Package ‘chicane’

October 1, 2019

Title  Capture Hi-C Analysis Engine
Type   Package
Version 0.1.1
Date  2019-09-29
Author  Erle Holgersen [aut, cre],
Olivia Leavy [aut],
Olivia Fletcher [aut],
Frank Dudbridge [aut],
Syed Haider [aut]
Maintainer  Erle Holgersen <erle.holgersen@gmail.com>
Description  Toolkit for processing and calling interactions in capture Hi-C data. Converts BAM files into counts of reads linking restriction fragments, and identifies pairs of fragments that interact more than expected by chance. Significant interactions are identified by comparing the observed read count to the expected background rate from a count regression model.
Depends  R (>= 3.5.0), gamlss.tr, gamlss
License  GPL-2
Encoding  UTF-8
LazyData  true
Imports  data.table, MASS, stats, foreach, doParallel, iterators
SystemRequirements  bedtools
Suggests  knitr, rmarkdown, testthat, GenomicInteractions, Gviz, countreg
Additional_repositories  http://R-Forge.R-project.org/
VignetteBuilder  knitr
RoxygenNote  6.1.1
NeedsCompilation  no
Repository  CRAN
Date/Publication  2019-10-01 19:20:02 UTC
R topics documented:

- add.covariates
- add.fragment.coordinates
- bedtools.installed
- bre80
- check.model.numerical.fit
- check.split.data.numerical.fit
- chicane
- combine.replicates
- convert.bam
- convert.hicup.digest.bed
- distance.bin
- distance.split
- fill.in.zeros
- filter.fragments
- fit.glm
- fit.model
- get.combination.count
- get.distance
- get.id.components
- get.trans.counts
- is.glm.nb.maxiter.warning
- is.glm.nb.theta.error
- is.glm.nb.theta.warning
- make.modelfit.plot
- model.rows.sanity.check
- model.try.catch
- multiple.testing.correct
- prepare.data
- read.bed
- run.model.fitting
- smart.split
- stratified.enrichment.sample
- test.enrichment
- verify.interaction.data

Index

add.covariates  add.covariates

Description

Add model covariates (trans counts and distance) to an interactions data table.

Usage

add.covariates(interaction.data)
add.fragment.coordinates

Arguments

interaction.data
data.table with interaction data. Must contain columns bait.id, target.id, bait.chr, bait.start, bait.end, target.chr, target.start, target.end and count.

Value

Updated data table with new columns

bait.trans.count
number of trans interactions of bait fragment
target.trans.count
number of trans interactions of target fragment
distance
distance between bait and target fragment, or NA if trans

Author(s)

Erle Holgersen <Erle.Holgersen@icr.ac.uk>

Examples

data(bre80);
input.cols <- c('bait.id', 'target.id', 'bait.chr', 'bait.start', 'bait.end', 'target.chr', 'target.start', 'target.end', 'count');
output <- add.covariates(bre80[, input.cols, with = FALSE]);

Description

Expand target and bait IDs of the form chrN:start-end to separate coordinate columns in the data table

Usage

add.fragment.coordinates(id.data)

Arguments

id.data
data table containing columns target.id and/or bait.id to be expanded

Value

Data table with added coordinate columns for target and bait (as applicable).
Author(s)
Erle Holgersen <Erle.Holgersen@icr.ac.uk>

Examples
```
data(bre80);
add.fragment.coordinates(bre80[, .(bait.id, target.id)]);
```

Description
Check if bedtools exists in PATH

Usage
```
bedtools.installed()
```

Value
Logical indicating if bedtools was found in PATH

Author(s)
Erle Holgersen <Erle.Holgersen@icr.ac.uk>

Examples
```
bedtools.installed();
```

Description
A dataset containing processed data from a capture Hi-C experiment in the Bre80 normal epithelial breast tissue cell line. The experiment targeted several breast cancer risk loci, and reads that mapped to the 2q35 SNPs rs13387042 and rs16857609 are included in the dataset.

Data was prepared using the prepare.data function. Coordinates are GRCh38.

Usage
```
data(bre80)
```
**check.model.numerical.fit**

**Format**

A data table object with 47,766 rows and 13 columns.

The variables are as follows:

- target.id String in chrN:start-end format identifying target fragment
- bait.id String in chrN:start-end format identifying bait fragment
- target.chr Chromosome of target fragment
- target.start Start coordinate of target fragment (zero-based)
- target.end End coordinate of target fragment
- bait.chr Chromosome of bait fragment
- bait.start Start coordinate of bait fragment (zero-based)
- bait.end End coordinate of bait fragment
- bait.to.bait Boolean indicating if the interaction is bait-to-bait (i.e. the fragment listed as target is also a bait)
- bait.trans.count The number of reads linking the bait to fragments in trans (a measure of "interactibility")
- target.trans.count The number of reads linking the target to fragments in trans (a measure of "interactibility")
- distance Distance between the midpoints of the bait and target fragments (basepairs). NA for trans interactions
- count The number of reads linking the two fragments

**References**


**Description**

Check if chicane model can be fit on a given dataset. **glm.nb** does not work when all responses are constant, or there are only two unique values and a covariate is a perfect predictor.

**Usage**

```r
check.model.numerical.fit(interaction.data)
```

**Arguments**

- `interaction.data`  
  Data table of interaction data on which model is to be fit
Value

boolean indicating if model can be fit

---

check.split.data.numerical.fit

Description

Helper function to check if the chicane model can be fit on each element of a split data list.

Usage

check.split.data.numerical.fit(split.data)

Arguments

split.data List of data.table objects with fragment interaction data

Value

Logical indicating if the model can be fit

---

chicane

Description

Run full method for detecting significant interactions in capture Hi-C experiments, starting either from a BAM file or preprocessed data from prepare.data

Usage

chicane(bam = NULL, baits = NULL, fragments = NULL, interactions = NULL, replicate.merging.method = "sum", distribution = "negative-binomial", include.zeros = "none", bait.filters = c(0, 1), target.filters = c(0, 1), distance.bins = NULL, multiple.testing.correction = c("bait-level", "global"), adjustment.terms = NULL, remove.adjacent = FALSE, temp.directory = NULL, keep.files = FALSE, maxit = 100, epsilon = 1e-08, cores = 1, trace = FALSE, verbose = FALSE, interim.data.dir = NULL)
Arguments

- **bam**: Path to a BAM file
- **baits**: Path to a BED file containing the baits
- **fragments**: Path to a BED file containing all restriction fragments in the genome
- **interactions**: Data table or path to a text file detailing fragment interactions, typically from `prepare.data`. Can be used instead of bam/baits/fragments specification if the text files have already been prepared.
- **replicate.merging.method**: Method that should be used for merging replicates, if applicable
- **distribution**: Name of distribution of the counts. Options are 'negative-binomial', 'poisson', 'truncated-poisson', and 'truncated-negative-binomial'
- **include.zeros**: String specifying what zero counts to include. Options are none (default), cis, and all.
- **bait.filters**: Vector of length two, where the first element corresponds to the lower-end filter and the second to the upper-end filter. When global multiple testing correction is performed, altering the bait filtering settings may affect the number of significant results.
- **target.filters**: Vector of length two, giving lower and higher filter, respectively. Changing this filtering setting may affect multiple testing correction by altering the number of tests performed.
- **distance.bins**: Number of bins to split distance into. Models are fit separately in each bin.
- **multiple.testing.correction**: String specifying how multiple testing correction should be performed, by bait or globally.
- **adjustment.terms**: Character vector of extra terms to adjust for in the model fit.
- **remove.adjacent**: Logical indicating whether to remove all reads mapping to adjacent restriction fragments.
- **temp.directory**: Directory where temporary files should be stored. Defaults to current directory.
- **keep.files**: Logical indicating whether to keep temporary files.
- **maxit**: Maximum number of IWLS iterations for fitting the model (passed to `glm.control`)
- **epsilon**: Positive convergence tolerance for Poisson and negative binomial models. Passed to `glm.control`
- **cores**: Integer value specifying how many cores to use to fit model for cis-interactions.
- **trace**: Logical indicating if output should be produced for each of model fitting procedure. Passed to `glm.control` or `gamlss.control`
- **verbose**: Logical indicating whether to print progress reports.
- **interim.data.dir**: Path to directory to store intermediate QC data and plots. NULL indicate skip intermediate results.
### Value

Data table with columns

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target.id</td>
<td>String in chrN:start-end format identifying target fragment</td>
</tr>
<tr>
<td>bait.id</td>
<td>String in chrN:start-end format identifying bait fragment</td>
</tr>
<tr>
<td>target.chr</td>
<td>Chromosome of target fragment</td>
</tr>
<tr>
<td>target.start</td>
<td>Start coordinate of target fragment (zero-based)</td>
</tr>
<tr>
<td>target.end</td>
<td>End coordinate of target fragment</td>
</tr>
<tr>
<td>bait.chr</td>
<td>Chromosome of bait fragment</td>
</tr>
<tr>
<td>bait.start</td>
<td>Start coordinate of bait fragment (zero-based)</td>
</tr>
<tr>
<td>bait.end</td>
<td>End coordinate of bait fragment</td>
</tr>
<tr>
<td>bait.to.bait</td>
<td>Boolean indicating if the interaction is bait-to-bait (i.e. the fragment listed as target is also a bait)</td>
</tr>
<tr>
<td>bait.trans.count</td>
<td>The number of reads linking the bait to fragments in trans (a measure of &quot;interactibility&quot;)</td>
</tr>
<tr>
<td>target.trans.count</td>
<td>The number of reads linking the target to fragments in trans (a measure of &quot;interactibility&quot;)</td>
</tr>
<tr>
<td>distance</td>
<td>Distance between the midpoints of the bait and target fragments (basepairs). NA for trans interactions</td>
</tr>
<tr>
<td>count</td>
<td>The number of reads linking the two fragments</td>
</tr>
<tr>
<td>expected</td>
<td>The expected number of reads linking the two fragments under the fitted model</td>
</tr>
<tr>
<td>p.value</td>
<td>P-value for test of the observed number of reads significantly exceeding the expected count</td>
</tr>
<tr>
<td>q.value</td>
<td>FDR-corrected p-value</td>
</tr>
</tbody>
</table>

### Author(s)

Erle Holgersen <Erle.Holgersen@icr.ac.uk>

### Examples

```r
# start from BAM file

bam <- system.file('extdata', 'Bre80_2q35.bam', package = 'chicane');
baits <- system.file('extdata', '2q35.bed', package = 'chicane');
fragments <- system.file('extdata', 'GRCh38_HindIII_chr2.bed.gz', package = 'chicane');
results <- chicane(  
  bam = bam,  
  baits = baits,  
  fragments = fragments  
);
```
# Combine biological replicates

## Description

Merge biological replicates.

## Usage

`combine.replicates(replicates, method = c("sum", "weighted-sum"))`

## Arguments

- `replicates`: list of data table objects from `prepare.data`
- `method`: string specifying the method for merging replicates. Options are 'sum' and 'weighted-sum'.

## Details

The parameter `method` determines which method is used for merging replicates. Available options are weighted-sum and sum.

'weighted-sum' implements the size factor scaling approach used in DEseq, rounded to the closest integer. See Anders and Huber 2010 for details.

'sum' is the naive sum of counts across biological replicates.

## Value

Data table object containing merged data, where counts are stored in columns

- `count.i`: count of interaction in ith replicate
- `count`: count after merging replicates

## References

Examples

# preprocess data
bam <- system.file('extdata', 'Bre80_2q35.bam', package = 'chicane');
baits <- system.file('extdata', '2q35.bed', package = 'chicane');
fragments <- system.file('extdata', 'GRCh38_HindIII_chr2.bed.gz', package = 'chicane');
input.data <- prepare.data(
  bam = bam,
  baits = baits,
  fragments = fragments
);

# combined two datasets into one
merged <- combine.replicates(list(input.data, input.data));

convert.bam

Description

Convert a BAM file to a format that can be used for replicate merging.

Note: This function does not process data enough to be used for interaction calling. Use prepare.data for full preprocessing.

Usage

convert.bam(bam, baits, fragments, temp.directory = NULL,
  keep.files = FALSE)

Arguments

- **bam**: Path to a BAM file
- **baits**: Path to a BED file containing the baits
- **fragments**: Path to a BED file containing all restriction fragments in the genome
- **temp.directory**: Directory where temporary files should be stored. Defaults to current directory.
- **keep.files**: Logical indicating whether to keep temporary files

Author(s)

Erle Holgersen <Erle.Holgersen@icr.ac.uk>

See Also

prepare.data
**convert.hicup.digest.bed**

**Description**
Convert a HiCUP digest file to BED format.

**Usage**

```r
convert.hicup.digest.bed(hicup.digest, file.name = \"")
```

**Arguments**
- `hicup.digest`: Path to HiCUP digest
- `file.name`: Path to output file. A blank string indicates output to the console.

**Examples**

```r
hicup.digest <- system.file('extdata', 'HiCUP_digest_example.txt', package = 'chicane');
convert.hicup.digest.bed(hicup.digest);
```

---

**distance.bin**

**Description**
Assign distances to a meaningful category.

**Usage**

```r
distance.bin(distance)
```

**Arguments**
- `distance`: Vector of distances that should be mapped to a distance bin

**Value**
- vector of same length as distance containing assigned distance bins
distance.split

distance.split

distance.split

Description

Split interaction data into subsets that are large enough for the chicane model to be fit (see Details), based on distance. This step allows the distance term in the model to be fit in a piecewise linear fashion.

Usage

distance.split(interaction.data, distance.bins = NULL, min.rows.bin = 50, verbose = FALSE)

Arguments

interaction.data
  Data table of interaction data, typically from prepare.data

distance.bins
  Number of distance bins desired. If NULL, a number is chosen to ensure that the negative binomial can be fit in all bins.

min.rows.bin
  The minimum number of expected rows in a distance bin. Ignored if distance.bins is set

verbose
  Logical indicating whether to print progress reports

Details

Fitting glm.nb fails when there is a lack of overdispersion in the data. The chicane method contains logic to catch these errors and instead fit a Poisson model. However, to avoid this happening more than necessary, an attempt is made to avoid distance splits that will clearly result in numerical errors. This includes bins of data where the count is the same for all rows, or a covariate is a perfect predictor of count.

Value

List where each element corresponds to a specified distance bin, and the final one corresponding to trans-interactions (if present)

Examples

data(bre80);
distance.split(bre80);
fill.in.zeros

Description
Add zero counts to interaction data

Usage
fill.in.zeros(interaction.data, baits, fragments)
fill.in.zeroses(interaction.data, baits, fragments)

Arguments
interaction.data
   Data table containing interaction data
baits
   Vector of bait IDs used in the experiment, in format chrN:start-end
fragments
   Vector of potential fragments the baits can link up to, in format chrN:start-end

Value
Data table containing origiina

Examples
data(bre80);
bait.file <- system.file('extdata', '2q35.bed', package = 'chicane');
fragment.file <- system.file('extdata', 'GRCh38_HindIII_chr2.bed.gz', package = 'chicane');
results <- fill.in.zeros(bre80,
   baits = read.bed(bait.file),
   fragments = read.bed(fragment.file));

filter.fragments

Description
Filter low and high-interacting restriction fragments based on the total number of trans counts

Usage
filter.fragments(interaction.data, bait.filters = c(0, 1),
   target.filters = c(0, 1), verbose = FALSE)
Arguments

interaction.data  Data table containing interactions
bait.filters  Vector of length two, where the first element corresponds to the lower-end filter and the second to the upper-end filter. When global multiple testing correction is performed, altering the bait filtering settings may affect the number of significant results.
target.filters  Vector of length two, giving lower and higher filter, respectively. Changing this filtering setting may affect multiple testing correction by altering the number of tests performed.
verbose  Logical indicating whether to print progress reports.

Value

Data table containing fragments that passed all filters

Author(s)

Erle Holgersen <Erle.Holgersen@icr.ac.uk>

Examples

# filter out lowest 10% of baits
filter.fragments(bre80, bait.filters = c(0.1, 1))

Description

Fit GLM according to a specified distribution. This needs to be done separately from glm in order to include negative binomial and truncated distributions as options.

Usage

fit.glm(formula, data, distribution = c("negative-binomial", "poisson", "truncated-poisson", "truncated-negative-binomial"), start = NULL, init.theta = NULL, maxit = 100, epsilon = 1e-08, trace = FALSE)

Arguments

formula  Formula specifying model of interest
data  Data frame containing variables specified in formula
distribution  Name of distribution of the counts. Options are 'negative-binomial', 'poisson', 'truncated-poisson', and 'truncated-negative-binomial'
**start**  
Starting values for model coefficients

**init.theta**  
Initial value of theta if fitting the negative binomial distribution

**maxit**  
Maximum number of IWLS iterations for fitting the model (passed to glm.control)

**epsilon**  
Positive convergence tolerance for Poisson and negative binomial models. Passed to glm.control

**trace**  
Logical indicating if output should be produced for each of model fitting procedure. Passed to glm.control or gamlss.control

**Value**

List with elements

- **model**  
  model object

- **expected.values**  
  vector of expected values for each element in original data

- **p.values**  
  vector of p-values for test of significantly higher response than expected

---

**Description**

Fit negative binomial model to obtain p-values for interactions.

**Usage**

```r
fit.model(interaction.data, distance.bins = NULL,  
distribution = "negative-binomial", bait.filters = c(0, 1),  
target.filters = c(0, 1), adjustment.terms = NULL, maxit = 100,  
epsilon = 1e-08, cores = 1, trace = FALSE, verbose = FALSE,  
interim.data.dir = NULL)
```

**Arguments**

- **interaction.data**  
  data.table object containing interaction counts. Must contain columns distance, count, and bait_trans_count.

- **distance.bins**  
  Number of bins to split distance into. Models are fit separately in each bin.

- **distribution**  
  Name of distribution of the counts. Options are 'negative-binomial', 'poisson', 'truncated-poisson', and 'truncated-negative-binomial'

- **bait.filters**  
  Vector of length two, where the first element corresponds to the lower-end filter and the second to the upper-end filter. When global multiple testing correction is performed, altering the bait filtering settings may affect the number of significant results.
target.filters Vector of length two, giving lower and higher filter, respectively. Changing this filtering setting may affect multiple testing correction by altering the number of tests performed.

adjustment.terms Character vector of extra terms to adjust for in the model fit.

maxit Maximum number of IWLS iterations for fitting the model (passed to glm.control)

epsilon Positive convergence tolerance for Poisson and negative binomial models. Passed to glm.control

cores Integer value specifying how many cores to use to fit model for cis-interactions.

trace Logical indicating if output should be produced for each of model fitting procedure. Passed to glm.control or gamlss.control

verbose Logical indicating whether to print progress reports.

interim.data.dir Path to directory to store intermediate QC data and plots.

Details

Fit a negative binomial model for obtaining p-value for interactions. The data is first sorted by distance, and models are fit separately in each quantile of the distance-sorted data.

Value

Interactions data with expected number of interactions and p-values added.

Examples

data(bre80);
fit.model(bre80);

get.combination.count (baits, fragments, cis.only = FALSE)
get.distance

Arguments

- **baits**: vector of bait IDs in form chrN:start-end
- **fragments**: vector of fragment IDs in form chrN:start-end
- **cis.only**: logical indicating whether cis-interactions only should be considered

Value

total number of possible combinations

Description

Calculate distance between bait and target region

Usage

```r
get.distance(interaction.data)
```

Arguments

- **interaction.data**: data.table with interaction data. Must contain columns bait.chr, bait.start, bait.end, target.chr, target.start, target.end

Value

vector of absolute distances (NA for trans-interactions)

Examples

```r
data(bre80);
input.cols <- c('bait.chr', 'bait.start', 'bait.end', 'target.chr', 'target.start', 'target.end');
get.distance(bre80[, input.cols, with = FALSE]);
```
get.id.components

Description
Split a segment ID in form \texttt{chrN:start-end} into its different components

Usage
get.id.components(id)

Arguments
id 

Value
A character vector of length three, where the elements are chromosome, start, and end, respectively. If \texttt{id} is a vector, a list of the same length is returned

Examples
get.id.components('chrX:6-30');
get.id.components(c('3:4-10', '22:1000-20000'))

get.trans.counts

Description
Calculate the number of trans-interactions per fragment, accounting for the fact that baits can be listed either as bait or target.

Usage
get.trans.counts(interaction.data)

Arguments
interaction.data 
Data table containing interactions
Value

Data table with columns `fragment.id` and `trans.count`.

- `fragment.id` : ID of restriction fragment in chrN:start-end format
- `trans.count` : Number of trans interactions involving the fragment

Examples

data(bre80);
get.trans.counts(bre80[, .(bait.chr, target.chr, bait.id, target.id, count)]);

Description

Check if a warning object is an iteration limit reached warning from `glm.nb`

Usage

`is.glm.nb.maxiter.warning(w)`

Arguments

- `w` : Warning object

Value

Logical indicating if warning matches iteration limit reached warning

Description

Check if an error matches the error raised by `glm.nb` due to an inflated theta estimate. This happens when the variance of the negative binomial does not exceed the mean (i.e. there is no overdispersion). In such cases, the Poisson distribution may be a suitable alternative.

Usage

`is.glm.nb.theta.error(e)`
Arguments
e Error object

Value
Boolean indicating if error matches

is.glm.nb.theta.warning

is.glm.nb.theta.warning

Description
Check if a warning matches the square root warning raised by glm.nb due to an inflated theta estimate. This happens when the variance of the negative binomial does not exceed the mean (i.e. there is no overdispersion). In such cases, the Poisson distribution may be a suitable alternative.

Usage
is.glm.nb.theta.warning(w)

Arguments
w Warning object

Value
Boolean indicating if warning matches

make.modelfit.plot

make.modelfit.plot

Description
create a plot representing model’s fit

Usage
make.modelfit.plot(model, file.name = NULL, resolution = 600)

Arguments
model An object of fitted model
file.name A string specifying plotting file name
resolution A numeric specifying plot’s resolution
**model.rows.sanity.check**

**Value**

TRUE if plot was successfully created

**Author(s)**

Syed Haider <Syed.Haider@icr.ac.uk>

---

**model.rows.sanity.check**

**Value**

None

---

**model.try.catch**

**Description**

Internal function for fitting model within a tryCatch loop, handling numerical errors gracefully.

**Usage**

model.try.catch(model.formula, data, distribution = "negative-binomial", maxit = 100, epsilon = 1e-08, init.theta = NULL, start = NULL, trace = FALSE, verbose = FALSE)
Arguments

- **model.formula**: formula
- **data**: model data
- **distribution**: Name of distribution of the counts. Options are 'negative-binomial', 'poisson', 'truncated-poisson', and 'truncated-negative-binomial'
- **maxit**: Maximum number of IWLS iterations for fitting the model (passed to glm.control)
- **epsilon**: Positive convergence tolerance for Poisson and negative binomial models. Passed to glm.control
- **init.theta**: Initial value of theta in negative binomial model
- **start**: starting values of coefficients in linear predictor
- **trace**: Logical indicating if output should be produced for each of model fitting procedure. Passed to glm.control or gamlss.control
- **verbose**: Logical indicating whether to print progress reports.

Value

List with elements

- **model**: model object. Set to NULL if no model could be fit.
- **expected.values**: vector of expected values for each element in original data, or vector of NAs if no model could be fit
- **p.values**: vector of p-values for test of significantly higher response than expected, or vector of NAs if no model could be fit

Description

Perform multiple testing correction on p-values from interaction test. By default, multiple testing correction is applied per bait. To change this to a global multiple testing correction, set bait.level = FALSE.

Usage

```r
multiple.testing.correct(interaction.data, bait.level = TRUE)
```

Arguments

- **interaction.data**: Data table of interaction calls. Must contain columns p.value and bait.id.
- **bait.level**: Logical indicating whether multiple testing correction should be performed per bait.
Value

Original data table with new column

q.value  FDR-corrected p-value

Examples

data(bre80);
results <- fit.model(bre80);
adjusted.results <- multiple.testing.correct(results);

Description

Prepare data for running interaction calling. Takes a BAM file and baits and restriction fragments as input, and returns a data table with data ready for analysis.

Usage

prepare.data(bam, baits, fragments, replicate.merging.method = "sum",
            include.zeros = c("none", "cis", "all"), remove.adjacent = FALSE,
            temp.directory = NULL, keep.files = FALSE, verbose = FALSE)

Arguments

bam  Path to a BAM file
baits  Path to a BED file containing the baits
fragments  Path to a BED file containing all restriction fragments in the genome
replicate.merging.method  Method that should be used for merging replicates, if applicable
include.zeros  String specifying what zero counts to include. Options are none (default), cis, and all.
remove.adjacent  Logical indicating whether to remove all reads mapping to adjacent restriction fragments.
temp.directory  Directory where temporary files should be stored. Defaults to current directory.
keep.files  Logical indicating whether to keep temporary files
verbose  Logical indicating whether to print progress reports.
### Value

Data table object with columns

- **target.id**: String in chrN:start-end format identifying target fragment
- **bait.id**: String in chrN:start-end format identifying bait fragment
- **target.chr**: Chromosome of target fragment
- **target.start**: Start coordinate of target fragment (zero-based)
- **target.end**: End coordinate of target fragment
- **bait.chr**: Chromosome of bait fragment
- **bait.start**: Start coordinate of bait fragment (zero-based)
- **bait.end**: End coordinate of bait fragment
- **bait.to.bait**: Boolean indicating if the interaction is bait-to-bait (i.e. the fragment listed as target is also a bait)
- **count**: The number of reads linking the two fragments
- **bait.trans.count**: The number of reads linking the bait to fragments in trans (a measure of "interactibility")
- **target.trans.count**: The number of reads linking the target to fragments in trans (a measure of "interactibility")
- **distance**: Distance between the midpoints of the bait and target fragments (basepairs). NA for trans interactions

### Examples

```r
bam <- system.file('extdata', 'Bre80_2q35.bam', package = 'chicane');
baits <- system.file('extdata', '2q35.bed', package = 'chicane');
fragments <- system.file('extdata', 'GRCh38_HindIII_chr2.bed.gz', package = 'chicane');
input.data <- prepare.data(
  bam = bam,
  baits = baits,
  fragments = fragments
);
```

---

### Description

Read a BED file and return regions in chrN:start-end format
**run.model.fitting**

**Usage**
```
read.bed(bed.path, zero.based = TRUE)
```

**Arguments**
- **bed.path**: Path to bed file
- **zero.based**: Whether to return ID in zero-based coordinates

**Value**
- vector of region IDs

**Examples**
```
bait.file <- system.file('extdata', '2q35.bed', package = 'chicane');
bait <- read.bed(bait.file);
```

---

**run.model.fitting**

**Description**
Run model fitting procedure for either bait-to-bait or other interactions. Meant for internal use only.

**Usage**
```
run.model.fitting(interaction.data, distance.bins = NULL,
                  distribution = "negative-binomial", bait.to.bait = FALSE,
                  adjustment.terms = NULL, maxit = 100, epsilon = 1e-08, cores = 1,
                  trace = FALSE, verbose = FALSE, interim.data.dir = NULL)
```

**Arguments**
- **interaction.data**: data.table object containing interaction counts. Must contain columns distance, count, and bait_trans_count.
- **distance.bins**: Number of bins to split distance into. Models are fit separately in each bin.
- **distribution**: Name of distribution of the counts. Options are 'negative-binomial', 'poisson', 'truncated-poisson', and 'truncated-negative-binomial'
- **bait.to.bait**: Logical indicating if model should be fit as bait-to-bait
- **adjustment.terms**: Character vector of extra terms to adjust for in the model fit
- **maxit**: Maximum number of IWLS iterations for fitting the model (passed to glm.control)
- **epsilon**: Positive convergence tolerance for Poisson and negative binomial models. Passed to glm.control
stratified.enrichment.sample

- **cores**
  Integer value specifying how many cores to use to fit model for cis-interactions.

- **trace**
  Logical indicating if output should be produced for each of model fitting procedure. Passed to glm.control or gamlss.control

- **verbose**
  Logical indicating whether to print progress reports.

- **interim.data.dir**
  Path to directory to store intermediate QC data and plots.

**Value**

Interactions data with expected number of interactions and p-values added.

---

**smart.split**

**Description**

Split a data frame into a prespecified number of bins, using split and cut. Unlike the default R functions, this does not fail when asked to split the data into a single bin.

**Usage**

`smart.split(dat, bins)`

**Arguments**

- **dat**
  Data frame or data table to be split

- **bins**
  Number of bins to split data into

**Value**

List with bins elements. Each element corresponds to one portion of the data

---

**stratified.enrichment.sample**

**Description**

Generate a stratified sample matching distance distribution of significant interactions.

**Usage**

`stratified.enrichment.sample(nonsignificant.results, significant.results)`
test.enrichment

Arguments

nonsignificant.results
Data table containing non-significant interactions that should be sampled from

significant.results
Data table of significant results. Used to determine size of strata in stratified
sampling procedure.

Description
test.enrichment

Usage
test.enrichment(interaction.data, feature.bed,
significance.cutoff = 0.05, span = 0, n = 1000,
remove.bait.to.bait = TRUE)

Arguments

interaction.data
Data table containing details on interactions

feature.bed
BED file with regions of features

significance.cutoff
q-value threshold for significant interactions

span
Distance around target restriction fragment to consider. If set to zero (default),
only features that overlap with the restriction fragment itself are considered.

n
Number of random samples to consider

remove.bait.to.bait
Logical specifying whether to exclude bait-to-bait interactions

Value

list with elements

observed observed overlap between significant interactions and features

random vector of length n giving overlap between random samples and features

Author(s)

Erle Holgersen <Erle.Holgersen@icr.ac.uk>
Examples

data(bre80);
ctcf.bed <- system.file('extdata', 'T47D_chr2_CTCF.bed.gz', package = 'chicane');
results <- chicane(interactions = bre80);
test.enrichment(results, ctcf.bed, significance.cutoff = 0.25);

Description

Verify that interaction.data object is in expected format. Throws an error if object does not fit requirements.

Usage

verify.interaction.data(interaction.data)

Arguments

interaction.data

Object to be verified.

Value

None
Index

*Topic datasets
  bre80, 4
  add.covariates, 2
  add.fragment.coordinates, 3
  bedtools.installed, 4
  bre80, 4
  check.model.numerical.fit, 5
  check.split.data.numerical.fit, 6
  chicane, 6
  combine.replicates, 9
  convert.bam, 10
  convert.hicup.digest.bed, 11
  distance.bin, 11
  distance.split, 12
  fill.in.zeros(fill.in.zeros), 13
  fill.in.zeros, 13
  filter.fragments, 13
  fit.glm, 14
  fit.model, 15
  get.combination.count, 16
  get.distance, 17
  get.id.components, 18
  get.trans.counts, 18
  is.glm.nb.maxiter.warning, 19
  is.glm.nb.theta.error, 19
  is.glm.nb.theta.warning, 20
  make.modelfit.plot, 20
  model.rows.sanity.check, 21
  model.try.catch, 21
  multiple.testing.correct, 22
  prepare.data, 10, 23
  read.bed, 24
  run.model.fitting, 25
  smart.split, 26
  stratified.enrichment.sample, 26
  test.enrichment, 27
  verify.interaction.data, 28