Package: methInheritSim (via r-universe)

June 12, 2026

Type Package

Title Simulating Whole-Genome Inherited Bisulphite Sequencing Data

Description Simulate a multigeneration methylation case versus control experiment with inheritance relation using a real control dataset.

Version 1.34.0

Date 2025-07-22

Author Pascal Belleau, Astrid Deschênes and Arnaud Droit

Author@R c(person(``Pascal", ``Belleau", email=``pascal_belleau@hotmail.com", role=c(``cre", ``aut")), person(``Astrid", ``Deschênes", email=``adeschen@hotmail.com", role=c(``aut")), person(``Arnaud", ``Droit", email=``arnaud.droit@crchuq.ulaval.ca", role=c(``aut")))

Depends R (>= 3.4)

Imports methylKit, GenomicRanges, Seqinfo, parallel, BiocGenerics, S4Vectors, methods, stats, IRanges, msm

Suggests BiocStyle, knitr, rmarkdown, RUnit, methylInheritance

Encoding UTF-8

License Artistic-2.0

URL https://github.com/belleau/methInheritSim

BugReports https://github.com/belleau/methInheritSim/issues

VignetteBuilder knitr

biocViews BiologicalQuestion, Epigenetics, DNAMethylation, DifferentialMethylation, MethylSeq, Software, ImmunoOncology, StatisticalMethod, WholeGenome, Sequencing

Maintainer Pascal Belleau <pascal_belleau@hotmail.com>

RoxygenNote 6.0.1

Config/pak/sysreqs make libbz2-dev libicu-dev liblzma-dev libxml2-dev libssl-dev xz-utils zlib1g-dev

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

Date/Publication 2026-04-28 12:45:44 UTC

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

RemoteRef RELEASE_3_23

RemoteSha 675d5f21aa6f52b236a719cedf55cc465dd6054e

Contents

`methInheritSim-package`

methInheritSim: Simulating Whole-Genome Inherited Bisulphite Sequencing Data

---

Description

This package generates simulations of multigeneration of bisulfite data using a real control dataset.

Author(s)

Pascal Belleau, Astrid Deschênes and Arnaud Droit

Maintainer: Pascal Belleau <pascal_belleau@hotmail.com>

See Also

- `runSim` for simulating a multigeneration methylation experiment with inheritance

---

`dataSimExample` *A* `list` *containing methylation information used by some internal functions (for demo purpose.*

Description

A `list` containing methylation information used by some internal functions (for demo purpose.

Usage

```
data(dataSimExample)
```

Format

a list containing:

- stateInfo a GRanges, a synthetic chromosome as generated by getSyntheticChr function.
- stateDiff a list containing:
  - stateDiff a vector of integer (0 and 1) with length corresponding the length of stateInfo. The vector indicates, using 1, the positions where the CpG sites are differentially methylated.
  - stateInherite a vector of integer (0 and 1) with length corresponding the length of stateInfo. The vector indicates, using 1, the positions where the CpG values are inherited.
- treatment a vector of integer (0 and 1) with length corresponding the number of samples. The vector indicates which samples are control (0) which samples are case (1).
- sample.id a list of 3 list. Each entry of the list correspond to one generation (first entry = first generation, etc..). Each list contains a list of 12 entries each containing a string of character, the name of the sample.

Value

a list containing:

- stateInfo a GRanges, a synthetic chromosome as generated by getSyntheticChr function.
- stateDiff a list containing:
  - stateDiff a vector of integer (0 and 1) with length corresponding the length of stateInfo. The vector indicates, using 1, the positions where the CpG sites are differentially methylated.
  - stateInherite a vector of integer (0 and 1) with length corresponding the length of stateInfo. The vector indicates, using 1, the positions where the CpG values are inherited.
- treatment a vector of integer (0 and 1) with length corresponding the number of samples. The vector indicates which samples are control (0) which samples are case (1).
- sample.id a list of 3 list. Each entry of the list correspond to one generation (first entry = first generation, etc..). Each list contains a list of 12 entries each containing a string of character, the name of the sample.

See Also

- runSim for running a simulation analysis using methylKit info as input

Examples

```
## Loading dataset
data(dataSimExample)

## Identify differentially methylated sites and among those, the ones
## that are inherited
methInheritSim:::getDiffMeth(stateInfo = dataSimExample$stateInfo,
```

```
rateDiff = 0.2, minRate = 0.3,propInherite = 0.3)
```

---

runSim *Simulate a multigeneration methylation experiment with inheritance*

Description

Simulate a multigeneration methylation case versus control experiment with inheritance relation using a real control dataset.

The simulation can be parametrized to fit different models. The number of cases and controls, the proportion of the case affected by the treatment (penetrance), the effect of the treatment on the mean of the distribution, the proportion of sites inherited, the proportion of the differentially methylated sites from the precedent generation inherited, etc..

The function simulates a multigeneration dataset like a bisulfite sequencing experiment. The simulation includes the information about control and case for each generation. Simulation dataset are saved in multiple files created in the directory specified by the user.

Usage

```
runSim(methData, nbSynCHR = 1, nbSimulation = 10, nbBlock = 100,
  nbCpG = 50, nbGeneration = 3, vNbSample = c(3, 6), vpDiff = c(0.9),
  vpDiffsd = c(0.1), vDiff = c(0.8), vInheritance = c(0.5),
  rateDiff = 0.01, minRate = 0.01, propInherite = 0.3, propHetero = 0.5,
  keepDiff = FALSE, outputDir = NULL, fileID = "s", minReads = 10,
  maxPercReads = 99.9, meanCov = 80, context = "CpG",
  assembly = "Rnor_5.0", saveGRanges = TRUE, saveMethylKit = TRUE,
  runAnalysis = FALSE, nbCores = 1, vSeed = -1)
```

Arguments

methData	an object of class methylBase, the CpG information from controls (CTRL) that will be used to create the synthetic chromosome. The methData object can also contain information from cases but only the controls are used.
nbSynCHR	a positive integer, the number of distinct synthetic chromosomes that will be generated. Default: 1.
nbSimulation	a positive integer, the number of simulations generated for each parameter (vNbSample, vpDiff, vDiff and vInheritance). The total number of simulation is nbSimulation * length(vNbSample) * length(vpDiff) * length(vInheritance)) Default: 10.
nbBlock	a positive integer, the number of blocks used for sampling. Default: 100.
nbCpG	a positive integer, the number of consecutive CpG positions used for sampling from methInfo. Default: 50.
nbGeneration	a positive integer, the number of generations simulated. Default: 3.

vNbSample — a vector of distinct positive integer, the number of controls (CTRL) and cases in the simulated dataset. In the simulated dataset, the number of CTRL equals the number of cases. The number of CTRL do not need to be equal to the number of Case in the real methData dataset. Default: c(3, 6).

vpDiff — a vector of distinct double superior to 0 and inferior or equal to 1, the mean value for the proportion of samples that will have, for a specific position, differentially methylated values. It can be interpreted as the penetrance. Note that vpDiff and vpDiffsd must be the same length. Default: c(0.9).

vpDiffsd — a vector of a non-negative double, the standard deviation associated to the vpDiff. Note that vpDiff and vpDiffsd must be the same length. Default: c(0.1).

vDiff — a vector of distinct non-negative double included in [0,1], the proportion of C/T for a case differentially methylated that follows a beta distribution where the mean is shifted by vDiff from the CTRL distribution. Default: c(0.8).

vInheritance — a vector of distinct non-negative double included in [0,1], the proportion of cases that inherits differentially methylated sites. Default: c(0.5).

rateDiff — a positive double inferior to 1, the mean of the chance that a site is differentially methylated. Default: 0.01.

minRate — a non-negative double inferior to 1, the minimum rate for differentially methylated sites. Default: 0.01.

propInherite — a non-negative double inferior or equal to 1, the proportion of differentially methylated regions that is inherited. Default: 0.3.

propHetero — a non-negative double between [0,1], the reduction of vDiff for the second and following generations. Default: 0.5.

keepDiff — a logical, when TRUE, the differentially methylated sites will be the same for all simulated datasets. Datasets generated using differents parameter values from vector parameters (vpDiff, vDiff and vInheritance) wil all have the same differentially methylated sites. Default: FALSE.

outputDir — a string of character or NULL, the path where the files created by the function will be saved. When NULL, the files are saved in a directory called "outputDir" that is located in the current directory. Default: NULL.

fileID — a string of character, a identifiant that will be included in each output file name. Each output file name is composed of those elements, separated by "_":

- a type name, ex: methylGR, methylObj, etc..
- a fileID
- the chromosome number, a number between 1 and nbSynCHR
- the number of samples, a number in the vNbSample vector
- the mean proportion of samples that has, for a specific position, differentially methylated values, a number in the vpDiff vector
- the proportion of C/T for a case differentially methylated that follows a shifted beta distribution, a number in the vDiff vector
- the proportion of cases that inherits differentially sites, a number in the vInheritance vector
- the identifiant for the simulation, a number between 1 and nbSimulation

- the file extension ".rds"

Default: "s".

minReads — a positive integer, sites and regions having lower coverage than this count are discarded. The parameter corresponds to the lo.count parameter in the methylKit package. Default: 10.

maxPercReads — a double between [0,100], the percentile of read counts that is going to be used as upper cutoff. Sites and regions having higher coverage than maxPercReads are discarded. This parameter is used for both CpG sites and tiles analysis. The parameter correspond to the hi.perc parameter in the methylKit package. Default: 99.9.

meanCov — a positive integer, the mean of the coverage at the simulated CpG sites. Default: 80.

context — a string of character, the short description of the methylation context, such as "CpG", "CpH", "CHH", etc.. Default: "CpG".

assembly — a string of character, the short description of the genome assembly, such as "mm9", "hg18", etc.. Default: "Rnor_5.0".

saveGRanges — a logical, when true, the package save two files type. The first generate for each simulation contains a list. The length of the list corresponds to the number of generation. The generation are stored in order (first entry = first generation, second entry = second generation, etc..). All samples related to one generations are contained in a GRangesList. The GRangeaList store a list of GRanges. Each GRanges stores the raw mehylation data of one sample. The second file a numeric vector denoting controls and cases (a file is generates by entry in the vector parameters vNbSample). Default: TRUE.

saveMethylKit — a logical, when TRUE, for each simulations save a file contains a list. The length of the list corresponds to the number of generation. The generation are stored in order (first entry = first generation, second entry = second generation, etc..). All samples related to one generations are contained in a S4 methylRawList object. The methylRawList object contains two Slots: 1. treatment: A numeric vector denoting controls and cases. 2. .Data: A list of methylRaw objects. Each object stores the raw methylation data of one sample. Default: TRUE.

runAnalysis — a logical, if TRUE, two files are saved for each simulation:

- 1. The first file is the methylObj... file formated with the methylkit package in a S4 methylBase object (using the methylKit functions: filterByCoverage, normalizeCoverage and unite).
- 2. The second file contains a S4 calculateDiffMeth object generated using the methylKit functions calculateDiffMeth on the first file.

Default: FALSE.

nbCores — a positive integer, the number of cores used when creating the simulated datasets. Default: 1 and always 1 for Windows.

vSeed — a integer, a seed used when reproducible results are needed. When a value inferior or equal to zero is given, a random integer is used. Default: -1.

Value

`0` indicating that the function have been successful.

Author(s)

Pascal Belleau, Astrid Deschenes

See Also

the vignette for detail description of the files created by the simulation.

Examples

```
## Load dataset containing methyl information
data(samplesForChrSynthetic)

## Set the output directory where files will be created
temp_dir <- "test_runSim"

## Create 2 simulated dataset (nbSimulation = 2)
## over 3 generations (nbGenration = 3) with
## 6 cases and 6 controls (nNbsample = 6) using only one set
## of parameters (vpDiff = 0.9, vpDiffsd = 0.1, vDiff = 0.8)
runSim(methData = samplesForChrSynthetic, nbSynCHR = 1, nbSimulation = 2,
    nbGeneration = 3, nbBlock = 10, nbCpG = 20, vNbSample = c(6),
    vpDiff = c(0.9), vpDiffsd = c(0.1), vDiff = c(0.8),
    vInheritance = c(0.5), rateDiff = 0.3, minRate = 0.2,
    propInherite = 0.3, propHetero = 0.5, outputDir = temp_dir,
    fileID = "F", nbCores = 1, vSeed = 32)

## Delete the output directory and its content
if (dir.exists(temp_dir)) {
    unlink(temp_dir, recursive = TRUE, force = FALSE)
}
```

---

samplesForChrSynthetic

All samples information, formated by `methylKit`*, in a* `methylBase` *format (for demo purpose).*

Description

The object is a `methylBase`. There is 12 samples (6 controls and 6 cases). Each sample information is stored in a `methylRaw` object.

Usage

```
data(samplesForChrSynthetic)
```

Format

A `methylBase` object contains the information for one generation. Each sample information is stored in a `methylRaw` object. There is `methylRaw` objects (6 controls and 6 cases).

Details

This dataset can be used to test the `runSim` function.

Value

A `methylBase` contains the information for one generation. Each sample information is stored in a `methylRaw` object. There is `methylRaw` objects (6 controls and 6 cases).

See Also

- `runSim` for running a simulation analysis using methylKit info as input

Examples

```
## Loading dataset
data(samplesForChrSynthetic)

## Set the output directory where files will be created
temp_dir <- "test_samplesForChrSynthetic"

## Create 4 simulated dataset (nbSimulation)
## over 3 generations (nbGenration = 3) with
## 6 cases and 6 controls (nNbsample = 6) using only one set
## of parameters (vpDiff = 0.85, vpDiffsd = 0.1, vDiff = 0.8)
runSim(outputDir = temp_dir, fileID = "F1", nbSynCHR = 1,
    methData = samplesForChrSynthetic, nbSimulation = 4,
    nbBlock = 10, nbCpG = 20,
    nbGeneration = 3, vNbSample = c(6), vpDiff = c(0.85),
    vpDiffsd = c(0.1), vDiff = c(0.8),
    vInheritance = c(0.5), propInherite = 0.3,
    rateDiff = 0.3, minRate = 0.2, propHetero = 0.5,
    nbCores = 1, vSeed = 32)

## Delete the output directory and its content
if (dir.exists(temp_dir)) {
    unlink(temp_dir, recursive = TRUE, force = FALSE)
}
```

Index