Package: scHiCcompare (via r-universe)

October 31, 2026

Title Differential Analysis of Single-cell Hi-C Data

Version 1.5.0

Description This package provides functions for differential chromatin interaction analysis between two single-cell Hi-C data groups. It includes tools for imputation, normalization, and differential analysis of chromatin interactions. The package implements pooling techniques for imputation and offers methods to normalize and test for differential interactions across single-cell Hi-C datasets.

Imports grDevices, graphics, stats, utils, dplyr, ggplot2, gtools, HiCcompare, lattice, mclust, mice, miceadds, ranger, rstatix, tidyr, rlang, data.table, BiocParallel

Suggests knitr, rmarkdown, testthat, BiocStyle, DT, gridExtra

License MIT + file LICENSE

Encoding UTF-8

LazyData false

RoxygenNote 7.3.2

Depends R (>= 4.5.0)

VignetteBuilder knitr

biocViews Software, SingleCell, HiC, Sequencing, Normalization

BugReports https://github.com/dozmorovlab/ScHiCcompare/issues

URL https://github.com/dozmorovlab/ScHiCcompare

Config/pak/sysreqs cmake make libicu-dev libssl-dev libx11-dev zlib1g-dev

Repository https://bioc.r-universe.dev

Date/Publication 2026-04-28 13:04:58 UTC

RemoteUrl https://github.com/bioc/scHiCcompare

RemoteRef HEAD

RemoteSha b574f248db2c15179314a0b2f94912c565b5905b

Contents

ODC.bandnorm_chr20_1 . . . 2
plot_HiCmatrix_heatmap . . . 3
plot_imputed_distance_diagnostic . . . 4
print.scHiCcompare . . . 5
pseudo_bulkHic . . . 6
read_files . . . 7
scHiC.table_MG_chr22 . . . 8
scHiC.table_ODC_chr22 . . . 10
scHiC_bulk_compare . . . 11
scHiC_table . . . 14
scHiCcompare . . . 15
scHiCcompare_impute . . . 18

Index **20**

`ODC.bandnorm_chr20_1` *scHi-C data from oligodendrocyte (ODC) cell type - chromosome 20 at 1 MB resolution*

Description

A Modified sparse upper triangular matrix containing the interacting regions cordination (chr1, start1, chr2, start2) and the corresponding Interaction Frequency (IF).

Usage

```
data("ODC.bandnorm_chr20_1")
```

Format

A dataframe with 335 rows and 5 columns:

chr The first interacting region chromosome

start1 The first interacting region starting cordination

chr The second interacting region chromosome

start2 The second interacting region starting cordination

IF The interaction frequency value between two regions

Value

A data frame

Source

Data downloaded by download_schic() by 'Bandnorm'. See their website at https://sshen82.github.io/BandNorm/articles/BandNorm-tutorial.html#download-existing-single-cell-hi-c-data The data is a single-cell Hi-C of ODC cell type of Lee et al. public dataset. The GEO link to download the data https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE130711

Examples

```
data("ODC.bandnorm_chr20_1")
```

`plot_HiCmatrix_heatmap`

Plot Hi-C Interaction Matrix Heatmap

Description

This function generates a heatmap of Hi-C interaction frequencies from a given sparse matrix, allowing users to visualize either original or imputed Hi-C data.

Usage

```
plot_HiCmatrix_heatmap(
  scHiC.sparse,
  zlim = NULL,
  color_low = "white",
  color_high = "red",
  main = NULL,
  figure_name = NULL
)
```

Arguments

`scHiC.sparse`	A modified sparse matrix of Hi-C interaction frequencies in the format (chr1, start1, chr2, start2, IF).
`zlim`	A numeric vector of length 2 specifying the limits of the color scale. If 'zlim' is not specified, it will include the minimum and maximum values of the matrix. For example, 'zlim = c(0, 10)' limits the color scale to values between 0 and 10.
`color_low`	A character string specifying the color for the lowest values. Default is "white". Other options include colors such as "lightblue", "yellow", or colors defined using hex codes (e.g., "#FFFFFF" for white). Users can refer to the R color names documentation by running 'colors()' in R.
`color_high`	A character string specifying the color for the highest values. Default is "red". Other options include colors such as "lightblue", "yellow", or colors defined using hex codes (e.g., "#FF0000" for red). Users can refer to the R color names documentation by running 'colors()' in R.
`main`	A character string for the main title of the plot. The default is NULL.
`figure_name`	A character string for additional figure labeling. The default is NULL.

Value

A heatmap plot visualizing the Hi-C interaction matrix.

Examples

```
data("ODC.bandnorm_chr20_1")

# Call the function with this sparse matrix to generate the heatmap
plot_HiCmatrix_heatmap(
  scHiC.sparse = ODC.bandnorm_chr20_1,
  zlim = c(0, 7), # Log scale color limits
  color_low = "white", # Color for low values
  color_high = "red", # Color for high values
  main = "Single-Cell Hi-C Heatmap", # Title of the plot
  figure_name = "Example Heatmap" # Subtitle for the plot
)
```

---

plot_imputed_distance_diagnostic

Plot Imputed Distance Diagnostic

Description

This function plots the distribution density curves of interaction frequencies from original and imputed single-cell Hi-C data at a given genomic distance.

Usage

```
plot_imputed_distance_diagnostic(raw_sc_data, imp_sc_data, D)
```

Arguments

raw_sc_data	A data frame containing the original single-cell Hi-C data in scHiC Table format. The data frame should include columns for 'region1', 'region2', 'Cell', 'Chr', and 'IF_i' for each of the "i" cells.
imp_sc_data	A data frame containing the imputed single-cell Hi-C data in scHiC Table format. The data frame should include columns for 'region1', 'region2', 'Cell', 'Chr', and 'IF_i' for each of the "i" cells.
D	An integer specifying the genomic distance for which to plot the density curves of interaction frequencies. Genomic distance refers to the distance between two regions in the genome, expressed in units of resolution bins (e.g., D = (start2 - start1)/resolution). For example, if the genomic regions are at positions 16,000,000 and 17,000,000 with a resolution of 1,000,000, then D = (17,000,000 - 16,000,000)/1,000,000 = 1.

Details

The distance D represents how far apart two genomic loci are. Observations indicate that chromatin interaction frequency (IF) in Hi-C data tends to decrease as the genomic distance between two loci increases. It is assumed that scHi-C interaction frequencies at the same distance share similar statistical properties. To diagnose whether the imputed IF values retain these statistical properties at each distance, the ‘plot_imputed_distance_diagnostic()‘ function plots the density curves of interaction frequencies from original and imputed single-cell Hi-C data for a specified genomic distance.

Value

A ggplot2 object representing the density plot comparing the original and imputed interaction frequencies.

Examples

```
data("scHiC.table_MG_chr22")
## Impute data above
scHiC.table_MG_imp <- scHiCcompare_impute(scHiC.table = scHiC.table_MG_chr22)

# Call the function with this sparse matrix to generate the plot
plot_imputed_distance_diagnostic(
  raw_sc_data = scHiC.table_MG_chr22,
  imp_sc_data = scHiC.table_MG_imp, D = 1
)
```

---

print.scHiCcompare *Print method for ‘scHiCcompare‘ objects*

Description

This method is used to print a summary of the ‘scHiCcompare‘ object. It provides an overview of the number of cells in each condition, the chromosome analyzed, and the processes involved in the analysis.

Usage

```
## S3 method for class 'scHiCcompare'
print(x, ...)
```

Arguments

x	An object of class ‘scHiCcompare‘. This object should contain intermediate results from the differential analysis of single-cell Hi-C data.
...	Further arguments to be passed to or from other methods.

Value

This function does not return any value. It is called for its side effect, which is printing a summary of the ‘scHiCcompare‘ object.

Examples

```
## Load example data for ODC and MG file paths
ODCs_example <- system.file("extdata/ODCs_example",
  package = "scHiCcompare"
)
MGs_example <- system.file("extdata/MGs_example",
  package = "scHiCcompare"
)

## Run scHiCcompare on example data
result <- scHiCcompare(
  file.path.1 = MGs_example,
  file.path.2 = ODCs_example,
  select.chromosome = "chr20",
  Plot = FALSE
)
print(result)
```

---

pseudo_bulkHic *Generate Pseudo-bulk Hi-C Data*

Description

This function generates pseudo-bulk Hi-C data from a single-cell Hi-C interaction frequency table. It returns the data in either sparse or full matrix format.

Usage

```
pseudo_bulkHic(scHiC.table, out = "sparse")
```

Arguments

scHiC.table	A data frame containing interaction frequencies across cells that are created by the scHiC_table() function. The first four columns should represent 'cell', 'chr', 'region1', and 'region2',followed by columns representing interaction frequencies ('IF') for individual cells.
out	A character string specifying the output format. It must be either 'sparse' or 'full'. Default is 'sparse'.

Value

A data frame representing the pseudo-bulk Hi-C data. If ‘out‘ is ’sparse’; returns a data frame containing a pseudo-bulk matrix in the sparse upper triangular Hi-C matrix format with five columns: chromosome 1 (chr1), start position 1 (start1), chromosome 2 (chr2), start position 2 (start2), and interaction frequency (IF). If ‘out‘ is ’full’; returns a full matrix representation of the pseudo-bulk data.

References

Stansfield JC, Cresswell KG, Vladimirov VI et al (2018). Hiccompare: an R-package for joint normalization and comparison of HI-C datasets. BMC Bioinformatics 2018;19:279.

Examples

```
data("scHiC.table_MG_chr22")
data("scHiC.table_ODC_chr22")
pseudo_bulk_MG <- pseudo_bulkHic(scHiC.table_MG_chr22)
pseudo_bulk_ODC <- pseudo_bulkHic(scHiC.table_ODC_chr22)
head(pseudo_bulk_MG)
head(pseudo_bulk_ODC)
```

---

read_files *Read Files for scHi-C Dataset*

Description

This function reads single-cell Hi-C data from a specified file path. It supports two file formats: ’txt’ and ’cool’. For ’txt’, it reads tab-delimited files and assumes the format contains five columns: chromosome 1 (chr1), start position 1 (start1), chromosome 2 (chr2), start position 2 (start2), and interaction frequency (IF). For ’cool’, it uses the ‘HiCcompare‘ package to transform cooler files to BEDPE format.

Usage

```
read_files(
  file.path,
  position.dataset = NULL,
  type = "txt",
  txt.sparse.heads.position = NULL,
  out = "sparse"
)
```

Arguments

`file.path`	The directory path where the data files are stored.
`position.dataset`	A vector of indices specifying the file positions to read from the directory. These indices help select specific files, determining which files will be included and the sequence in which they are processed. If all single cell data should be included, set this to NULL.
`type`	The file type, either 'txt'. The default is 'txt'. Each 'txt' file should be in the format of a sparse upper triangular Hi-C matrix, where each row contains the interaction frequency value (IFs) of two interacting regions.
`txt.sparse.heads.position`	A vector of four integers specifying the column positions of chromosome, start1, start2, and IF in the 'txt' file.
`out`	Character string specifying the output format. Options are "sparse" (for a sparse matrix format) or "original" (to retain the original structure of each single-cell Hi-C dataset). Default is "sparse".

Details

This function reads single-cell Hi-C data in 'txt', with output options of 'sparse' and 'original'. Each input 'txt' file should be in the form of a sparse upper triangular Hi-C matrix, storing pairwise interaction frequencies of loci pairs. The 'txt' dataset should have one column indicating the interaction frequency (IF) of each pair of interacting regions, with tab-separated columns and no row names, column names, or quotes around character strings.

Value

A list of datasets, where each element corresponds to a dataset from the selected files. If 'out' is "sparse", each dataset element is transformed into a sparse matrix format (chr, start1, start2, IF). If 'out' is "original", the original structure of each single-cell Hi-C dataset is preserved.

Examples

```
# Load MG data folder example
MGs_example <- system.file("extdata/MGs_example", package = "scHiCcompare")
datasets <- read_files(
  file.path = MGs_example, position.dataset = c(1, 2, 3, 4, 5),
  txt.sparse.heads.position = c(1, 2, 3, 4, 5)
)
```

`scHiC.table_MG_chr22` *scHi-C table from microglia (MG) cell type - chromosome 22 at 1 MB resolution*

Description

A table that gathers 10 single-cell Hi-C data modified into a sparse upper triangular matrix. It contains each cell's interacting region coordinates (cell, chr, start1, start2) and the corresponding Interaction Frequencies (IF) for each single cell (IF_1, IF_2, ..., IF_10).

Usage

```
data("scHiC.table_MG_chr22")
```

Format

A data frame with 335 rows and 5 columns:

cell Name of the target cell type

chr The chromosome of the interacting region's starting coordinate

region1 The starting coordinate of the first interacting region

region2 The starting coordinate of the second interacting region

IF_1 Interaction frequency values for single cell 1

IF_2 Interaction frequency values for single cell 2

IF_3 Interaction frequency values for single cell 3

IF_4 Interaction frequency values for single cell 4

IF_5 Interaction frequency values for single cell 5

IF_6 Interaction frequency values for single cell 6

IF_7 Interaction frequency values for single cell 7

IF_8 Interaction frequency values for single cell 8

IF_9 Interaction frequency values for single cell 9

IF_10 Interaction frequency values for single cell 10

Value

A data frame

Source

Single-cell data of MG cell type downloaded using 'download_schic()' from 'Bandnorm'. See their website at https://sshen82.github.io/BandNorm/articles/BandNorm-tutorial.html#download-existing-single

Examples

```
data("scHiC.table_MG_chr22")
head(scHiC.table_MG_chr22)
```

scHiC.table_ODC_chr22 *scHi-C table from oligodendrocyte (ODC) cell type - chromosome 22 at 1 MB resolution*

Description

A table gather 10 single cells Hi-C data modified in sparse upper triangular matrix containing the cell's interacting regions coordination (cell, chr, start1, start2) and the corresponding Interaction Frequencies for each single cell (IF_1, IF2,... IF10).

Usage

```
data("scHiC.table_ODC_chr22")
```

Format

An object of class `data.frame` with 666 rows and 14 columns.

Details

cell Name of the target cell type

chr The chromosome of the interacting region's starting coordinate

region1 The starting coordinate of the first interacting region

region2 The starting coordinate of the second interacting region

IF_1 Interaction frequency values for single cell 1

IF_2 Interaction frequency values for single cell 2

IF_3 Interaction frequency values for single cell 3

IF_4 Interaction frequency values for single cell 4

IF_5 Interaction frequency values for single cell 5

IF_6 Interaction frequency values for single cell 6

IF_7 Interaction frequency values for single cell 7

IF_8 Interaction frequency values for single cell 8

IF_9 Interaction frequency values for single cell 9

IF_10 Interaction frequency values for single cell 10

Value

A data frame

Source

Single-cells data of ODC cell type download by download_schic() by 'Bandnorm'. See their website at https://sshen82.github.io/BandNorm/articles/BandNorm-tutorial.html#download-existing-single-ce

Examples

```
data("scHiC.table_ODC_chr22")
```

---

scHiC_bulk_compare *Compare Bulk Hi-C Data*

Description

This function compares single-cell Hi-C data between two groups using the 'scHicCompare' differential analysis workflow. It detects chromatin interaction differences between the single-cell Hi-C data of two cell types or conditions by Gaussian Mixture Model (GMM) cluster and outlier differences by 'hic_compare()' analysis of 'HiCcompare'

Usage

```
scHiC_bulk_compare(
  norm.hic.table,
  D.interval,
  fprControl.logfc = 0.8,
  alpha = 0.05,
  SD = 2,
  numChanges = NULL,
  FC = 3,
  A.min = NULL,
  Plot = TRUE,
  BPPARAM = bpparam()
)
```

Arguments

norm.hic.table	A data frame representing a jointly normalized pseudo-bulk Hi-C table output from two conditions, generated by the 'hic_loess()' function. This should be a pre-processed table that has been jointly normalized before differential analysis.
D.interval	A numeric vector defining the distance intervals to consider in the analysis, or a character string 'full' indicating the inclusion of all genomic distances in the analysis. Genomic distance refers to the number of base pairs between two regions in the genome (e.g., loci or bins) that is scaled by resolution D = (start2 - start1)/resolution (e.g., D = (16,000,000 - 17,000,000)/1,000,000 -> D = 1)
fprControl.logfc	A numeric value controlling the false positive rate by setting the threshold for the log fold change in the 'difference' cluster. Detected differences identified by Gaussian Mixed Model (GMM) clusters only include values that are larger than this threshold. Default is 0.8.
alpha	A numeric value for the significance level of outlier detection in the 'hic_compare()' function from HiCcompare. The default is 0.05.

SD — A numeric value specifying the standard deviation threshold for fuzzing, used to produce a simulated Hi-C matrix. This value is used to modify the process of finding the optimal 'A.min' quantile cutoff for detecting significant outliers. Users can select the value based on their assumption of the scHi-C data. The default is 2.

numChanges — An integer or NULL, indicating the number of changes to add to the simulated Hi-C matrix. This value is used to modify the process of finding the optimal 'A.min' quantile cutoff for detecting significant outliers. Based on the user's assumption about the possible number of differences, they can set the number of changes that should be proportional to the resolution of the data. High-resolution data should be assumed to have more changes. If 'numChanges' = NULL, the function is set to the number of changes (or simulated difference) scaled by a factor of 30 (e.g., 1MB resolution - 30 changes, 500KB resolution - 60 changes, etc.) The default is NULL.

FC — A numeric value representing the fold change threshold added to the simulated Hi-C matrix. This value is used to identify the optimal 'A.min' quantile cutoff for detecting significant outliers. Users can select the FC value based on their assumption of difference fold change in their data. The default is 3.

A.min — A numeric value or NULL, specifying the A-value quantile cutoff to filter lower average expression in the 'hic_compare()' function from HiCcompare. 'hic_compare()' is used to detect outliers, which is assumed to be 'differences' bins in case of its number is too small (or none) to be clustered by GMM method. If not provided, an optimized minimum A threshold that maximizes MCC and TPR while minimizing FPR in the simulated Hi-C matrix.

Plot — A logical value indicating whether to plot the differential results in an MD plot. The default is TRUE.

BPPARAM — Parameters for 'BiocParallel', to be passed to the 'bpparam()' function. See ''?bpparam()' for more info.

Details

The 'scHiC_bulk_compare()' function performs differential chromatin interaction comparisons between single-cell Hi-C data from two cell type groups or conditions. The workflow includes clustering normalized log fold changes between interaction frequencies of 2 cell types at each genomic distance into "difference" and "non-difference" groups. The non-difference group is assumed to follow a normal distribution centered around 0 and is clustered by a Gaussian Mixture Model. The difference cluster consists of points belonging to other distributions. In cases where the size of the differences is not large enough to form distinct distributions, these differences are assumed to be outliers of the normal distribution, which are identified by the 'hic_compare()' function.

Value

A data frame containing the results of the differential Hi-C analysis, including a normalized report and a 'Difference.cluster' column indicating the clusters of differences identified in the analysis.

References

Stansfield JC, Cresswell KG, Vladimirov VI et al (2018). Hiccompare: an R-package for joint normalization and comparison of HI-C datasets. BMC Bioinformatics 2018;19:279.

Scrucca L., Fop M., Murphy T. B. and Raftery A. E. (2016) mclust 5: clustering, classification and density estimation using Gaussian finite mixture models, The R Journal, 8/1, pp. 289-317.

C. Fraley and A. E. Raftery (2007) Bayesian regularization for normal mixture estimation and model-based clustering. Journal of Classification, 24, 155-181.

Fraley C. and Raftery A. E. (2002) Model-based clustering, discriminant analysis and density estimation, Journal of the American Statistical Association, 97/458, pp. 611-631.

Patrick Royston (1982). Algorithm AS 181: The W test for Normality. Applied Statistics, 31, 176–180. doi:10.2307/2347986.

Patrick Royston (1982). An extension of Shapiro and Wilk's W test for normality to large samples. Applied Statistics, 31, 115–124. doi:10.2307/2347973.

Examples

```
# Load data folder example to the current working directory
ODCs_example <- system.file("extdata/ODCs_example", package = "scHiCcompare")
MGs_example <- system.file("extdata/MGs_example", package = "scHiCcompare")
# Input single-cell Hi-C in sparse format (.txt) from a path
scHiC.table_ODC <- scHiC_table(
  file.path = ODCs_example,
  cell.type = "ODC",
  select.chromosome = "chr20"
)
scHiC.table_MG <- scHiC_table(
  file.path = MGs_example,
  cell.type = "MG",
  select.chromosome = "chr20"
)
# Bulk matrix in sparse format
bulk.sparse.1 <- na.omit(pseudo_bulkHic(
  scHiC.table = scHiC.table_ODC,
  out = "sparse"
))
bulk.sparse.2 <- na.omit(pseudo_bulkHic(
  scHiC.table = scHiC.table_MG,
  out = "sparse"
))
# Create the `hic.table` object
library(HiCcompare)
bulk.hic.table <- create.hic.table(bulk.sparse.1, bulk.sparse.2,
  chr = "chr20", scale = FALSE
)
# Jointly normalize data for a single chromosome
hic.table_normalize <- hic_loess(bulk.hic.table,
  Plot = TRUE,
  Plot.smooth = FALSE
)
```

```
# Example usage of the BulkHiC_compare function
result <- scHiC_bulk_compare(hic.table_normalize,
  D.interval = c(1:10),
  fprControl.logfc = 0.8
)
```

---

scHiC_table *Create scHiC Interaction Frequency Table*

Description

This function generates a single-cell Hi-C interaction frequency (IF) table for all single cells for a selected chromosome. The resulting table is usable for the `Pooling_RF_impute()` and `pseudo_bulkHic()` functions. It reads the input files, extracts the relevant data, and outputs a table of interaction frequencies between genomic regions for each single cell dataset.

Usage

```
scHiC_table(file.path, cell.type, select.chromosome)
```

Arguments

`file.path`	Character string specifying the directory containing scHi-C data for condition (a cell-type group). The folder should contain '.txt' scHi-C files in a modified sparse upper triangular format (chr1, start1, chr2, start2, IF)
`cell.type`	The cell type name used in the analysis (e.g., 'NSN', 'SN').
`select.chromosome`	The chromosome name to be studied (e.g., 'chr1' or 'chrX').

Details

This function processes single-cell Hi-C data in a folder directory, then transforms them into a single 'scHiC table' data frame. Each element in the list should be in the form of a sparse upper triangular Hi-C matrix with five tab-separated columns (chr1, start1, chr2, start2, IF) with no row or column names and no quotes around character strings.

Value

A data frame containing the interaction frequency table with genomic loci (cell, chromosome, start1, end1) and interaction frequencies (IF) of each single cell. This table can be used with the `Pooling_RF_impute()` and `pseudo_bulkHic()` functions.

Examples

```
# Load MG data folder example
MGs_example <- system.file("extdata/MGs_example", package = "scHiCcompare")

# Create scHiC table to be used in scHiCcompare
IF_table <- scHiC_table(
  file.path = MGs_example, cell.type = "MG",
  select.chromosome = "chr20"
)
```

---

scHiCcompare *ScHiCcompare: Differential Analysis of Single-Cell Hi-C Data*

Description

This function performs a differential analysis between two single-cell Hi-C data groups. It includes the steps of imputation, normalization, and detection of differential chromatin interactions (DCIs).

Usage

```
scHiCcompare(
  file.path.1,
  file.path.2,
  select.chromosome,
  imputation = "RF",
  normalization = "LOESS",
  differential.detect = "MD.cluster",
  main.Distances = seq(1e+07),
  pool.style = "progressive",
  n.imputation = 5,
  maxit = 1,
  outlier.rm = TRUE,
  missPerc.threshold = 95,
  A.min = NULL,
  fprControl.logfc = 0.8,
  alpha = 0.05,
  Plot = TRUE,
  Plot.normalize = FALSE,
  save.output.path = NULL,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

file.path.1	Required character string specifying the directory containing scHi-C data for the first condition (first cell-type group). The folder should contain '.txt' scHi-C

files in modified sparse upper triangular format with 5 columns (chr1, start1, chr2, start2, IF).

`file.path.2` Required character string specifying the directory containing Hi-C data for the second condition (second cell-type group). The folder should contain '.txt' scHi-C files in modified sparse upper triangular format with 5 columns (chr1, start1, chr2, start2, IF).

`select.chromosome`
Required integer or character indicating the chromosome to be analyzed (e.g., 'chr1' or 'chr10').

`imputation` Character string 'RF' or NULL indicating the imputation method. Default is 'RF' for Random Forest imputation.

`normalization` Character string 'LOESS' or NULL indicating the normalization method. Default is 'LOESS'.

`differential.detect`
Character string 'MD.cluster' indicating the differential detection method. The default is 'MD.cluster'.

`main.Distances` Numeric vector indicating the range of interacting genomic distances (in base pairs) between two regions (e.g., loci or bins) to focus on (e.g., 1:100000, Inf, etc). The 'main.Distance' vector needs to be proportional to the data's resolution (e.g., for 10kb - 1:10000, 1:50000, 1:100000, Inf, etc). Selecting a large distance range at higher resolution (e.g., below 200kb) can make the function take longer to run due to extreme sparsity. The default is 1:10000000.

`pool.style` Character string specifying the pooling style for 'imputation'. Options are 'none', 'progressive', or 'Fibonacci'. The default is 'progressive'. If 'imputation' is NULL, then 'pool.style' should also be NULL.

`n.imputation` Integer specifying the number of multiple imputations for the imputation step, with final imputed values computed as the average of these multiple imputation values. Increasing the number of imputations enhances the accuracy of imputed values, though it may increase the imputation runtime. The default is 5.

`maxit` Integer specifying the maximum number of iterations for the internal refinement process within a single 'imputation' cycle. Increasing 'maxit' can help stabilize imputed values, though it may increase the imputation runtime. Default is 1.

`outlier.rm` Logical. If TRUE, outliers are removed during 'imputation'. The default is TRUE.

`missPerc.threshold`
Numeric value specifying the maximum allowable percentage of missing data in pool bands outside the 'main.Distances' to be imputed by the 'imputation' method. A higher threshold includes more sparse distances for imputation (e.g., above 95 percent), increasing memory and runtime, while a lower threshold (e.g., below 50 percent) might reduce the number of distances imputed. The default is 95.

`A.min` Numeric value or NULL that sets the A-value quantile cutoff (e.g., 7, 10, etc) for filtering low average interaction frequencies in outlier detection during the differential step of 'hic_compare()' from 'HiCcompare'. If not provided (NULL), A is auto-detected.

`fprControl.logfc`
: Numeric value controlling the false positive rate for GMM difference clusters (‘differential.detect‘) (e.g., 0.5, 0.8, 1, 1.5, etc). Increasing ‘fprControl.logfc‘ may reduce the false positive rate but can also reduce the number of chromatin interaction differences detected. Default is 0.8, equivalent to a 2-fold change.

`alpha`
: Numeric value for the significance level of outlier detection during the ‘differential.detect‘ step by ‘hic_compare()‘ from HiCcompare. The default is 0.05.

`Plot`
: Logical value indicates whether to plot the ‘differential.detect‘ results in an MD plot. The default is TRUE.

`Plot.normalize`
: Logical value indicates whether to plot the ‘normalization‘ results in an MD plot. The default is FALSE.

`save.output.path`
: Character string specifying the directory to save outputs, including the imputed cells in a modified sparse upper triangular format, a normalization result table, and a differential analysis result table. If NULL, no files are saved. The default is NULL.

`BPPARAM`
: Parameters for ‘BiocParallel‘, to be passed to the ‘bpparam()‘ function. See ‘?bpparam()‘ for more info.

Details

This function implements the ScHiCcompare workflow. It first reads sparse Hi-C data from two conditions and, by default, imputes missing interaction frequencies using a random forest model (RF) with the choice of ‘pool.style‘ (either progressive or Fibonacci). With the progressive pooling of interaction frequencies, genomic distance ranges increase linearly to form subsequent pooled bands, while Fibonacci pooling uses the Fibonacci sequence to increase the size of genomic distance ranges. Then, the random forest method is applied to individual genomic distance (’none’ pooling) or pooled bands (when ‘pool.style‘ is selected).

Next, pseudo-bulk Hi-C matrices are generated, followed by joint normalization using Loess regression (from ‘HiCcompare‘) before detecting differential chromatin interactions via a Gaussian Mixture Model (GMM) clustering approach. GMM clusters normalized log fold changes in interaction frequencies between the two cell types at each genomic distance into "difference" and "non-difference" groups. The non-difference group is assumed to follow a normal distribution centered around 0. The difference cluster comprises points that belong to other distributions. If the size of the differences is insufficient to form distinct distributions, these differences are identified by the ‘HiCcompare::hic_compare()‘ function.

Value

A custom class object ("checkNumbers") summarize information of workflow’s steps. At the same time, the object result also contain the differential analysis results and intermediate results (imputation, pseudo-bulk, normalization). If ‘save.output.path‘ is provided, the imputed results for both conditions are saved in a sparse format in the given directory. Normalization and differential analysis results are also saved if ‘save.output.path‘ is provided. See the vignette for more details.

Examples

```
## Load example data for ODC and MG file paths
```

```
ODCs_example <- system.file("extdata/ODCs_example", package = "scHiCcompare")
MGs_example <- system.file("extdata/MGs_example", package = "scHiCcompare")

## Run scHiCcompare on example data
result <- scHiCcompare(
  file.path.1 = MGs_example,
  file.path.2 = ODCs_example,
  select.chromosome = "chr20"
)
print(result)
```

`scHiCcompare_impute` *Random Forest Imputation with Pooling options for scHi-C Data*

Description

This function performs imputation of single-cell Hi-C (scHi-C) interaction frequencies (IF) using Random Forest imputation methods with different options for distance-based pooling strategies.

Usage

```
scHiCcompare_impute(
  scHiC.table,
  n.imputation = 5,
  maxit = 1,
  outlier.rm = TRUE,
  main.Distances = seq(1e+07),
  pool.style = "progressive",
  missPerc.threshold = 95
)
```

Arguments

`scHiC.table`	A data frame containing interaction frequencies across single cells, created by the 'scHiC_table' function. The first four columns should represent 'cell', 'chr', 'region1', and 'region2', followed by columns representing interaction frequencies ('IF') for individual cells.
`n.imputation`	An integer specifying the number of imputations to be performed. The default is 5.
`maxit`	An integer specifying the number of iterations for the internal refinement process within a single imputation cycle. The default is 1.
`outlier.rm`	A logical value indicate whether to remove outliers during the imputation process. The default is TRUE.
`main.Distances`	A vector of integers or 'full' representing the scHiC data in the main distance range to focus the imputation on, in bp units (e.g., 1:1,000,000). Genomic distances (in bp) are the number of base pairs between two regions in the genome (e.g., loci or bins). The default is from 1 to 10,000,000.

`pool.style` A string specifying the pooling technique to use. Options are 'none', 'progressive', or 'Fibonacci'. Default is 'progressive'.

`missPerc.threshold` An integer specifying the missing value percentage threshold in each pool band.

Details

The function first identifies important pools based on the given scHi-C genomic distance effect by pooling distance data according to the chosen method. For progressive pooling, pools of distances are consecutively combined to form larger sets, while Fibonacci pooling uses a Fibonacci sequence to combine distances. If the pooling style 'none' is selected, the band contains individual 1 genomic distance. During the imputation process, the function imputes all missing values (NAs) within each pool within the main distance range. For distances outside this main focus range, if any pool contains more than 'missPerc.threshold' missing values, it triggers an alternative imputation method, filling in missing values based on the mean for distances.

Value

A table in the format of a scHiC table (same structure as the output of the 'scHiC_table' function) with imputed interaction frequencies (IF) across all single cells. The output table is formatted with regions and single cells in wide format, with one column per single cell containing imputed IF values.

References

Doove, L.L., van Buuren, S., Dusseldorp, E. (2014), Recursive partitioning for missing data imputation in the presence of interaction Effects. Computational Statistics & Data Analysis, 72, 92-104.

Shah, A.D., Bartlett, J.W., Carpenter, J., Nicholas, O., Hemingway, H. (2014), Comparison of random forest and parametric imputation models for imputing missing data using MICE: A CALIBER study. American Journal of Epidemiology, doi:10.1093/aje/kwt312.

Van Buuren, S. (2018). Flexible Imputation of Missing Data. Second Edition. Chapman & Hall/CRC. Boca Raton, FL.

Examples

```
# Load MG data folder example
MGs_example <- system.file("extdata/MGs_example", package = "scHiCcompare")
# Create scHicCompare table to be used in scHicCompare
IF_table <- scHiC_table(
  file.path = MGs_example, cell.type = "MG",
  select.chromosome = "chr20"
)
# Example usage of Pooling_RF_impute
library(tidyr)
imputed_table <- scHiCcompare_impute(IF_table,
  n.imputation = 5, outlier.rm = TRUE,
  main.Distances = 1:10000000, pool.style = "progressive"
)
```

Index

∗ **datasets**
ODC.bandnorm_chr20_1, 2
scHiC.table_MG_chr22, 8
scHiC.table_ODC_chr22, 10

ODC.bandnorm_chr20_1, 2

plot_HiCmatrix_heatmap, 3
plot_imputed_distance_diagnostic, 4
print.scHiCcompare, 5
pseudo_bulkHic, 6

read_files, 7

scHiC.table_MG_chr22, 8
scHiC.table_ODC_chr22, 10
scHiC_bulk_compare, 11
scHiC_table, 14
scHiCcompare, 15
scHiCcompare_impute, 18