Package 'ELMER'

March 17, 2015

Title Inferring Regulatory Element Landscapes and Transcription Factor Networks Using Cancer Methylomes

Version 0.0.0.9005

Author Lijing Yao<lijingya@usc.edu>

Maintainer Lijing Yao<lijingya@usc.edu>

Description
ELMER is designed to use DNA methylation and gene expression from a large number of samples to infere regulatory element landscape and transcription factor network in primary tissue.

Depends R (>= 3.0.1), GenomicRanges

License GPL-3

LazyData true

Imports GenomicRanges,ggplot2,reshape,grid,IRanges,GenomeInfoDb,S4Vectors

Suggests parallel, snow, BiocStyle

biocViews DNAMethylation, GeneExpression, MotifAnnotation, Software, GeneRegulation

R topics documented:

`Binary` *binary data*

Description

binary data

Usage

```
Binary(x, Break = 0.3, Break2 = NULL)
```

Arguments

`x`	A matrix.
`Break`	A value to binarize the data.
`Break2`	A value to cut value to 3 categories.

Value

A binarized matrix.

fetch.mee *fetch.mee*

Description

fetch.mee

Usage

```
fetch.mee(meth, exp, sample, probeInfo, geneInfo, probes = NULL,
  genes = NULL, TCGA = FALSE)
```

Arguments

meth	A matrix or path of rda file only containing a matrix of DNA methylation data.
exp	A matrix or path of rda file only containing a matrix of expression data.
sample	A data frame or path of rda file only containing sample information in data frame format.
probeInfo	A GRnage object or path of rda file only containing a GRange of probe information
geneInfo	A GRnage object or path of rda file only containing a GRange of gene information (Coordinates, GENEID and SYMBOL)
probes	A vector lists probes' name. If probes are specified, the methylation and probeInfo will only contain this list of probes.
genes	A vector lists genes' ID. If gene are specified, the methylation and probeInfo will only contain this list of probes.
TCGA	A logical. FALSE indicate data is not from TCGA (FALSE is default). TRUE indicates data is from TCGA and sample section will automatically filled in.

Value

MEE.data object

Examples

```
meth <- matrix(data=c(1:20),ncol=5,dimnames=list(paste0("probe",1:4),paste0("sample",1:5)))
exp <- matrix(data=c(101:110),ncol=5,dimnames=list(c("gene1","gene2"),paste0("sample",1:5)))
mee <- fetch.mee(meth=meth, exp=exp)
## only fetch probe 1 and 3
mee <- fetch.mee(meth=meth, exp=exp, probes=c("probe1","probe3"))
## only fetch gene 1
mee <- fetch.mee(meth=meth, exp=exp, genes="gene1")
```

fetch.pair *fetch.pair*

Description

fetch.pair

Usage

```
fetch.pair(pair, probeInfo, geneInfo)
```

Arguments

pair	A data.frame or path of csv file containing pair information.
probeInfo	A GRnage object or path of rda file only containing a GRange of probe information
geneInfo	A GRnage object or path of rda file only containing a GRange of gene information (Coordinates, GENEID and SYMBOL)

Value

pair.data object

Examples

```
df <- data.frame(Probe=c("cg19403323","cg12213388","cg26607897"),
GeneID =c("ID255928","ID84451","ID55811"),
Symbol =c("SYT14","KIAA1804","ADCY10"),
Pe=c(0.003322259,0.003322259,0.003322259))
geneInfo <- system.file("extdata","UCSC_gene_hg19.rda",package = "ELMER")
## input can be a path
pair <- fetch.pair(pair = df, geneInfo=geneInfo)
```

get.diff.meth *get.diff.meth*

Description

get.diff.meth

Usage

```
get.diff.meth(mee, diff.dir = "both", cores = NULL, percentage = 0.2,
  pvalue = 0.01, sig.dif = 0.3, dir.out = "./", save = TRUE)
```

Arguments

mee	A MEE.data object containing at least meth and probeInfo.
diff.dir	A character showing differential methylation dirction. It can be "hypo" which is only selecting hypomethylated probes; "hyper" which is only selecting hypermethylated probes; "both" which select both hypomethyalted and hypermethylated probes.
cores	A interger which defines number of core to be used in parallel process. Default is NULL: don't use parallel process.
percentage	A number ranges from 0 to 1 specifying the percentage of samples used to identify the differential methylation. Default is 0.2.
pvalue	A number specify the significant Pvalue cutoff for significant hypo/hyper-methylated probes. Default is current directory.
sig.dif	A number specify the significant methylation difference cutoff for significant hypo/hyper-methylated probes. Default is 0.3.
dir.out	A path specify the directory for outputs. Default is is current directory.
save	A logic. When TRUE, output file will be saved.

Value

Statistics for all probes and significant hypo or hyper-methylated probes.

Examples

```
load(system.file("extradata","mee.example.rda",package = "ELMER"))
Hypo.probe <- get.diff.meth(mee, diff.dir="hypo") # get hypomethylated probes
```

---

get.enriched.motif *get.enriched.motif*

Description

get.enriched.motif

Usage

```
get.enriched.motif(probes.motif, probes, background.probes, lower.OR = 1.1,
  min.incidence = 10, dir.out = "./", label = NULL)
```

Arguments

probes.motif	A matrix contains motifs occurrence within probes regions.
probes	A vector lists the probes' names in which motif enrichment will be calculated.
background.probes	A vector list of probes' names which are considered as background for motif.enrichment calculation.
lower.OR	A number specify the lower boundary of Odds ratio which defines the significant enriched motif. 1.1 is default.

min.incidence	A non-negative integer specify the minimum incidence of motif in the given probes set. 10 is default.
dir.out	A path specify the directory for outputs. Default is current directory
label	A character labels the outputs.

Value

A list contains enriched motifs with the probes regions harboring the motif.

Examples

```
probes <- c("cg00329272","cg10097755","cg08928189", "cg17153775","cg21156590",
"cg19749688","cg12590404","cg24517858","cg00329272","cg09010107",
"cg15386853", "cg10097755", "cg09247779","cg09181054","cg19371916")
load(system.file("extradata","mee.example.rda",package = "ELMER"))
bg <- rownames(getMeth(mee))
enriched.motif <- get.enriched.motif(probes=probes,background.probes = bg,
min.incidence=2, label="hypo")
```

---

get.feature.probe *get.feature.probe*

Description

get.feature.probe

Usage

```
get.feature.probe(probe, distal = TRUE, feature, TSS,
  TSS.range = list(upstream = 2000, downstream = 2000), rm.chr = NULL)
```

Arguments

probe	A GRange object containing probes coordinate information. Default is Illumina-methyl-450K probes coordinates.
distal	A logical. If FALSE, function will output the all probes overlaping with features. If TRUE, function will ouput the distal probes overlaping with features.
feature	A GRange object containing biofeature coordinate such as enhancer coordinates. Default is comprehensive genomic enhancer regions from REMC and FANTOM5.
TSS	A GRange object containing the transcription start site. Default is UCSC gene TSS.
TSS.range	A list specify how to define promoter regions. Default is upstream =2000bp and downstream=2000bp.
rm.chr	A vector of chromosome need to be remove from probes such as chrX chrY or chrM

Value

A GRange object containing probes that satisfy selecting critiria.

Examples

```
# get distal enhancer probe
Probe <- get.feature.probe()
# get distal enhancer probe remove chrX chrY
Probe2 <- get.feature.probe(rm.chr=c("chrX", "chrY"))
```

`get.pair` *get pair*

Description

get pair

Usage

```
get.pair(mee, probes, nearGenes, percentage = 0.2, permu.size = 1000,
  permu.dir = NULL, Pe = 0.01, dir.out = "./", cores = NULL,
  label = NULL)
```

Arguments

`mee`	A MEE.data object containing at least meth, exp, probeInfo, geneInfo.
`probes`	A vector lists the probes' name that need to be linked to genes.
`nearGenes`	Can be either a list containing output of GetNearGenes function or path of rda file containing output of GetNearGenes function.
`percentage`	A number ranges from 0 to 1 specifying the percentage of samples used to link probes to genes. Default is 0.2.
`permu.size`	A number specify the times of permuation. Default is 1000.
`permu.dir`	A path where the output of permuation will be.
`Pe`	A number specify the empircal pvalue cutoff for defining signficant pairs. Default is 0.01
`dir.out`	A path specify the directory for outputs. Default is current directory
`cores`	A interger which defines number of core to be used in parallel process. Default is NULL: don't use parallel process.
`label`	A character labels the outputs.

Value

Statistics for all pairs and significant pairs

Examples

```
load(system.file("extradata","mee.example.rda",package = "ELMER"))
nearGenes <-GetNearGenes(TRange=getProbeInfo(mee,probe=c("cg00329272","cg10097755")),
                         geneAnnot=getGeneInfo(mee))
                       Hypo.pair <-get.pair(mee=mee,probes=c("cg00329272","cg10097755"),
                                            nearGenes=nearGenes,permu.size=5,Pe = 0.2,
                                            dir.out="./",
                                            label= "hypo")
```

get.permu *get.permu*

Description

get.permu

Usage

```
get.permu(mee, geneID, percentage = 0.2, rm.probes = NULL,
  permu.size = 1000, permu.dir = NULL, cores = NULL)
```

Arguments

mee	A MEE.data object containing at least meth, exp, probeInfo, geneInfo.
geneID	A vector lists the genes' ID.
percentage	A number ranges from 0 to 1 specifying the percentage of samples used to link probes to genes. Default is 0.2.
rm.probes	A vector lists the probes name.
permu.size	A number specify the times of permuation. Default is 1000.
permu.dir	A path where the output of permuation will be.
cores	A interger which defines number of core to be used in parallel process. Default is NULL: don't use parallel process.

Value

permutation

Examples

```
load(system.file("extradata","mee.example.rda",package = "ELMER"))
permu <-get.permu(mee=mee,geneID=rownames(getExp(mee)),
                  rm.probes=c("cg00329272","cg10097755"),
                  permu.size=5)
```

Get.Pvalue.p *Calculate empirical Pvalue*

Description

Calculate empirical Pvalue

Usage

```
Get.Pvalue.p(U.matrix, permu)
```

Arguments

U.matrix	A data.frame of raw pvalue from U test. Output from .Stat.nonpara
permu	data frame of permutation. Output from .Stat.nonpara.permu

Value

A data frame with empirical Pvalue.

---

get.TFs *get.TFs*

Description

get.TFs

Usage

```
get.TFs(mee, enriched.motif, TFs, motif.relavent.TFs, percentage = 0.2,
  dir.out = "./", label = NULL, cores = NULL)
```

Arguments

mee	A MEE.data object containing at least meth, exp, probeInfo, geneInfo.
enriched.motif	Can be either a list containing output of get.enriched.motif function or path of rda file containing output of get.enriched.motif function.
TFs	Can be either a data.frame containing TF GeneID and Symbol or path of csv file containing TF GeneID and Symbol. If missing, human TF list will be used. For detail information, refer reference paper.
motif.relavent.TFs	Can be either a list containing motif (list name) and relavent TF (content of list) or path of rda file containing a list containing motif (list name) and relavent TF (content of list). If missing, human TF list will be used. For detail information, refer reference paper.
percentage	A number ranges from 0 to 1 specifying the percentage of samples used to link probes to genes. Default is 0.2.
dir.out	A path specify the directory for outputs. Default is current directory
label	A character labels the outputs.
cores	A interger which defines number of core to be used in parallel process. Default is NULL: don't use parallel process.

Value

potential responsible TFs will be reported.

Examples

```
load(system.file("extradata","mee.example.rda",package = "ELMER"))
enriched.motif <- list("TP53"= c("cg00329272", "cg10097755", "cg08928189",
                                 "cg17153775", "cg21156590", "cg19749688", "cg12590404",
                                 "cg24517858", "cg00329272", "cg09010107", "cg15386853",
                                 "cg10097755", "cg09247779", "cg09181054"))
TF <- get.TFs(mee, enriched.motif,
TFs=data.frame(GeneID=c("ID7157","ID8626","ID7161"),
               Symbol=c("TP53","TP63","TP73"),
               stringsAsFactors = FALSE),
               label="hypo")
```

`get450K` *get450K*

Description

get450K

Usage

```
get450K(disease, basedir = "./Data")
```

Arguments

`disease`	A character to specify disease in TCGA such as BLCA
`basedir`	A path shows where the data will be stored.

`getClinic` *getClinic*

Description

getClinic

Usage

```
getClinic(disease, basedir = "./Data")
```

Arguments

`disease`	A character to specify disease in TCGA such as BLCA
`basedir`	A path shows where the data will be stored.

getExp *getExp*

Description

getExp

Usage

```
getExp(object, geneID, ID)

## S4 method for signature 'MEE.data'
getExp(object, geneID, ID)
```

Arguments

object	MEE.data object
geneID	A vector of genes' id. When specified, gene expression only for these genes will be output.
ID	A vector of sample ID. When specified, gene expression only for these samples will be output.

Examples

```
exp <- matrix(data=c(101:110),ncol=5,dimnames=list(c("gene1","gene2"),paste0("sample",1:5)))
mee <- fetch.mee(exp=exp)
Exp <- getExp(mee, geneID = "gene1") ## get gene expression for certain genes
Exp <- getExp(mee, ID = c("sample1","sample5")) ## get gene expression for certain samples
```

getGeneID *getGeneID*

Description

getGeneID

Usage

```
getGeneID(mee, symbol)
```

Arguments

mee	A MEE.data or Pair object.
symbol	A character which is the geneID

Value

gene symbol

Examples

```
geneInfo <- system.file("extdata","UCSC_gene_hg19.rda",package = "ELMER")
## input can be a path
pair <- fetch.pair(geneInfo=geneInfo)
getGeneID(pair, symbol="KIAA1804")
```

`getGeneInfo` *getProbeInfo*

Description

getProbeInfo

Usage

```
getGeneInfo(object, geneID, symbol, range)

## S4 method for signature 'ANY'
getGeneInfo(object, geneID, symbol, range)
```

Arguments

`object`	MEE.data or Pair object
`geneID`	A vector of genes' id. When specified, only the these genes' coordinate will be output.
`symbol`	A vector of genes' symbols . When specified, only the these genes' coordinate will be output.
`range`	A GRanges object. When specified, only the geneInfo locating within these loci will be output.

Examples

```
geneInfo <- system.file("extdata","UCSC_gene_hg19.rda",package = "ELMER")
mee <- fetch.mee(geneInfo=geneInfo)
Genes <- getGeneInfo(mee, geneID = "55811")
Genes <- getGeneInfo(mee, symbol ="ADCY10")
Genes <- getGeneInfo(mee, range = GRanges(seqnames="chr1", ranges=IRanges(1000000,1600000)))
```

`getMeth` *getMeth*

Description

getMeth

Usage

```
getMeth(object, probe, ID)

## S4 method for signature 'MEE.data'
getMeth(object, probe, ID)
```

Arguments

object	MEE.data object
probe	A vector of probes' name. When specified, DNA methylation only for these probes will be output.
ID	A vector of sample ID. When specified, DNA methylation only for these samples will be output.

Examples

```
meth <- matrix(data=c(1:20),ncol=5,dimnames=list(paste0("probe",1:4),paste0("sample",1:5)))
mee <- fetch.mee(meth=meth)
Meth <- getMeth(mee,probe = "probe1")
Meth <- getMeth(mee, ID = c("sample1","sample2"))
```

GetNearGenes *Collect nearby gene for one locus.*

Description

Collect nearby gene for one locus.

Usage

```
GetNearGenes(geneNum = 20, geneAnnot = NULL, TRange = NULL,
  cores = NULL)
```

Arguments

geneNum	A number determine how many gene will be collected from each side of target (number shoule be even) Default to 20.
geneAnnot	A GRange object contains coordinates of promoters for human genome.
TRange	A GRange object contains coordinate of a list targets.
cores	A number to specific how many cores to use to compute. Default to detectCores()/2.

Value

A data frame of nearby genes and information: genes' IDs, genes' symbols, distance with target and side to which the gene locate to the target.

Examples

```
load(system.file("extdata","UCSC_gene_hg19.rda",package = "ELMER"))
probe <- GRanges(seqnames = c("chr1","chr2"),
range=IRanges(start = c(16058489,236417627), end= c(16058489,236417627)),
name= c("cg18108049","cg17125141"))
NearbyGenes <- GetNearGenes(geneNum=20,geneAnnot=txs,TRange=probe)
```

getPair *getPair*

Description

getPair

Usage

```
getPair(object, probe, geneID)

## S4 method for signature 'Pair'
getPair(object, probe, geneID)
```

Arguments

object	Pair object
probe	A vector of probes' name. When specified, only the pair containing these probes will be output.
geneID	A vector of genes' id. When specified, only the pair containing these genes will be output.

Examples

```
df <- data.frame(Probe=c("cg19403323","cg12213388","cg26607897"),
GeneID =c("ID255928","ID84451","ID55811"),
Symbol =c("SYT14","KIAA1804","ADCY10"),
Pe=c(0.003322259,0.003322259,0.003322259))
pair <- fetch.pair(pair = df)
Pairs <- getPair(pair, probe = "cg19403323") # get pair information for a probe
Pairs <- getPair(pair, geneID = "ID55811") # get pair information for a gene
```

getProbeInfo *getProbeInfo*

Description

getProbeInfo

Usage

```
getProbeInfo(object, chr, probe, range)

## S4 method for signature 'ANY'
getProbeInfo(object, chr, probe, range)
```

Arguments

object	MEE.data or Pair object
chr	A vector of chromosome such chr1, chr2. When specified, only the probeInfo locating on these chromosome will be output.
probe	A vector of probes' name. When specified, only the these probes' coordinate will be output.
range	A GRanges object. When specified, only the probeInfo locating within these loci will be output.

Examples

```
probeInfo <- GRanges(seqnames = c("chr1","chr1","chr3"),
ranges = IRanges(start = c(1,6,20),end = c(2,7,21)),
name=c("cg1","cg2","cg3"))
mee <- fetch.mee(probeInfo=probeInfo)
Probes <- getProbeInfo(mee,chr="chr1") # get probes which locate on the chr1
Probes <- getProbeInfo(mee, probe = "cg1") # get certain probes information
Probes <- getProbeInfo(mee, range = GRanges(seqnames="chr1", ranges=IRanges(5,20)))
```

---

getRNAseq *getRNAseq*

Description

getRNAseq

Usage

```
getRNAseq(disease, basedir = "./Data")
```

Arguments

disease	A character to specify disease in TCGA such as BLCA
basedir	A path shows where the data will be stored.

---

getSample *getSample*

Description

getSample

Usage

```
getSample(object, ID, cols)

## S4 method for signature 'MEE.data'
getSample(object, ID, cols)
```

Arguments

object	MEE.data object
ID	A vector of sample ID. When specified, sample informtion only for these samples will be output.
cols	A vector of columns names of Sample slots of MEE.data object.

Examples

```
SampleInfo <- data.frame(ID=paste0("sample",1:5),
TN=c("Tumor","Tumor","Normal","Normal","Tumor"))
mee <- fetch.mee(sample = SampleInfo)
Samples <- getSample(mee,ID = "sample2") ## get sample2's information
Samples <- getSample(mee, cols = "TN")  ## get 'TN' information for each samples
```

---

getSymbol *getSymbol*

Description

getSymbol

Usage

```
getSymbol(mee, geneID)
```

Arguments

mee	A MEE.data or Pair object.
geneID	A character which is the geneID

Value

gene symbol

Examples

```
geneInfo <- system.file("extdata","UCSC_gene_hg19.rda",package = "ELMER")
## input can be a path
pair <- fetch.pair(geneInfo=geneInfo)
getSymbol(pair, geneID="84451")
```

getTCGA *getTCGA*

Description

getTCGA

Usage

```
getTCGA(disease, Meth = TRUE, RNA = TRUE, Clinic = TRUE,
  basedir = "./Data", RNAtype = "gene", Methfilter = 0.2)
```

Arguments

disease	A character to specify disease in TCGA such as BLCA
Meth	A logic if TRUE HM450K DNA methylation data will download.
RNA	A logic if TRUE RNA-seq Hiseq-V2 from TCGA level 3 will be download.
Clinic	A logic if TRUE clinic data will be download for that disease.
basedir	A path shows where the data will be stored.
RNAtype	A charactor to specify whether use isoform level or gene level. When RNAtype=gene, gene level gene expression will be used. When isoform, then isoform data will be used.
Methfilter	A number. For each probe, the percentage of NA among the all the samples should smaller than Methfilter.

Examples

```
getTCGA("BRCA",Meth=FALSE, RNA=FALSE, Clinic=TRUE, basedir="~")
```

lm_eqn *lable linear regression formula*

Description

lable linear regression formula

Usage

```
lm_eqn(df, Dep, Exp)
```

Arguments

df	A data.frame object contains two variables: dependent variable (Dep) and explanation variable (Exp).
Dep	A character specify dependent variable. The first column will be dependent variable as default.
Exp	A character specify explanation variable. The second column will be explanation variable as default.

Value

a linear regression formula

`matrixClinic` *matrixClinic*

Description

matrixClinic

Usage

```
matrixClinic(disease, basedir = "./Data")
```

Arguments

`disease`	A character to specify disease in TCGA such as BLCA
`basedir`	A path shows where the data will be stored.

`matrixMeth` *matrixMeth*

Description

matrixMeth

Usage

```
matrixMeth(disease, basedir = "./Data", filter = 0.2)
```

Arguments

`disease`	A character to specify disease in TCGA such as BLCA
`basedir`	A path shows where the data will be stored.
`filter`	For each probe, the percentage of NA among the all the samples should smaller than filter.

matrixRNA *matrixRNA*

Description

matrixRNA

Usage

```
matrixRNA(disease, basedir = "./Data", type = "gene")
```

Arguments

disease	A character to specify disease in TCGA such as BLCA
basedir	A path shows where the data will be stored.
type	A charactor to specify whether use isoform level or gene level. When RNAtype=gene, gene level gene expression will be used. When isoform, then isoform data will be used.

MEE.data-class *MEE.data An S4 class that methylation, expression, sample information, probe information and gene information.*

Description

MEE.data An S4 class that methylation, expression, sample information, probe information and gene information.

Slots

`meth` A matrix of DNA methylation. Each row is one probe and each column is one sample

`exp` A matrix of expression. Each row is one gene and each column is one sample

`sample` A data.frame contains sample information

`probeInfo` A GRange object contains probe information

`geneInfo` A GRange object contains gene information

`motif.enrichment.plot` *motif.enrichment.plot*

Description

motif.enrichment.plot

Usage

```
motif.enrichment.plot(motif.enrichment, significant = NULL, dir.out = "./",
  save = TRUE, label = NULL)
```

Arguments

`motif.enrichment`	A data frame or file path of get.enriched.motif output motif.enrichment.csv file. See detial for the format of motif.enrichment if a data frame is specified.
`significant`	A list to select subset of motif. Default is NULL. See detail
`dir.out`	A path specify the directory to which the figures will be saved. Current directory is default.
`save`	A logic. If true (default), figure will be saved to dir.out.
`label`	A character labels the outputs figure.

Details

motif.enrichment If input data.frame object, it should contain "motif", "OR", "lowerOR", "upperOR" columns. motif specifies name of motif; OR specifies Odds Ratio, lowerOR specifies lower boundary of OR (95 upperOR specifies upper boundary of OR(95

Examples

```
motif.enrichment <- data.frame(motif=c("TP53","NR3C1","E2F1","EBF1","RFX5",
"ZNF143", "CTCF"),
OR=c(19.33,4.83,1, 4.18, 3.67,3.03,2.49),
lowerOR =c(10,3,1.09,1.9,1.5,1.5, 0.82),
upperOR =c(23,5,3,7,6,5,5),
stringsAsFactors=FALSE)
motif.enrichment.plot(motif.enrichment=motif.enrichment,
                    significant=list(OR=3),
                    label="hypo", save=FALSE)
```

NearGenes *NearGenes*

Description

NearGenes

Usage

```
NearGenes(Target = NULL, Gene = NULL, geneNum = 20, TRange = NULL)
```

Arguments

Target	A charactor which is name of TRange or one of rownames of TBed.
Gene	A GRange object contains coordinates of promoters for human genome.
geneNum	A number determine how many gene will be collected from each side of target (number shoule be even).
TRange	A GRange object contains coordinate of targets.

Value

A data frame of nearby genes and information: genes' IDs, genes' symbols, distance with target and side to which the gene locate to the target.

Normalize *Normalization to 0 to 1*

Description

Normalization to 0 to 1

Usage

```
Normalize(x, col = FALSE, row = FALSE, na.rm = FALSE)
```

Arguments

x	A matrix.
col	A boolean to determine normalize by column or not.
row	A boolean to determine normalize by row or not.
na.rm	A boolean to determine to remove na number or not.

Value

A normalized matrix.

NormalizeMean *Normalization based on mean*

Description

Normalization based on mean

Usage

```
NormalizeMean(x, col = FALSE, row = FALSE, na.rm = FALSE)
```

Arguments

x	A matrix.
col	A boolean to determine normalize by column or not.
row	A boolean to determine normalize by row or not.
na.rm	A boolean to determine to remove na number or not.

Value

A normalized matrix.

NormalizeMedian *Normalization based on median*

Description

Normalization based on median

Usage

```
NormalizeMedian(x, col = FALSE, row = FALSE, na.rm = FALSE)
```

Arguments

x	A matrix.
col	A boolean to determine normalize by column or not.
row	A boolean to determine normalize by row or not.
na.rm	A boolean to determine to remove na number or not.

Value

A normalized matrix.

Pair-class *An S4 class that pairs information, probe information and gene information.*

Description

An S4 class that pairs information, probe information and gene information.

Slots

`pairInfo` A data.frame

`probeInfo` A GRanges object.

`geneInfo` A GRanges object.

RandomLoci *Generate random loci of genome.*

Description

Generate random loci of genome.

Usage

```
RandomLoci(SampleSize = NULL, exclusion = NULL, regionWidth = 0)
```

Arguments

`SampleSize`	A number of random loci you want to generate.
`exclusion`	The chromosome you want to exclude such as chrX chrY.
`regionWidth`	The width of each random loci.

Value

GRange object.

ReadBed *Read a bed file.*

Description

Read a bed file.

Usage

```
ReadBed(x, strand = FALSE, skip = 0, cols = NULL, seqLength = NULL)
```

Arguments

x	A path of bed file (characters)
strand	A boolean to specific strands. If true, strand column will be filled as input. If false, strand column will be filled "*""
skip	A number to specify how many lines should be removed from bed file.
cols	Specify the column to read from bed file.
seqLength	Specify custmer seqLength parameter in GRange function

Value

GRange object containning bed file information.

Examples

```
file <- system.file("extdata","Union_strong_enhancer_REMC_FANTOM.bed.xz",
package = "ELMER")
Bed <- ReadBed(file)
```

scatter *scatter*

Description

scatter

Usage

```
scatter(meth, exp, category = NULL, xlab = NULL, ylab = NULL,
  title = NULL, color.value = NULL, lm_line = FALSE)
```

Arguments

meth	A vector of number.
exp	A vector of number or matrix with sample in column and gene in rows.
category	A vector of sample labels.
xlab	A character specify the title of x axis.
ylab	A character specify the title of y axis.
title	A character specify the figure title.
color.value	A vector specify the color of each category, such as color.value=c("Tumor"="red","Normal"="darkgre
lm_line	A logic. If it is TRUE, regression line will be added to the graph.

Value

ggplot figure object

`scatter.plot` *scatter.plot*

Description

scatter.plot

Usage

```
scatter.plot(mee, byPair = list(probe = c(), gene = c()),
  byProbe = list(probe = c(), geneNum = 20), byTF = list(TF = c(), probe =
  c()), category = NULL, dir.out = "./", save = TRUE, ...)
```

Arguments

mee	A MEE.data object includes DNA methylation data, expression data, probeInfo and geneInfo.
byPair	A list: byPair =list(probe=c(),gene=c()); probe contains a vector of probes' name and gene contains a vector of gene ID. The length of probe should be the same with length of gene. Output see detail.
byProbe	A list byProbe =list(probe=c(), geneNum=20); probe contains a vector of probes'name and geneNum specify the number of gene near the probes will ploted. 20 is default for geneNum. Output see detail.
byTF	A list byTF =list(TF=c(), probe=c()); TF contains a vector of TF's symbol and probe contains the a vector of probes' name. Output see detail.
category	A vector labels subtype of samples or a character which is the column name in the sampleInfo in the MEE.data object. Once specified, samples will label different color. The color can be customized by using color.value.
dir.out	A path specify the directory to which the figures will be saved. Current directory is default.
save	A logic. If true, figure will be saved to dir.out.
...	color.value, lm_line in scatter function

Details

byPair The output will be scatter plot for individual pairs.

Value

Scatter plots.

Examples

```
load(system.file("extradata","mee.example.rda",package = "ELMER"))
scatter.plot(mee,byProbe=list(probe=c("cg19403323"),geneNum=20),
           category="TN", save=FALSE)
scatter.plot(mee,byProbe=list(probe=c("cg19403323"),geneNum=20),
           category="TN", save=TRUE) ## save to pdf
# b. generate one probe-gene pair
scatter.plot(mee,byPair=list(probe=c("cg19403323"),gene=c("ID255928")),
            category="TN", save=FALSE,lm_line=TRUE)
```

`schematic` *schematicPlot*

Description

schematicPlot

Usage

```
schematic(probe.range, gene.range, special = list(names = c(), colors = c()),
  interaction = list(probe = c(), gene = c(), colors = c()), label,
  save = TRUE)
```

Arguments

`probe.range`	A GRanges object contains probes coordinates.
`gene.range`	A GRanges object contains gene TSS coordinates.
`special`	A list: special=list(names=c(), colors=c()) show the name of feature you want to highlight and specify the color respectively.
`interaction`	A list: interaction=list(probe=c(),gene=c(),colors=c()) show the interacted features and specify the color respectively.
`label`	A character labels the outputs figure.
`save`	A logic. If true, figure will be saved to dir.out.

Details

byProbes When a vector of probes' name are provided, function will produce schematic plot for each individual probes. The schematic plot contains probe, nearby 20 (or the number of gene user specified.) genes and the significantly linked gene with the probe.

schematic.plot *schematicPlot*

Description

schematicPlot

Usage

```
schematic.plot(pair, byProbe, byGene, byCoordinate = list(chr = c(), start =
  c(), end = c()), dir.out = "./", save = TRUE, ...)
```

Arguments

pair	A pair object. All slots should be included
byProbe	A vector of probe names.
byGene	A vector of gene ID
byCoordinate	A list contains chr, start and end. byCoordinate=list(chr=c(),start=c(),end=c()).
dir.out	A path specify the directory for outputs. Default is current directory
save	A logic. If true, figure will be saved to dir.out.
...	Parameters for GetNearGenes

Details

byProbes When a vector of probes' name are provided, function will produce schematic plot for each individual probes. The schematic plot contains probe, nearby 20 (or the number of gene user specified.) genes and the significantly linked gene with the probe.

Value

a schematic plot will be produced.

Examples

```
library(grid)
load(system.file("extradata","mee.example.rda",package = "ELMER"))
nearGenes <-GetNearGenes(TRange=getProbeInfo(mee,probe=c("cg00329272","cg19403323")),
                         geneAnnot=getGeneInfo(mee))
Hypo.pair <-get.pair(mee=mee,probes=c("cg00329272","cg19403323"),
                     nearGenes=nearGenes,permu.size=5,Pe = 0.2,dir.out="./",
                     label= "hypo")
pair <- fetch.pair(pair=Hypo.pair,
                   probeInfo = getProbeInfo(mee),
                   geneInfo = getGeneInfo(mee))
# a. generate schematic plot of one probe with nearby 20 genes and label
#the gene significantly linked with the probe.
grid.newpage()
schematic.plot(pair=pair, byProbe="cg19403323" ,save=FALSE)
#b. generate schematic plot of ont gene with the probe which the gene significanlty linked to.
grid.newpage()
schematic.plot(pair=pair, byGene="ID255928",save=FALSE)
```

splitmatrix *splitmatix*

Description

splitmatix

Usage

```
splitmatrix(x, by = "row")
```

Arguments

x	A matrix
by	A character specify if split the matix by row or column.

standardizeTcgaId *tcgaSampleType.*

Description

tcgaSampleType.

Usage

```
standardizeTcgaId(tcgaId)
```

Arguments

tcgaId	A vector list TCGA sample barcode.

Value

standardized TCGA ID

`Stat.diff.meth` *Stat.diff.meth*

Description

Stat.diff.meth

Usage

```
Stat.diff.meth(probe, meths, TN, test = t.test, percentage = 0.2,
  Top.m = NULL)
```

Arguments

probe	A charactor specify probe name
meths	A matrix contain DNA methylation data.
TN	A vector of category of samples.
test	A function specify which statistic test will be used.
percentage	A number specify the percentage of normal and tumor samples used in the test.
Top.m	A logic. If to identify hypomethylated probe Top.m should be FALSE. hypermethylated probe is TRUE.

`Stat.nonpara` *U test (non parameter test) for permutation. This is one probe vs nearby gene which is good for computing each probes for nearby genes.*

Description

U test (non parameter test) for permutation. This is one probe vs nearby gene which is good for computing each probes for nearby genes.

Usage

```
Stat.nonpara(Probe, NearGenes, K, Top = NULL, Meths = Meths, Exps = Exps)
```

Arguments

Probe	A character of name of Probe in array.
NearGenes	A list of nearby gene for each probe which is output of GetNearGenes function.
K	A number determines the methylated groups and unmethylated groups.
Top	A number determines the percentage of top methylated/unmethylated samples.
Meths	A matrix contains methylation for each probe (row) and each sample (column).
Exps	A matrix contains Expression for each gene (row) and each sample (column).

Value

U test results

Stat.nonpara.permu *Stat.nonpara.permu*

Description

Stat.nonpara.permu

Usage

```
Stat.nonpara.permu(Probe, Gene, Top = 0.2, Meths = Meths, Exps = Exps,
  permu.dir = NULL)
```

Arguments

Probe	A character of name of Probe in array.
Gene	A vector of gene ID.
Top	A number determines the percentage of top methylated/unmethylated samples.
Meths	A matrix contains methylation for each probe (row) and each sample (column).
Exps	A matrix contains Expression for each gene (row) and each sample (column).
permu.dir	A path to store permuation data.

Value

U test results

TCGA.pipe *ELMER analysis pipe for TCGA data.*

Description

ELMER analysis pipe for TCGA data.

Usage

```
TCGA.pipe(disease, analysis = "all", wd = "./", cores = NULL,
  Data = NULL, ...)
```

Arguments

disease	TCGA short form disease name such as COAD
analysis	a vector of characters listing the analysis need to be done. Analysis are "download","distal.enhancer","diffMeth","pair","motif","TF.search". Default is "all" meaning all the analysis will be processed.
wd	a path showing working dirctory. Default is "./"
cores	A interger which defines number of core to be used in parallel process. Default is NULL: don't use parallel process.
Data	A path showing the folder containing DNA methylation, expression and clinic data
...	A list of parameters for functions: GetNearGenes, get.feature.probe, get.diff.meth, get.pair,

Value

Different analysis results.

Examples

```
distal.probe <- TCGA.pipe(disease = "LUSC", analysis="distal.enhancer", wd="~/")
```

`tcgaSampleType` *tcgaSampleType.*

Description

tcgaSampleType.

Usage

```
tcgaSampleType(x)
```

Arguments

`x`	A TCGA sample barcode.

Value

Tissue type: Tumor, Normal, Control, TRB, cell_line, XP, XCL

`TF.rank.plot` *TF.rank.plot*

Description

TF.rank.plot

Usage

```
TF.rank.plot(motif.pvalue, motif, TF.label, dir.out = "./", save = TRUE)
```

Arguments

`motif.pvalue`	A matrix or a path specifying location of "XXX.with.pvalue.rda" which is output of getTF.
`motif`	A vector of charactor specify the motif to plot
`TF.label`	A list show the label for each motif. If TF.label is not specified, the motif relevant TF and top3 TF will be labeled.
`dir.out`	A path specify the directory to which the figures will be saved. Current directory is default.
`save`	A logic. If true (default), figure will be saved to dir.out.

Examples

```
load(system.file("extradata","getTF.hypo.TFs.with.motif.pvalue.rda",package="ELMER"))
TF.rank.plot(motif.pvalue=TF.meth.cor, motif="TP53", TF.label=list(TP53=c("TP53","TP63","TP73")),
            save=FALSE)
```

`WriteBed` *Write a bed file from GRange object.*

Description

Write a bed file from GRange object.

Usage

```
WriteBed(x, save = TRUE, fn = NULL)
```

Arguments

`x`	GRange object
`save`	if save is false, function will return a bed format data.frame. If save is true, fn parameter need to be specific and it output bed file in the path you specified in fn.
`fn`	A name of bed file you want to output.

Value

A data.frame bed object or save output bed file.

Examples

```
probeInfo <- GRanges(seqnames = c("chr1","chr1","chr3"),
ranges = IRanges(start = c(1,6,20),end = c(2,7,21)),
name=c("cg1","cg2","cg3"))
WriteBed(probeInfo, fn="test.bed")
```

Index