Package 'chromoR'

February 19, 2015

Type Package

Title Analysis of chromosomal interactions data (correction, segmentation and comparison)

Version 1.0

Date 2014-02-06

Author Yoli Shavit

Maintainer Yoli Shavit <ys388@cam.ac.uk>

Description chromoR provides users with a statistical pipeline for analysing chromosomal interactions data (Hi-C data).It combines wavelet methods and a Bayesian approach for correction (bias and noise) and comparison (detecting significant changes between Hi-C maps) of Hi-C contact maps.In addition, it also support detection of change points in 1D Hi-C contact profiles.

License GPL

Depends R (>= 2.15.2), haarfisz, gdata

NeedsCompilation no

Repository CRAN

Date/Publication 2014-02-11 17:34:44

R topics documented:

chromoR-package *Analysis of chromosomal interactions data (Hi-C data)*

Description

ChromoR combines wavelet change point with Bayes Factor, for useful correction, segmentation and comparison of Hi-C contact maps. It provides a user friendly software solution, addressing the entire statistical pipeline required for the analysis of chromosomal interactions data.

Details

Package:	chromoR
Type:	Package
Version:	1.0
Date:	2014-02-07
License:	GPL-2

For an easy start with chromoR, check out the documentation and examples for correctCIM, compareCIM and segmentCIM

See also http://www.cl.cam.ac.uk/~ys388/chromoR/ for more examples and data sets.

Author(s)

Yoli Shavit <ys388@cam.ac.uk>

References

http://www.cl.cam.ac.uk/~ys388/chromoR/

buildCIM *Builds Chromosomal Interactions Maps*

Description

This function takes mapped positions of fragments pairs (Hi-C data) in a given format (supported formats are "nodup", "maq" or "sam") and genome coordinates of all relevant regions (segmentation)and writes pairwise contact maps for all chromosome pairs. A cell M[i,j] in the pairwise matrix generetaed for a pair of chromosomes, chromosomeA and chromosomeB takes the values of the number of interactions between region i in chromosome A and region j in chromosome B (B and A may be the same chromosome). To improve processing times, this function calls a python executable. Thus, users should verify python (> 2.6) is installed and added to their PATH.

Usage

```
buildCIM(HiCFile, segFile, format, outputPrefix, resolution, header = FALSE,
inclusive = FALSE, verbose = TRUE, combineToSingle = TRUE)
```

Arguments

HiCFile	The name of the Hi-C file. See for example: http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM455134
segFile	The name of the segmentation file. The file provides the genomic coordinates of each region. It should be a tab delimited file with the following columns: chormosome, start, end, giving the chromosome name, start and end positions of each region.
format	The format of the Hi-C file, taking one of the following values: "nodup", "maq" or "sam"
outputPrefix	The prefix of the output files generated by this function. Each file is appended with the name of the 2 chromosomes, that correspond to the output contact map.
resolution	An integer value specifying the resolution of the given segmentation, if applicable. Specifically, if the segmentation file defines regions of the same size (for example: 1000000) this variable should be set accordingly. Otherwise it should be set to -1. Note that specifying the resolution greatly improves processing times.
header	optional: a boolean specifying whether the segmentation file includes a header or not. Set to FALSE by default
inclusive	optional: a boolean specifying whether the segmentation is inclusive. (i.e. whether the end position of one region overlaps with the start position of the next region). Set to FALSE by default.
verbose	optional: a boolean specifying whether to report on the progress of the CIM build. Set to TRUE by default.
combineToSingle	optional: a boolean specifying whether to also combine all the pairwise matrices into a single matrix and write it to a file.If set to TRUE, an additional file will be written, depending on available memory. Set to TRUE by default.

Value

This function generates a file for every pairwise chromosomal interaction map from the given input. No value is returned.

Note

Users should note that for large Hi-C files (> 10Gb), the pre-processing time is typically long (30-60 minutes).In order to generate Hi-C mapped positions given raw fragments pairs users should refer to related pipelines such as the HiCuP pipeline (http://www.bioinformatics.babraham.ac.uk/projects/hicup/). Additionally, different Hi-C data sets (raw fragment pairs and mapped positions) are publicly available from the Gene Expression Omnibus (GEO): http://www.ncbi.nlm.nih.gov/geo/

Author(s)

Yoli Shavit

References

http://www.cl.cam.ac.uk/~ys388/chromoR/

See Also

See Also as correctCIM, correctPairCIM

`compareCIM` *Compares 2 CIMs*

Description

Finds regions pairs that present significantly different interaction frequency between 2 chromosomal interactions maps (CIMs). Specifically, a Bayes Factor (BF) term is calculated for each pair and pairs with a BF score above a given threshold are reported.

Usage

```
compareCIM(m1, m2, seg, bfThreshold = 6.1)
```

Arguments

`m1`	The first CIM (matrix) to be compared, where the cell m[i,j] is the interaction frequency between the genomic region i and the genomic region j.
`m2`	The second CIM (matrix) to be compared, where the cell m[i,j] is the interaction frequency between the genomic region i and the genomic region j. (m1 and m2 should be of the same dimensions)
`seg`	A data table definding a genome segmentation, giving the genomic coordinates of each row/col in the given CIM. The data table should include the following columns for each region: chromosome name, start position and end position. Users may also provide additional properties in the the next columns such as gene expression or cytogenetic band.
`bfThreshold`	Optional: the Bayes Factor cuttoff (set to 6.1 by default).

Value

Returns a list with the following objects:

`sigChanges`	a data table listing all significant changes: the corrdinates of each region, the Bayes Factor value, the original and standartized values of the region pair in m1 and m2
`mBf`	the Bayes Factor matrix giving the Bayes Factor value for each regions pair

Author(s)

Yoli Shavit

References

http://www.cl.cam.ac.uk/~ys388/chromoR/ (including more examples and data sets)

See Also

See Also segmentCIM

Examples

```
data(imr90.1) # corrected contact map of imr90 (replicate 1) for chromosomes 1,2
data(imr90.2) # corrected contact map of imr90 (replicate 2) for chromosomes 1,2
data(seg)
# compare replicates of the same celll type
res = compareCIM(imr90.1, imr90.2, seg)
# no significant changes identified
res$sigChanges
```

---

correctCIM *Corrects a chromosomal interactions map (CIM)*

Description

This function applies a wavelet based correction to remove bias and noise, typical of Hi-C data, from a chromosomal interactions map (CIM).

Usage

```
correctCIM(m, seg, removeUncovered = FALSE)
```

Arguments

m	A CIM (matrix), where the cell m[i,j] is the interaction frequency between the genomic region i and the genomic region j.
seg	A data table definding a genome segmentation, giving the genomic coordinates of each row/col in the given CIM. The data table should include the following columns for each region: chromosome name, start position and end position.
removeUncovered	Optional: a boolean specifying whether regions that are uncovered (i.e. no interaction with other regions) should be removed before correction. set to FALSE by default.

Value

Returns a list with the following objects:

mCorrected	the corrected contat map
seg	a data table defining the genomic coordinates of each row/col in the corrected map (if removeUncoevered is set to FALSE, this would be the same as the input segmentation)

Author(s)

Yoli Shavit

References

http://www.cl.cam.ac.uk/~ys388/chromoR/ (including more examples and data sets)

See Also

See Also correctPairCIM, compareCIM

Examples

```
# correction
data(imr90.1.obs) #contact map of imr90 (replicate 1)
                  #for chromosomes 1 and 2, before correction
data(seg.imr90.obs)
head(seg.imr90.obs)
res = correctCIM(imr90.1.obs, seg.imr90.obs, removeUncovered = TRUE)
# the corrected CIM - look at the first cell in the matrix.
# Note the difference in dimensions
#(with respect to imr90.1.orig) since we set removeUncevered to TRUE
(res$mCorrected)[1,1]
dim(res$mCorrected)
# the coordinates for the corrected matrix
head(res$seg)
nrow(res$seg)
```

---

correctPairCIM *Corrects a pairwise CIM*

Description

This function applies a wavelet based correction to remove bias and noise, typical of Hi-C data, from pairwise chromosomal interaction maps(CIM).

Usage

```
correctPairCIM(m, isCis)
```

Arguments

m	A pairwise (i.e between 2 chromosomes) CIM (matrix), where the cell m[i,j] is the interaction frequency between the genomic region i and the genomic region j.
isCis	A boolean specifying whether this matrix describes cis (the same chromosome) or trans (different chromosomes) pairwise contacts.

Value

Returns the corrected CIM

Author(s)

Yoli Shavit

References

http://www.cl.cam.ac.uk/~ys388/chromoR/

See Also

See Also correctCIM, compareCIM

Examples

```
# correction of a single pair
data(imr90.1.obs)
data(seg.imr90.obs)
# take a pairwise matrix and correct it
indices = which(seg.imr90.obs$chr == "chr1")
i1 = indices[1]
iN = indices[length(indices)]
indices = which(seg.imr90.obs$chr == "chr2")
j1 = indices[1]
jN = indices[length(indices)]
m.1.2 = imr90.1.obs[i1:iN, j1:jN]
# set isCis to FALSE because we correct for a pair of different chromosomes
m.1.2.corrected = correctPairCIM(m.1.2, FALSE)
m.1.2.corrected
```

---

imr90.1 *A corrected CIM of replicate 1 of IMR90, for chromosomes 1 and 2*

Description

A corrected chromosomal interaction map (CIM) for replicate 1 of IMR90, for chromosomes 1 and 2, generated from a publicly available Hi-C data set [1]. This matrix was built with BuildCIM given Hi-C mapped positions (publicly available from

http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE35156) and then further corrected with CorrectCIM.

Usage

```
data(imr90.1)
```

Format

A matrix

References

[1] J. R. Dixon, S. Selvaraj, F. Yue, A. Kim, Y. Li, Y. Shen, M. Hu, J. S. Liu and B. Ren, Nature, 2012, 485, 376.

See also http://www.cl.cam.ac.uk/~ys388/chromoR/ for the genome wide CIM.

Examples

```
data(imr90.1)
```

---

imr90.1.obs *A non-corrected CIM of replicate 1 of IMR90, for chromosomes 1 and 2*

Description

A non corrected chromosomal interaction map (CIM) for replicate 1 of IMR90, for chromosomes 1 and 2, generated from a publicly available Hi-C data set [1]. This matrix was build with BuildCIM given Hi-C mapped positions, publicly available

from http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE35156

Usage

```
data(imr90.1.obs)
```

Format

A matrix

References

[1] J. R. Dixon, S. Selvaraj, F. Yue, A. Kim, Y. Li, Y. Shen, M. Hu, J. S. Liu and B. Ren, Nature, 2012, 485, 376.

See also http://www.cl.cam.ac.uk/~ys388/chromoR/ for the genome wide CIM.

Examples

```
data(imr90.1.obs)
```

imr90.2 *A CIM of replicate 2 of IMR90, for chromosomes 1 and 2*

Description

A corrected chromosomal interaction map (CIM) for replicate 2 of IMR90, for chromosomes 1 and 2, generated from a publicly available Hi-C data set [1]. This matrix was built with BuildCIM given Hi-C mapped positions (publicly available from http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE35156) and then further corrected with CorrectCIM.

Usage

```
data(imr90.2)
```

Format

A matrix

References

[1] J. R. Dixon, S. Selvaraj, F. Yue, A. Kim, Y. Li, Y. Shen, M. Hu, J. S. Liu and B. Ren, Nature, 2012, 485, 376.

See also http://www.cl.cam.ac.uk/~ys388/chromoR/ for the genome wide CIM.

Examples

```
data(imr90.2)
```

seg *A segmentation for imr90.1 and imr90.2 CIMs*

Description

A data table definding the genomic coordinates of each row/col in the corresponding CIMs (genome segmentation).

Usage

```
data(seg)
```

Format

A data frame 3 variables: chr, start, end

Examples

```
data(seg)
```

seg.imr90.obs *A segmentation for imr90.1.obs CIM*

Description

A data table definding the genomic coordinates of each row/col in imr90.1.obs (genome segmentation).

Usage

```
data(seg.imr90.obs)
```

Format

A data frame with 3 variables: chr, start, end

Examples

```
data(seg.imr90.obs)
```

segmentCIM *Segments a 1D contact profile (1D CIM)*

Description

This function finds changes points in a 1D Hi-C contact profile generated by taking the row sums of a pairwise chromosomal interactions map (CIM). It applies a wavelet change point algorithm [1] in order to find the location of change points (the segmentation of the profile).

Usage

```
segmentCIM(y, minL = 1, maxL = 1)
```

Arguments

y	A 1D Hi-C contact profile (a vector generated by taking row sums of a pairwise chromosomal interactions map (CIM)).
minL	Optional: a positive integer specifying the minimum level at which the search should be carried out. Set to 1 by default
maxL	Optional: a positive integer specifying the maximum level at which the search should be carried out and set to 1 by default. Should not exceed 0.1 of the length of the input. If the algorithm cannot find change points at the minimum level, it will repeat the search at higher levels and stop at the first level where change points were found or when it has reached the maximum level.

Details

This function implemments a wavelet change point algorithm as described in [1] and based on its matlab implementation (http://homepages. ulb.ac.be/~majansen/software/index.html)

Value

Returns a list including the following objects:

changePoints	the location of change points (or NULL if none were found)
L	the level at which the change points were found (or the given maximal level if no change points were found)

Author(s)

Yoli Shavit

References

website: http://www.cl.cam.ac.uk/~ys388/chromoR/ (including more examples and data sets)

[1] M. Jansen, Chemometrics and intelligent laboratory systems, 2007, 85, 159.

See Also

See Also as compareCIM

Examples

```
data(imr90.1) # corrected contact map of imr90 (replicate 1) for chromosomes 1,2
data(seg)
#generate a 1D profile for cis CIM of chromosome 2
indices = which(seg$chr == "chr2")
i1 = indices[1]
iN = indices[length(indices)]
p = rowSums(imr90.1[i1:iN, i1:iN]) - diag(imr90.1[i1:iN, i1:iN])
res = segmentCIM(p)
res
```

Index