tag, rev.tag, fwd.primer, rev.primer, strand, lower.threshold, upper.threshold, header, skip, sep, comment.char, fill, method, position, ...)`.

Slots

sequence: Nucleotide sequence for unconverted amplicon
chr: Chromosomal position of amplicon
start: Chromosomal position of amplicon
end: Chromosomal position of amplicon
strand: DNA strand used for amplicon (can be '+' or '-')
fwd.tag: Nucleotide tag sequence 5' of the forward primer
rev.tag: T7-containing nucleotide tag sequence 5' of the reverse primer
fwd.primer: Length (in bp) of forward primer
rev.primer: Length (in bp) of reverse primer
lower.threshold: Lower limit (in Da) of usable mass window (default: '1500')
upper.threshold: Upper limit (in Da) of usable mass window (default: '7000')
fragments.T: List containing objects of class MassArrayFragment, corresponding to the T-cleavage reaction for the amplicon on the specified strand
fragments.C: List containing objects of class MassArrayFragment, corresponding to the C-cleavage reaction for the amplicon on the specified strand
samples: List containing object of class MassArraySpectrum, each corresponding to spectral data from a single sample
groups: List of the group name to which each sample belongs
CpG.data: Matrix containing analyzed methylation data, where each row is a sample and each column is a CG dinucleotide site
CpG.data.combined: Matrix containing methylation data combined from multiple objects (or collapsed from within a single object), where each row is a sample and each column is a CG dinucleotide site
MassArrayFragment-class

Methods

\$ signature(x = "MassArrayData"): ...
\$<- signature(x = "MassArrayData"): ...
[ signature(x = "MassArrayData"): ...
initialize signature(.Object = "MassArrayData"): ...

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

showClass("MassArrayData")

MassArrayFragment-class

Class "MassArrayFragment"

Description

A data structure containing information for a single fragment of an amplicon

Objects from the Class

Objects can be created by calls of the form new("MassArrayFragment", ID, sequence, assay.name, name, position, type, direction, extra, bisulfite.converted, assayable, primer, ...).

Slots

ID: Unique integer indexing the fragment’s position within a potential list of multiple fragments
assay.name: (currently not supported)
name: (currently not supported)
sequence: Bisulphite converted nucleotide sequence of fragment
position: Relative position of fragment within the amplicon
length: Length (in bp) of fragment sequence
CpGs: Number of CG dinucleotides contained within the fragment
MW: Predicted molecular weight(s) of fragment, including methylated and unmethylated mass, adducts, etc.
collisions: Number of fragments that share the same molecular weight as the current fragment
collision.IDs: IDs of other fragments that share the same molecular weight as the current fragment
CG.collisions: Number of CG-containing fragments that share the same molecular weight as the current fragment
CG.collision.IDs: IDs of other CG-containing fragments that share the same molecular weight as the current fragment
type: Specifies either 'T' or 'C' cleavage reaction
direction: DNA strand used for fragment sequence (can be '+' or '-')
MassArrayPeak-class

extra: One of "5PPP-3P", "5OH", or "5PPP-3OH" (default)

bisulfite.converted: Logical indicating whether the fragment sequence represents bisulfite converted sequence

assayable: Logical indicating whether or not the fragment molecular weight is within the usable mass window

conversion.control: Logical indicating whether or not the fragment is designated as a potential conversion control

required: Logical indicating whether or not the fragment is designated as 'required' by the user

ignored: Logical indicating whether or not the fragment is to be ignored

primer: Logical indicating whether or not the fragment overlaps with primer or tagged sequence

Methods

\$ signature(x = "MassArrayFragment"): ...
\$<- signature(x = "MassArrayFragment"): ...
initialize signature(.Object = "MassArrayFragment"): ...

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also

See Also MassArrayData

Examples

showClass("MassArrayFragment")

MassArrayPeak-class  Class "MassArrayPeak"

Description

A data structure containing information and data for a single peak from a single spectrum

Objects from the Class

Objects can be created by calls of the form \texttt{new("MassArrayPeak", ID, MW.theoretical, MW.actual, probability, ...)}

Slots

ID: Peak ID indicating indexed position within a potentially large list of peaks

MW.theoretical: Expected molecular weight of peak based on nucleotide sequence

MW.actual: Observed molecular weight from experimental data

probability: Object of class "numeric"

SNR: Signal-to-noise ratio

height: Raw peak height
sample.intensity: Raw sample intensity
ref.intensity: Object of class "numeric"
sequence: Nucleotide composition or sequence(s) corresponding to peak
adduct: One of 'Na', 'K', or " indicating whether or not peak represents a salt adduct of another expected peak
type: Object of class "character"
charge: Degree of ionization of fragment (default is '1' indicating a single positive charge per fragment)
collisions: Number of peaks that share the same molecular weight as the current peak
components: Number of fragments expected to give rise to a peak of this molecular weight
missing: Logical indicating whether or not the expected peak is missing from the spectral data
new: Logical indicating whether or not the observed peak is unexpected given the amplicon sequence

Methods
\$ signature(x = "MassArrayPeak"): ...
\$<- signature(x = "MassArrayPeak"): ...
initialize signature(.Object = "MassArrayPeak"): ...

Author(s)
Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also
See Also MassArrayData

Examples
showClass("MassArrayPeak")
numCollisions

Slots

  sample: Sample name
  rxn: Cleavage reaction (either 'T' or 'C')
  strand: DNA strand for amplicon (either '+' or '-')
  peaks: List containing objects of class MassArrayPeak
  quality.conversion: Overall level(s) of remnant unconverted cytosines, as measured by one or more conversion controls
  quality.spectra: (currently not supported)
  quality.primerdimer: (currently not supported)
  quality.contaminant: (currently not supported)
  quality.adducts: Overall ratio(s) of Na and/or K adduct peak heights to expected peak heights

Methods

$signature(x = "MassArraySpectrum")$ ...
$signature(x = "MassArraySpectrum")$ ...
initialize signature(.Object = "MassArraySpectrum") ...

Author(s)

  Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also

  See Also MassArrayData

Examples

  showClass("MassArraySpectrum")

numCollisions Count peak collisions

Description

  Function to count the number of peak collisions (i.e. molecular weights are indistinguishable given the specified resolution)

Usage

  numCollisions(peaks, resolution = 0.5)

Arguments

  peaks: Numerical list of molecular weights (in Da) corresponding to a set of peaks
  resolution: Resolution (in Da), used to specify the ability to distinguish two different molecular weights. For a resolution of '0.5' (default), two molecular weights are considered identical if they are less than '0.5' Da apart.
Value

Returns a list of peak collision counts for each peak in the original list.

Author(s)

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

Examples

numCollisions(1:5, 1.5)

plot.MassArrayData  Plot MassArrayData

Description

Function to generate graphical output for methylation data in a MassArrayData object

Usage

## S3 method for class 'MassArrayData'
plot(x, ..., collapse = TRUE, bars = TRUE, scale = TRUE, sequence = TRUE, labels = TRUE, colors = TRUE, main = position(x), width = 1.5)

Arguments

x  Object of class MassArrayData

... Other arguments to plot, currently not supported at this time

collapse  Logical specifying whether or not to combine samples by unique group (see MassArrayData). If TRUE, each methylation values are averaged across all samples in each unique group. If FALSE, all samples are retained and plotted individually

bars  Logical specifying whether or not to display error bars. If TRUE (and collapse is TRUE), the median absolute deviation is calculated for each group and plotted as an error bar for each methylation value. If FALSE, no error bars are displayed

scale  Logical specifying whether or not to keep the x axis to scale. If TRUE, methylation values are plotted as a function of relative position within the amplicon sequence. If FALSE, positional information is ignored and methylation values are evenly spaced across the plot window.

sequence  Logical specifying whether or not to display the nucleotide sequence for the amplicon

labels  Logical specifying whether or not to display data labels

colors  Logical specifying whether or not to plot in color. If TRUE, colors are used. If FALSE, plotting occurs in black and white and grayscale.

main  Label/title for overall plot (default is """)

width  Numerical value specifying the display width to use for each methylation value; number corresponds to the number of base pairs to include in both directions from the methylation position (default is 1.5)
**position**

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**See Also**

See Also **MassArrayData**

**Examples**

```r
data(MassArray.example.data)
plot(MassArray.example.data, collapse=FALSE, bars=FALSE, scale=FALSE)
```

---

**Operate on positional information**

**Description**

Function to access (and/or assign) positional information for a MassArrayData object

**Usage**

```r
position(object)
pattern(object) <- value
```

**Arguments**

<table>
<thead>
<tr>
<th>object</th>
<th>Object of class MassArrayData</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Character string containing positional information of the form &quot;chrXX:XXXX-XXXX&quot;</td>
</tr>
</tbody>
</table>

**Value**

Returns a character string containing positional information of the form "chrXX:XXXX-XXXX" if accessing a MassArrayData object. If updating a MassArrayData object, function returns the object with updated positional information

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**See Also**

See Also **MassArrayData**

**Examples**

```r
data(MassArray.example.data)
position(MassArray.example.data)
position(MassArray.example.data) <- "chrB:2001-2374"
position(MassArray.example.data)
```
position-methods  

*Operate on positional information (methods)*

**Description**

Methods to access (and/or assign) positional information for a MassArrayData object

**Methods**

- **object = "MassArrayData"**  Access positional information for MassArrayData object
- **object = "MassArrayData", value = "missing"**  Handle empty function call, simply return the MassArrayData object
- **object = "MassArrayData", value = "character"**  Assign position of MassArrayData object to value

**See Also**

- position

---

**revComplement**  

*Reverse complement*

**Description**

Function to find the reverse complement

**Usage**

```
revComplement(x)
```

**Arguments**

- **x**  sequence input to use for reverse complement. x can be a character string or a MassArrayData object.

**Value**

Returns the reverse complement of a character string or MassArrayData object, depending upon input data type.

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**See Also**

- revComplement-methods

**Examples**

```
revComplement("AATCCGGGGAA")
```
revComplement-methods  Reverse complement (methods)

Description
Methods for reverse complement

Methods

  x = "MassArrayData"  Finds reverse complement of a MassArrayData object, a function that consists of altering sequence, strand, fragmentation, and methylation data
  x = "character"  Calculates reverse complement from nucleotide sequence as character input

Author(s)
Reid F. Thompson (<rthompso@aeom.yu.edu>), John M. Greally (<jgreally@aeom.yu.edu>)

See Also

  revComplement

rnaDigest  RNA digest

Description
Function to perform an in silico RNAse A digest for either the T or C cleavage reactions

Usage

  rnaDigest(sequence, type = c("T", "C"))

Arguments

  sequence  Nucleotide sequence input
  type  One of either 'T' or 'C', specifying which cleavage reaction mixture was used

Value
Returns a list of MassArrayFragment objects, each containing information about a given fragment generated by the RNA digest

Author(s)
Reid F. Thompson (<rthompso@aeom.yu.edu>), John M. Greally (<jgreally@aeom.yu.edu>)

See Also

  See Also as MassArrayFragment
Examples

```
rnaDigest("AAAAACCCCTGCGAGAGAGGCCGACAAAA", type="T")
```

---

**Description**

Function to access (and/or assign) sample name information for a MassArrayData object

**Usage**

```
samples(object)
samples(object) <- value
```

**Arguments**

- **object**: Object of class MassArrayData
- **value**: List of character strings containing sample name information, one for each sample contained in the MassArrayData object

**Value**

Returns a list of character strings containing sample name information for each sample in MassArrayData object. If updating a MassArrayData object, function returns the object with updated sample name information

**Author(s)**

Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

**See Also**

See Also as MassArrayData

**Examples**

```
data(MassArray.example.data)
samples(MassArray.example.data)
samples(MassArray.example.data)[2] <- "SECOND"
samples(MassArray.example.data)
```
### samples-methods

**Operate on sample name(s) (methods)**

**Description**

Methods to access (and/or assign) sample name information for a MassArrayData object

**Methods**

- **object = "MassArrayData"**  
  Access sample name information for MassArrayData object

- **object = "MassArrayData", value = "missing"**  
  Handle empty function call, simply return the MassArrayData object

- **object = "MassArrayData", value = "character"**  
  Assign sample name of MassArrayData object to value

**See Also**

- samples

### sum.MassArraySpectrum

*Sum MassArraySpectrum objects*

**Description**

Function to collapse multiple MassArraySpectrum objects into a single MassArraySpectrum representing the sum of each

**Usage**

```r
## S3 method for class 'MassArraySpectrum'
sum(x, ..., trim = 0, na.rm = TRUE)
```

**Arguments**

- **x**  
  One or multiple MassArraySpectrum objects to include in sum

- **...**  
  Any additional MassArraySpectrum objects to include in sum

- **trim**  
  Numerical value between 0 and 0.5 specifying the proportion of spectra to remove from consideration on a per peak basis, such that the SNR of each peak is calculated as the trimmed mean of the same peak across all included spectra.

- **na.rm**  
  Logical value passed to mean, indicating whether NA values should be stripped before the computation proceeds.

**Value**

Returns a single MassArraySpectrum object that represents the union of all unique peaks from the component MassArraySpectrum objects, with SNR for each peak representing the average value of that peak across all spectra.
Author(s)
Reid F. Thompson (<rthompso@aecom.yu.edu>), John M. Greally (<jgreally@aecom.yu.edu>)

See Also
See Also as MassArraySpectrum

Examples
```r
data(MassArray.example.data)
MassArray.example.data$samples[[1]]$peaks[[11]]$height
MassArray.example.data$samples[[1]] <- sum.MassArraySpectrum(MassArray.example.data$samples[[1]], MassArray.example.data$samples[[1]]$peaks[[11]]$height
```
Index

*Topic **aplot**
  ampliconPrediction, 2
  plot.MassArrayData, 28

*Topic **arith**
  analyzeCpGs, 4
  calcMeth, 5
  calcMW, 6
  calcPercentAdduct, 7
  calcPercentConversion, 8

*Topic **attribute**
  ampliconPrediction, 2
  analyzeCpGs, 4
  calcPercentAdduct, 7
  calcPercentConversion, 8
  evaluateSNPs, 14
  importEpiTyperData, 19
  importEpiTyperData.new, 19
  inSilicoFragmentation, 20
  position, 29
  samples, 32

*Topic **character**
  ampliconPrediction, 2
  bisConvert, 5
  calcMW, 6
  convControl, 10
  countCGs, 11
  expandSequence, 15
  identifySNPs, 18
  inSilicoFragmentation, 20
  revComplement, 30
  rnaDigest, 31

*Topic **classes**
  MassArrayData-class, 23
  MassArrayFragment-class, 24
  MassArrayPeak-class, 25
  MassArraySpectrum-class, 26

*Topic **datasets**
  MassArray.example.data, 22

*Topic **file**
  importEpiTyperData, 19
  importEpiTyperData.new, 19

*Topic **graphs**
  ampliconPrediction, 2

*Topic **list**
  findCollisions, 16
  findFragments, 16
  findPeaks, 17
  numCollisions, 27

*Topic **logic**
  isAssayable, 21

*Topic **manip**
  calcMeth, 5
  combine, 9
  combine-methods, 10
  convControl, 10
  estimate Primer Dimer, 13
  evaluateSNPs, 14
  expandSequence, 15
  findCollisions, 16
  findFragments, 16
  findPeaks, 17
  identifySNPs, 18
  inSilicoFragmentation, 20
  numCollisions, 27
  rnaDigest, 31
  sum.MassArraySpectrum, 33

*Topic **methods**
  combine-methods, 10
  position-methods, 30
  revComplement-methods, 31
  samples-methods, 33

*Topic **print**
  createWiggle, 12
  [,MassArrayData-method
    (MassArrayData-class), 23
  $,MassArrayData-method
    (MassArrayData-class), 23
  $,MassArrayFragment-method
    (MassArrayFragment-class), 24
  $,MassArrayPeak-method
    (MassArrayPeak-class), 25
  $,MassArraySpectrum-method
    (MassArraySpectrum-class), 26
  $<-,MassArrayData-method
    (MassArrayData-class), 23
  $<-,MassArrayFragment-method
    (MassArrayFragment-class), 23
ampliconPrediction, 2
analyzeCpGs, 4
bisConvert, 5
calcMeth, 4, 5, 8
calcMW, 6
calcPercentAdduct, 7
calcPercentConversion, 8
cat, 12
combine, 9, 10
combine, MassArrayData, MassArrayData-method (combine-methods), 10
combine, MassArrayData, missing-method (combine-methods), 10
combine-methods, 10
covControl, 8, 10
countCGs, 11
createWiggle, 12
estimatePrimerDimer, 13
evaluateSNPs, 14
expandSequence, 15
findCollisions, 16
findFragments, 16
findPeaks, 17
identifySNPs, 15, 18
importEpiTyperData, 19
importEpiTyperData.new, 19
initialize, MassArrayData-method (MassArrayData-class), 23
initialize, MassArrayFragment-method (MassArrayFragment-class), 24
initialize, MassArrayPeak-method (MassArrayPeak-class), 25
initialize, MassArraySpectrum-method (MassArraySpectrum-class), 26
inSilicoFragmentation, 20
isAssayable, 21
MassArray.example.data, 22
MassArrayData, 9, 13, 25–29, 32
MassArrayData-class, 23
MassArrayFragment, 11, 21, 31
MassArrayFragment-class, 24
MassArrayPeak, 6
MassArrayPeak-class, 25
MassArraySpectrum, 19, 20, 34
MassArraySpectrum-class, 26
numCollisions, 27
plot.MassArrayData, 28
position, 29, 30
position, MassArrayData-method (position-methods), 30
position-methods, 30
position<-(position), 29
position<-, MassArrayData, character-method (position-methods), 30
position<-, MassArrayData, missing-method (position-methods), 30
revComplement, 30, 31
revComplement, character-method (revComplement-methods), 31
revComplement, MassArrayData-method (revComplement-methods), 31
revComplement-methods, 31
rnaDigest, 31
samples, 32, 33
samples, MassArrayData-method (samples-methods), 33
samples-methods, 33
samples<-(samples), 32
samples<-, MassArrayData, character-method (samples-methods), 33
samples<-, MassArrayData, missing-method (samples-methods), 33
sum.MassArraySpectrum, 33
write, 12