Package 'rnaSeqMap'

October 18, 2017

Type Package

Title rnaSeq secondary analyses

Version 2.34.0

Date 2014-09-30

Author Anna Lesniewska <alesniewska@cs.put.poznan.pl>; Michal Okoniewski <michal@fgcz.ethz.ch>

Maintainer Michal Okoniewski <michal@fgcz.ethz.ch>

Depends R (>= 2.11.0), methods, Biobase, Rsamtools, GenomicAlignments

Imports GenomicRanges , IRanges, edgeR, DESeq, DBI

Description The rnaSeqMap library provides classes and functions to analyze the RNA-sequencing data using the coverage profiles in multiple samples at a time

License GPL-2

Collate zzz.R utils.R plots.R NucleotideDistr.R SeqReads.R NDplots.R NDtransforms.R bam2sig.R parseGff3.R pipelines.R normalizations.R measures.R generators.R camelWrapper.R

biocViews Annotation, ReportWriting, Transcription, GeneExpression, DifferentialExpression, Sequencing, RNASeq, SAGE, Visualization

NeedsCompilation yes

R topics documented:

`addBamData` *addBamData - getting sample data from BAM file.*

Description

Add data from experimental samples stored in BAM file.

Usage

```
addBamData(rs, file, exp, phenoDesc=NULL)
```

Arguments

`rs`	SeqReads object to modify
`file`	BAM file to read
`exp`	Numbers of sample slot in the object
`phenoDesc`	A vector to add to phenoData

Value

SeqReads object with samples added from the BAM files. List of BAM files comes from the covdesc. The covdesc content becomes phenoData of the object.

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    rs <- addBamData(rs,1:3)
#  }
```

`addDataToReadset` *addDataToReadset - adding one more sample in the SeqRead on R level*

Description

Add another reads matrix to the readset. No control of region consistency, the matrix needs just 2 columns: starts and ends.

Usage

```
addDataToReadset(rs, datain, spl)
```

Arguments

`rs`	
`datain`	
`spl`	Number or name of the experimental sample

Value

SeqReads object with one more sample added.

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  rs <- newSeqReads(1,1,20000,1)
#  my.data1 <- rbind(c(1,50), c(3,53), c(11,60))
#  rs <- addDataToReadset(rs, my.data1, 1)
```

addExperimentsToReadset *addExperimentsToReadset - getting sample data from the database.*

Description

Add data from experimental samples in the xXMAP database to the readset. Connection to the database required.

Usage

```
addExperimentsToReadset(rs, exps)
```

Arguments

rs	SeqReads object to modify
exps	Vector of numbers of experimental samples in xXMAP

Value

SeqReads object with samples added from the database.

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    rs <- addExperimentsToReadset(rs,1:3)
#  }
```

averageND *averageND, sumND, combineNS, log2ND - operations on distributions*

Description

Set of functions to operate on NucleotideDistr objects.

averageND calculates the mean for samples, sumND adds up selected samples' distributions, combineND adds two objects with the same size of distribution matrix, log2ND transforms all numeric data in the object into log space.

Usage

```
averageND(nd, exps);
sumND(nd, exps);
combineND(nd1, nd2);
log2ND(nd);
```

Arguments

nd, nd1, nd2	NucleotideDistr objects
exps	a pair of numbers of samples in the experiment

Value

NucleotideDistr object of the same type as input objects

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    nd.cov <- getCoverageFromRS(rs,1:3)
#    nd.avg <- averageND(nd.cov,c(1,3))
#    nd.sum <- averageND(nd.cov,c(1,3))
#    nd.sum <- combineND(nd.cov,nd.cov)
#    nd.log <- log2ND(nd.cov)
#  }
```

---

bam2sig *bam2sig - encapsulated pipeline of finding significant expression*

Description

Reads BAM files according to annotation and produces output table processed with DESeq negative binomial test.

Usage

```
bam2sig(annotlib, covdesc="covdesc", species=NULL, level="gene")
```

Arguments

annotlib	Character table or data frame with colums: chr, start, end, strand, name
covdesc	Name of the file that includes BAM files (experiment description file)
species	Species name - needed for .chr.convert function - to match BAM and annotation chromosome names
level	The level of annotation for calculating the counts: gene, transcript of exon

Value

Output table with significant expression results, as from DESeq

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
if (1==0)
{
 all.g <- all.genes(as.vector=F)
 ss <- sample(1:20000, 10)
 genes <- as.data.frame(all.g[ss,])

 deseqRes <- bam2sig("cassava.db")
 deseqRes[1:10,]
}
```

`buildDESeq` *buildDESeq - create CountDataSet*

Description

Creates `CountDataSet` from the data in the database using the list of genes supplied - for further analysis with DESeq

Usage

```
buildDESeq(genes,exps,conds=NULL)
```

Arguments

`genes`	vector of Ensembl gene IDs
`exps`	vector of experiments
`conds`	Vector of experimental condition descriptions for the samples

Value

`CountDataSet` object filled with the data of gene-level counts of reads

Author(s)

Michal Okoniewski, Anna Lesniewska

See Also

buildDGEList

Examples

```
#  if (xmapConnected())
#  {
#    data(sample_data_rnaSeqMap)
#    gg <- names(rs.list)
#    cds <- buildDESeq(gg,1:6, c("a","b","b","a","a","b"))

# }
```

buildDGEList *buildDGEList - create DGEList (edgeR)*

Description

Creates DGEList from the data in the database using the list of genes supplied - for further analysis with edgeR

Usage

```
buildDGEList(genes,exps,conds=NULL)
```

Arguments

genes	vector of Ensembl gene IDs
exps	vector of experiments
conds	Vector of experimental condition descriptions for the samples

Value

DGEList object filled with the data of gene-level counts of reads

Author(s)

Michal Okoniewski, Anna Lesniewska

See Also

buildDESeq

Examples

```
#  if (xmapConnected())
#  {
#    data(sample_data_rnaSeqMap)
#    gg <- names(rs.list)
#    cds <- buildDGEList(gg,1:6, c("a","b","b","a","a","b"))
#  }
```

findRegionsAsIR *findRegionsAsIR - finding regions of high coverage using Lindell-Aumann algorithm.*

Description

The function is running Lindell-Aumann algorithm to find regions of irreducible expression on the coverage data in the NucleotideDistr object. The function may be used to find the location and boundaries of significant expression of exons and small RNA.

Usage

```
findRegionsAsIR(nd, mi, minsup=5, exp)
```

Arguments

nd	An object of NucleotideDistr class that has coverage values for a given region
mi	The threshold of coverage that makes the region significant
minsup	Minimal support of the numeric association rule - namely, in this case, the mininmal length of the discovered region
exp	Sample (experiment) number

Value

IRanges object with irreducible regions with high coverage.

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    rs <- addExperimentsToReadset(rs,1:3)
#    nd.cov <- getCoverageFromRS(rs,1:3)
#    nd.regs <- findRegionsAsND(nd.cov, 10)
#  }
```

---

findRegionsAsND — *findRegionsAsND - finding regions of high coverage using Lindell-Aumann algorithm.*

Description

The function is running Lindell-Aumann algorithm to find regions of irreducible expression on the coverage data in the NucleotideDistr object. The function may be used to find the location and boundaries of significant expression of exons and small RNA.

Usage

```
findRegionsAsND(nd, mi, minsup=5)
```

Arguments

nd	An object of NucleotideDistr class that has coverage values for a given region
mi	The threshold of coverage that makes the region significant
minsup	Minimal support of the numeric association rule - namely, in this case, the mininmal length of the discovered region

Value

NucleotideDistr object that includes a matrix with zeros where no region was found and the value of mi for all the nucleotides included in the region. The type fo the object is "REG".

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    rs <- addExperimentsToReadset(rs,1:3)
#    nd.cov <- getCoverageFromRS(rs,1:3)
#    nd.regs <- findRegionsAsND(nd.cov, 10)
#  }
```

`fiveCol2GRanges` *fiveCol2GRanges*

Description

Set of functions to operate on `NucleotideDistr` objects.

averageND calculates the mean for samples, sumND adds up selected samples' distributions, combineND adds two objects with the same size of distribution matrix, log2ND transforms all numeric data in the object into log space.

Usage

```
fiveCol2GRanges(t)
```

Arguments

`t`	A matrix or data frame including genomic regions in 5 columns: ID, chr/contig name, start, end, strand

Value

GenomicRanges object with the same values

Author(s)

Michal Okoniewski

geneInChromosome *geneInChromosome*

Description

Finds all the genes in the given chromosome regions

Usage

```
geneInChromosome(chr, start, end, strand)
```

Arguments

chr	Chromosome
start	Start of the region on a chromosome
end	End of the region on a chromosome
strand	Genome strand: 1 or -1

Value

table of the genes in a given regions, produced with stored procedure

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    geneInChromosome(1, 1, 80000, 1)
#  }
```

generators *Generators for synt data and*

Description

Various generators for experiments.

Usage

```
generatorAddSquare(nd, deg, length.prop=0.5)
generatorAdd(nd, deg, length.prop=0.5)
generatorMultiply(nd, deg, length.prop=0.5)
generatorTrunc(nd,deg)
generatorSynth(nd, deg, length.prop=0.5)
generatorPeak(nd, deg, sr=10, mult=10)
```

Arguments

nd	nucleotide distribution object
deg	degeneration level for the output profile
length.prop	a fraction of the genome region to be degenerated - (0,1)
sr	distance from the 5' end for the peak
mult	multiplier - how many times the peak is supposed to be higher than the maximum of the distribution
	Generators of synthetic and semi-synthetic coverage profiles, for RNA-seq measures testing.

Author(s)

Anna Lesniewska,Michal Okoniewski

Examples

```
if (1==0)
{
rs <- newSeqReads('chr2', 220238268, 220254744, -1)
f <- c("test1.bam", "test2.bam", "test3.bam", "test4.bam", "test5.bam")
ff <- sapply(f, function(x) system.file("extdata", x, package = "rnaSeqMap"))
rs <- getBamData(rs, 1:5)
nd <- getCoverageFromRS(rs, 1:5)
generatorTrunc(nd,0.5)
}
```

---

getBamData *getBamData - getting sample data from BAM file.*

Description

Add data from experimental samples stored in BAM file.

Usage

```
getBamData(rs, exps = NULL, cvd = NULL, covdesc.file = "covdesc")
```

Arguments

rs	SeqReads object to modify
exps	Vector of numbers of experimental samples
cvd	Covdesc-like data frame - BAM files are read from row names
covdesc.file	Alternatively, the experiment description file

Value

SeqReads object with samples added from the BAM files. List of BAM files comes from the covdesc. The covdesc content becomes phenoData of the object.

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    rs <- getBamData(rs,1:3)
#  }
```

`getCoverageFromRS` *getCoverageFromRS - conversion to coverage object*

Description

Calculates the coverage function for the reads encapsulated in the SeqReads object.

Usage

```
getCoverageFromRS(rs, exps)
```

Arguments

`rs`	SeqReads object to modify
`exps`	Vector of numbers of experimental samples in xXMAP

Value

NucleotideDistr object with coverage matrix in assayData slot and type "COV".

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    rs <- addExperimentsToReadset(rs,1:6)
#    nd.cov <- getCoverageFromRS(rs,1:3)
#  }
```

getData — *Data accessor function for rnaSeqMap objects containing 'data' field*

Description

This function gets the 'data' field from rnaSeqMap objects

Usage

```
getData(iND)
```

Arguments

iND	rnaSeqMap object containing 'data' field

Value

A list containing 'data' field

Author(s)

Michal Okoniewski, Anna Lesniewska, Marek Wiewiorka

getExpDescription — *getExpDescription*

Description

Gets the bio_sample table from the database. May be used as phenoData.

Usage

```
getExpDescription()
```

Value

Table of experimental factors assigned to numbers of samples.

Author(s)

Michal Okoniewski, Anna Lesniewska

getFCFromND *getFCFromND - calculating fold change of coverages*

Description

This function calculates the fold change of two sample coverages from a NucleotideDistr objects. The coverages are assumed to be after logarithmic transformation, so the function basically subtracts the value and generates new NucleotideDistr object with a single vector of fold changes.

Usage

```
getFCFromND(nd, exps)
```

Arguments

nd	NucleotideDistr object with coverages
exps	a pair of numbers of samples in the experiment

Value

NucleotideDistr object of type "FC" with a single vector of fold changes

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    rs <- addExperimentsToReadset(rs,1:3)
#    nd.cov <- getCoverageFromRS(rs,1:3)
#    nd.fc <- getFCFromND(nd.cov,c(1,3))
#  }
```

getSIFromND *getSIFromND - calculating splicing index of two coverages*

Description

This function calculates the splicing index value of two sample coverages from a NucleotideDistr object. It is assumed that the region in the NucleotideDistr is a single gene. Splicing index is calculated in similar way to the implementation for exon Affy microarrays (see Gardina et al, Genome Biology, 2007 for details), but it is run for each nucleotide in the region and instead of gene-level average expression values, it uses sums of reads for both samples.

Usage

```
getSIFromND(nd, exps)
```

Arguments

nd	NucleotideDistr object with coverages
exps	a pair of numbers of samples in the experiment

Value

NucleotideDistr object of type "FC" with a single vector of splicing index values

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,1,20000,1)
#    nd.cov <- getCoverageFromRS(rs,1:3)
#    nd.fc <- getSIFromND(nd.cov,c(1,3))
#  }
```

---

getSumsExp *getSumsExp*

Description

Gets the sum of reads in all the samples present in the database in the seq_read table

Usage

```
getSumsExp()
```

Value

Vector of sums

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#   sums <- getSumsExp()
#   nsums
#  }
```

gRanges2CamelMeasures *Genomic plots based upon NucleotideDistr objects*

Description

Various plots of genomic coverage for data from NucleotideDistr objects

Usage

```
gRanges2CamelMeasures(gR, cvd,  sample.idx1, sample.idx2, sums=NULL, progress=NULL)
allCamelMeasuresForRegion(ch, st, en, str, cvd, sample.idx1, sample.idx2, sums=NULL)
```

Arguments

ch	chromosome name
st	genomic start
en	genomic end
str	strand
cvd	name of the file with BAM description - covdesc
gR	GenomicRanges object to use as a set of genomic regions to query
sample.idx1,sample.idx2	sample indices
sums	the vector of sums for normalization
progress	every how many regions print a dot for progress indicator

Author(s)

Michal Okoniewski

Examples

```
#
```

measures *Measures*

Description

Various measures to find differential expression.

Usage

```
ks_test(dd)
diff_area(dd, cconst)
diff_derivative_area(dd, cconst)
qq_plot(dd)
qq_derivative_plot(dd)
pp_plot(dd)
pp_derivative_plot(dd)
hump_diff1(dd)
hump_diff2 (dd)
```

Arguments

dd	a matrix with 2 columns for samples and rows for nucleotides, containing coverage data (like from BED files)
cconst	NULL default
	The measures give various assesment of the difference between two sequencing samples shapes. Full description will follow in the paper.

Author(s)

Anna Lesniewska,Michal Okoniewski

Examples

```
# if (xmapConnected())
#  {
#   ks_test(dd)
#  }
```

NDplots *Genomic plots based upon NucleotideDistr objects*

Description

Various plots of genomic coverage for data from NucleotideDistr objects

Usage

```
distrCOVPlot(nd,exps)
distrSIPlot(nd, ex1, ex2, mi,minsup=5)
```

Arguments

nd	NucleotideDistr object
exps	vectors of experiment numbers to plot
ex1,ex2	experiment numbers to plot
mi	threshold in the region mining algorithm
minsup	minimal support - minimal length of the irreducible region found

Author(s)

Michal Okoniewski, Anna Lesniewska

`normalizations` *Normalization Methods*

Description

Various normalization methods.

Usage

```
standarizationNormalize(nd)
min_maxNormalize(nd)
densityNormalize(nd)
globalCountsNormalize(nd, sums)
```

Arguments

`nd`	nucleotide distribution object
`sums`	sum of reads in a sequencing sample

Normalizations of a single coverage profile for multiple samples contained in the `NucleotideDistr` object. Full description will follow in a paper.

Author(s)

Anna Lesniewska,Michal Okoniewski

Examples

```
# if (xmapConnected())
#  {
#  s <- newSeqReads('chr2', 220238268, 220254744, -1)
#  f <- c("test1.bam", "test2.bam", "test3.bam", "test4.bam", "test5.bam")
#  ff <- sapply(f, function(x) system.file("extdata", x, package = "rnaSeqMap"))
#  rs <- getBamData(rs, 1:5, files = ff)
#  nd <- getCoverageFromRS(rs, 1:5)
#  min_maxNormalize(nd)
#  }
```

normalizeBySum *Normalization of NucleotideDistr by global number of reads*

Description

normalizeBySum function normalizes the coverage values in NucleotideDistr by dividing all the numbers for all samples by the sum of reads for each sample. The number of reads from each sample may be taken from the database by the function getSumsExp, which is a wrapper for an appropriate SQL procedure. Alternatively, it is passed directly as a vector of numeric values of the same length as the number of samples analyzed. Such simple normalization allows comparisons of the coverage values for samples with different number of reads

Usage

```
normalizeBySum(nd,r=NULL)
```

Arguments

nd	NucleotideDistr object with raw read counts
r	Vector of numbers. If there is no such parameter, a database procedure summarizing reads is run

Value

NucleotideDistr object

Author(s)

Michal Okoniewski, Anna Lesniewska

See Also

getSumsExp

Examples

```
#  if (xmapConnected())
#  {
#    rs <- newSeqReads(1,10000,20000,1)
#    nd.cov <- getCoverageFromRS(rs,1:3)
#    nd.norm <- normalizeBySum(nd.cov)
#    nd.norm <- normalizeBySum(nd.cov, r=c(100, 200, 1000))
#  }
```

NucleotideDistr-class *Numeric distributions by nucleotide - class*

Description

An S4 class that inherits from eSet and holds all the numeric distributions of functions defined over the genome. The values may include coverage, splicing, fold change, etc. for a region defined by genomic coordinates.

Slots/List Components

Objects of this class contain (at least) the following list components:

chr: numeric matrix containing the read counts.

start: data.frame containing the library size and group labels.

end: data.frame containing the library size and group labels.

strand: data.frame containing the library size and group labels.

start: data.frame containing the library size and group labels.

Methods

distribs gives the matrix of distributions from assayData

getDistr gives a single distributions from assayData as a vector

newNuctleotideDistr (distribs, chr, start, end, strand, type="UNKNOWN", phenoData=NULL, featureDa
constructor from a matrix of data and chromosome coordinates.

Author(s)

Anna Lesniewska, Michal Okoniewski

See Also

SeqReads, NDtransforms

parseGff3 *parseGff3 - parsing gff3 file format*

Description

Parses gff3 file into genes, transcripts and exons.

Usage

```
parseGff3(filegff, fileg="genes.txt", filet="transcripts.txt", filee="exons.txt", nofiles=FALSE)
```

Arguments

filegff	Input file in GFF3 format
fileg	Filename for output: genes
filet	Filename for output: transcripts
filee	Filename for output: exons
nofiles	Flag: just optput list, no files

Value

List with elements "genes", "transcripts", "exons" with appropriate tables.

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#     parseGff3("Athaliana.gff3")
#  }
```

---

plotGeneCoverage *Genomic plots with rnaSeqMap*

Description

Various plots of genomic coverage for experiments.

Usage

```
plotGeneCoverage(gene_id, ex)
plotRegionCoverage(chr, start, end, strand, ex)
plotExonCoverage (exon_id,ex)
plotCoverageHistogram (chr,start,end,strand,ex, skip)
plotGeneExonCoverage(gene_id, ex)
plotSI(chr,start,end,strand, exp1, exp2 )
```

Arguments

ex	vectors of experiment numbers to plot
exp1, exp2	experiment numbers for splicing index
gene_id	Ensembl gene ID
exon_id	Ensembl exon ID
chr	Chromosome
start	Start position of region on the chromosome
end	Start position of region on the chromosome
strand	Strand
skip	size of the bucket in histogram

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    plotGeneCoverage( "ENSG00000144567", 1:3) # plotting FAM134A for experiments 1,2,3
#    plotRegionCoverage( 2, 220040947, 220050201, 1, 1:3 ) # the same, using coordinates
#  }
```

`readsInRange` *readsInRange*

Description

Finds all the reads for a genomic range

Usage

```
readsInRange(chr, start, end, strand, ex)
```

Arguments

`chr`	Chromosome
`start`	Start of the region on a chromosome
`end`	End of the region on a chromosome
`strand`	Genome strand: 1 or -1
`ex`	experiment

Value

table of reads, as in the database

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
 # {
#   tmp <- readsInRange( 1, 10000, 20000, 1,3)
#  }
```

`regionBasedCoverage` *regionBasedCoverage - transformation of the region coverage by the Lindell-Aumann regions*

Description

The function builds a `NucleotideDistr` object from another object of coverage, using sequential call of Lindell-Aumann algorithm on the same data with a sequence of mi-levels. Each nucleotide is assigned the maximum mi-value of a region that covers it.

The output `NucleotideDistr` object has the distribution without peaks and small drops of coverage, but the thade-off is that the level of coverage are discrete: seq*maxexp.

Usage

```
regionBasedCoverage(nd, seqq=1:10, maxexp=20, minsup=5)
```

Arguments

`nd`	An object of `NucleotideDistr` class that has coverage values for a given region
`seqq`	Vector of numbers used to divide the range of coverage for subsequent mi-levels
`maxexp`	The maximal mi-level for coverage
`minsup`	Minimal support of the numeric association rule - namely, in this case, the mininmal length of the discovered region

Value

NucleotideDistr object that includes a matrix with zeros where no region was found and a maximum of mi-levels used for the sequential region searched. The distributions are similar to coverage, but have removed outliers of coverage peaks and short drops of coverage.

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#  rs <- newSeqReads(1,1,20000,1)
#  rs <- addExperimentsToReadset(rs,1:3)
#  nd.cov <- getCoverageFromRS(rs,1:3)
#  nd.regs <- regionBasedCoverage(nd.cov, 1:10, 100)
  #runs the Lindell-Aumann algorithm at 100, 90, ... and picks maximal mi-level, where the nucleotide has a re
#  }
```

regionCoverage *regionCoverage*

Description

Finds all the reads for a genomic range

Usage

```
regionCoverage(chr, start, end, strand, ex, db = "FALSE" )
```

Arguments

chr	Chromosome
start	Start of the region on a chromosome
end	End of the region on a chromosome
strand	Genome strand: 1 or -1
ex	experiment
db	Use database (SQL) implementation of the algorithm

Value

coverage vector, independent from NucleotideDistr

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#   tmp <- regionCoverage( 1, 10000, 20000, 1,3)
#  }
```

RleList2matrix *RleList2matrix*

Description

Function transforms list of Rle objects to matrix.

Usage

```
RleList2matrix(ll);
```

Arguments

ll	list of Rle objects.

Value

Produces the full, unpacked coverage matrix from a list of Rle objects. Used to re-format the coverage data.

Author(s)

Michal Okoniewski, Anna Lesniewska

`rs.list` *Example of sequencing data for rnaSeqMap library*

Description

A fragment of sequencing data from 6 samples - human.

Usage

```
data(sample_data_rnaSeqMap)
```

Format

A list with 17 SeqReads objects,each with sequencing reads from 6 samples sequenced with ABI SOLID machine.

Examples

```
#  data(sample_data_rnaSeqMap)
#  length(rs.list)
#  gene1rs <- rs.list[[1]]
```

`SeqReads` *SeqReads - a container for RNAseq reads*

Description

SeqReads objects keep the reads information in the form of a list, containing one matrix of reads per experiment. Matrices of dimension n x 2 should come from a mapping to the regions defined by genome coordinates (chromosome, start, end, strand) in the SeqReads object.

The object may be filled in from the database or from list with read data. It is recommended to create one SeqReads object per gene or intergenic region. The object are used then ot create object of class NucleotideDistr

Usage

```
newSeqReads(chr, start, end, strand, datain=NULL, phenoData=NULL, featureData=NULL, covdesc=NULL)
newSeqReadsFromGene(g)
```

Arguments

chr	Chromosome
start	Start of the region on a chromosome
end	End of the region on a chromosome
strand	Genome strand: 1 or -1
datain	If supplied, it must be a list of matrices of reads start and stop
g	Ensembl identifier of a gene
phenoData	
featureData	
covdesc	Filename for experiment description

Value

Object of a class SeqReads

Author(s)

Michal Okoniewski, Anna Lesniewska

---

setData *Data accessor function for rnaSeqMap objects containing 'data' field*

Description

This function sets the 'data' field from one rnaSeqMap object with 'data' field from the other one.

Usage

```
setData(iND1,iND2)
```

Arguments

iND1	target rnaSeqMap object containing 'data' field
iND2	source rnaSeqMap object containing 'data' field

Value

NULL

Author(s)

Michal Okoniewski, Anna Lesniewska, Marek Wiewiorka

setSAXPYData *Data accessor function for rnaSeqMap objects containing 'data' field*

Description

This function sets the 'data' field at i position. The new value is the old one multiplied by a iParam.

Usage

```
setSAXPYData(iND1,iParam,i)
```

Arguments

iND1	rnaSeqMap object containing 'data' field
iParam	Scalining parameter
i	Index of the 'data' field to be modified

Value

NULL

Author(s)

Michal Okoniewski, Anna Lesniewska, Marek Wiewiorka

setSpecies *setSpecies*

Description

Sets the species name for chromosomes X, Y and MT translation into consecutive numbers. If you use `xmap.connect`, no need to call `setSpecies`. Both set the internal variable of `xmapcore`.

Usage

```
setSpecies(name=NULL)
```

Arguments

name	Species name

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
setSpecies("mus_musculus")
```

simplePlot *simplePlot - quick plot for the coverages*

Description

Plots 2 or 3 coverages with fixed colors.

Usage

```
simplePlot (nd, exps, xlab="genome coordinates", ylab="coverage")
```

Arguments

nd	NucleotideDistr object to plot
exps	Samples to plot - numeric vector
xlab	
ylab	

Author(s)

Michal Okoniewski

spaceInChromosome *spaceInChromosome*

Description

Finds all the intergenic spaces in the given chromosome region

Usage

```
spaceInChromosome(chr, start, end, strand)
```

Arguments

chr	Chromosome
start	Start of the region on a chromosome
end	End of the region on a chromosome
strand	Genome strand: 1 or -1

Value

table of the intergenic spaces in a given regions, produced with stored procedure

Author(s)

Michal Okoniewski, Anna Lesniewska

Examples

```
#  if (xmapConnected())
#  {
#    spaceInChromosome(1, 1, 80000, 1)
#  }
```

Index