Package 'methyLImp2'

April 15, 2025

Title Missing value estimation of DNA methylation data

Version 1.3.1

Description This package allows to estimate missing values in DNA methylation data. methyLImp method is based on linear regression since methylation levels show a high degree of inter-sample correlation. Implementation is parallelised over chromosomes since probes on different chromosomes are usually independent. Mini-batch approach to reduce the runtime in case of large number of samples is available.

Imports BiocParallel, parallel, stats, methods, corpcor, SummarizedExperiment, utils

Depends R (>= 4.3.0), ChAMPdata

URL https://github.com/annaplaksienko/methyLImp2

BugReports https://github.com/annaplaksienko/methyLImp2/issues

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.1

biocViews DNAMethylation, Microarray, Software, MethylationArray, Regression

Suggests BiocStyle, knitr, rmarkdown, spelling, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Language en-US

git_url https://git.bioconductor.org/packages/methyLImp2

git_branch devel

git_last_commit 38134b5

git_last_commit_date 2025-02-04

Repository Bioconductor 3.21

Date/Publication 2025-04-14

Author Pietro Di Lena [aut] (ORCID: <https://orcid.org/0000-0002-1838-8918>),
Anna Plaksienko [aut, cre] (ORCID:
<https://orcid.org/0000-0001-9607-057X>),
Claudia Angelini [aut] (ORCID: <https://orcid.org/0000-0001-8350-8464>),
Christine Nardini [aut] (ORCID:
<https://orcid.org/0000-0001-7601-321X>)

Maintainer Anna Plaksienko <anna@plaxienko.com>

Contents

`beta` *A subset of GSE199057 dataset for vignette demonstration*

Description

The GSE199057 Gene Expression Omnibus dataset contains 68 mucosa samples from non-colon-cancer patients, from which we randomly sampled 24. Methylation data were measured on EPIC arrays and after removal of sex chromosomes and SNPs loci, it contains 816 126 probes. Pre-processing can be found on the _methyLImp2_ simulation github page https://github.com/annaplaksienko/methyLImp2_simu Here we subset only a quarter of probes from two smallest chromosomes (18 and 21) for the sake of demonstration.

Usage

```
data(beta)
```

Format

A numeric matrix

Value

A numeric matrix

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi

`beta_meta` *Metadata information for GSE199057 dataset for vignette demonstration*

Description

The GSE199057 Gene Expression Omnibus dataset contains 68 mucosa samples from non-colon-cancer patients, from which we randomly sampled 24. Dataset contains metadata for those.

Usage

```
data(beta_meta)
```

Format

A data.frame

Value

A data.frame.

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi

`custom_anno_example` *An example of how custom (user provided) annotation data frame should look like*

Description

A snippet from an annotation for 450K methylation dataset from ChAMPdata package. Only 5 CpGs are chosen simply to provide an example of the data frame organization.

Usage

```
data(custom_anno_example)
```

Format

A data.frame

Value

A data.frame.

`evaluatePerformance` *Evaluate performance metrics: root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE) and Pearson correlation coefficient (PCC).*

Description

This function computes performance metrics as an element-wise difference between two matrices (skipping NA elements that were not imputed): * $RMSE = \sqrt{\sum_i (true_i - est_i)^2 / \#NAs}$; * $MAE = \sum_i |true_i - est_i| / \#NAs$; * $MAPE = \frac{100}{n} \sum_i |true_i - est_i / true_i|$ (here we omit the true beta-values equal to 0 and their predicted values to avoid an indeterminate measure); * $PCC = \frac{\sum_i (true_i - \bar{true_i}) \sum_i (est_i - \bar{est_i})}{\sqrt{\sum_i (true_i - \bar{true_i})^2}} \sqrt{\sum_i (est_i - \bar{est_i})^2}$.

Usage

```
evaluatePerformance(beta_true, beta_imputed, na_positions)
```

Arguments

`beta_true`	first numeric data matrix.
`beta_imputed`	second numeric data matrix
`na_positions`	a list where each element is a list of two elements: column id and ids of rows with NAs in that column (structure matches the output of generateMissingData function). We need this because some NAs in the dataset are from real data and not artificial, so we can't evaluate the performance of the method on them since we do not know real value. Therefore, we need to know the positions of artificial NAs.

Value

A numerical vector of four numbers: root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE) and Pearson correlation coefficient (PCC).

Examples

```
data(beta)
with_missing_data <- generateMissingData(beta, lambda = 3.5)
beta_with_nas <- with_missing_data$beta_with_nas
na_positions <- with_missing_data$na_positions
beta_imputed <- methyLImp2(input = beta_with_nas, type = "EPIC",
                           minibatch_frac = 0.5,
                           BPPARAM = BiocParallel::SnowParam(workers = 1))
evaluatePerformance(beta, beta_imputed, na_positions)
```

`extract_values` *Extract a vector of beta values of interest out of a matrix, given the positions of artificial NAs*

Description

Extract a vector of beta values of interest out of a matrix, given the positions of artificial NAs

Usage

```
extract_values(beta, na_positions)
```

Arguments

`beta`	numeric data matrix
`na_positions`	a list where each element is a list of two elements: column id and ids of rows with NAs in that column (structure matches the output of generateMissingData function). We need this because some NAs in the dataset are from real data and not artificial, so we can't evaluate the performance of the method on them since we do not know real value. Therefore, we need to know the positions of artificial NAs.

Value

A numerical vector of beta values

`generateMissingData` *Generation of artificial missing values*

Description

This function generates missing values for the simulation purposes (to apply *methyLImp* method and then compare the imputed values with the true ones that have been replaced by NAs). First, we randomly choose 3% of all probes. Then for each of the chosen probes, we randomly define the number of NAs from a Poisson distribution with λ, appropriate to the sample size of the dataset (unless specified by the user, here we use $\lambda = 0.15 * \#samples + 0.2$). Finally, these amount of NAs is randomly placed among the samples.

Usage

```
generateMissingData(beta, lambda = NULL)
```

Arguments

`beta`	a numeric data matrix into which one wants to add some missing values
`lambda`	a number, parameter of the Poisson distribution that will indicate how many samples will have missing values in each selected probe.

Value

A list with two slots: a numeric data matrix with generated NAs in some entries and a list of positions of those NAs.

Examples

```
data(beta)
beta_with_nas <- generateMissingData(beta, lambda = 3.5)
```

`inv.plogit` *Inverse of the pseudo-logit function.*

Description

Inverse of the pseudo-logit function.

Usage

```
inv.plogit(X, min = 0, max = 1)
```

Arguments

`X`	a numeric matrix
`min`	a number, default value is 0
`max`	a number, default value is 1

Value

A numeric matrix

`methyLImp2` *Impute missing values in methylation dataset*

Description

This function performs missing value imputation specific for DNA methylation data. The method is based on linear regression since methylation levels show a high degree of inter-sample correlation. Implementation is parallelised over chromosomes to improve the running time.

Usage

```
methyLImp2(
  input,
  which_assay = NULL,
  type = c("450K", "EPIC", "user"),
  annotation = NULL,
  groups = NULL,
  range = NULL,
  skip_imputation_ids = NULL,
  BPPARAM = BiocParallel::bpparam(),
  minibatch_frac = 1,
  minibatch_reps = 1,
  overwrite_res = TRUE
)
```

Arguments

input	either a numeric data matrix with missing values to be, with named samples in rows and variables (probes) in named columns, or a SummarizedExperiment object, with an assay with variables in rows and samples in columns, as standard.
which_assay	a character specifying the name of assay of the SummarizedExperiment object to impute. By default the first one will be imputed.
type	a type of data, 450K or EPIC. Type is used to split CpGs across chromosomes. Match of CpGs to chromosomes is taken from ChAMPdata package. If you wish to provide your own match, specify "user" in this argument and provide a data frame in the next argument.
annotation	a data frame, user provided match between CpG sites and chromosomes. Must contain two columns: cpg and chr. Choose "user" in the previous argument to be able to provide user annotation.
groups	a vector of the same length as the number of samples that identifies what groups does each sample correspond, e.g. `c(1, 1, 2, 3)` or `c("group1", "group1", "group2", "group3")`. Unique elements of the vector will be identified as groups and data will be split accordingly. Imputation will be done for each group separately consecutively. The default is NULL, so all samples are considered as one group.
range	a vector of two numbers, min and max, specifying the range of values in the data. Since we assume the beta-value representation of the methylation data, the default range is $[0, 1]$. However, if a user wishes to apply the method to the other kind of data, they can change the range in this argument.
skip_imputation_ids	a numeric vector of ids of the columns with NAs for which *not* to perform the imputation. If `NULL`, all columns are considered.
BPPARAM	set of options for parallelization through BiocParallel package. For details we refer to their documentation. The one thing most users probably wish to customize is the number of cores. By default it is set to $\#cores - 2$. If you wish to change is, supply `BBPARAM = SnowParam(workers = ncores)` with your desired `ncores`. If the default or user-specified number of workers is higher than

number of chromosomes, it will be overwritten. We also recommend setting `exportglobals = FALSE` since it can help reduce running time.

`minibatch_frac` a number between 0 and 1, what fraction of samples to use for mini-batch computation. Remember that if your data has several groups, mini-batch will be applied to each group separately but with the same fraction, so choose it accordingly. However, if your chosen fraction will be smaller than a matrix dimension for some groups, mini-batch will be just ignored. We advise to use mini-batch only if you have large number of samples, order of hundreds. The default is 1 (i.e., 100% of samples are used, no mini-batch).

`minibatch_reps` a number, how many times to repeat computations with a fraction of samples specified above (more times -> better performance but more runtime). The default is 1 (as a companion to default fraction of 100%, i.e. no mini-batch).

`overwrite_res` a boolean specifying whether to overwrite an imputed slot of the SummarizedExperiment object or to add another slot with imputed data. The default is `TRUE` to reduced the object size.

Value

Either a numeric matrix with imputed data or a SummarizedExperiment object.

Examples

```
data(beta)
beta_with_nas <- generateMissingData(beta, lambda = 3.5)$beta_with_nas
beta_imputed <- methyLImp2(input = beta_with_nas, type = "EPIC",
                           minibatch_frac = 0.5,
                           BPPARAM = BiocParallel::SnowParam(workers = 1))
```

`methyLImp2_internal` *Impute missing values in methylation dataset*

Description

Impute missing values in methylation dataset

Usage

```
methyLImp2_internal(
  dat,
  min,
  max,
  skip_imputation_ids,
  minibatch_frac,
  minibatch_reps
)
```

Arguments

`dat`	a numeric data matrix with missing values, with samples in rows and variables (probes) in columns.
`min`	a number, minimum value for bounded-range variables. Default is 0 (we assume beta-value representation of the methylation data). Can be user provided in case of other types of data.
`max`	a number, maximum value for bounded-range variables. Default is 1 (we assume beta-value representation of the methylation data). Can be user provided in case of other types of data.
`skip_imputation_ids`	a numeric vector of ids of the columns with NAs for which *not* to perform the imputation. If `NULL`, all columns are considered.
`minibatch_frac`	a number, what percentage of samples to use for mini-batch computation. The default is 1 (i.e., 100% of samples are used, no mini-batch).
`minibatch_reps`	a number, how many times repeat computations with a fraction of samples (more times - better performance). The default is 1 (as a companion to default fraction of 100%. i.e. no mini-batch).

Value

A numeric matrix out with imputed data is returned.

`pinvr` *Computes the Moore-Penrose generalized inverse of a matrix. Allows rank reduction of the generalized inverse. This function is directly taken from MASS package (code on GPLv3 license) and modified in order to include the rank reduction option. The added code for rank reduction is commented in the implementation.*

Description

Computes the Moore-Penrose generalized inverse of a matrix. Allows rank reduction of the generalized inverse. This function is directly taken from MASS package (code on GPLv3 license) and modified in order to include the rank reduction option. The added code for rank reduction is commented in the implementation.

Usage

```
pinvr(X, max.sv = min(dim(X)), tol = sqrt(.Machine$double.eps))
```

Arguments

`X`	a numeric matrix
`tol`	a number

Value

A numeric matrix

---

plogit *Pseudo-logit function*

Description

Pseudo-logit function

Usage

```
plogit(X, min = 0, max = 1)
```

Arguments

X	a numeric matrix
min	a number, default value is 0
max	a number, default value is 1

Value

A numeric matrix

---

split_by_chromosomes *Split methylation dataset into a list by chromosomes*

Description

This function split a given methylation dataset into a list of datasets according to a given annotation.

Usage

```
split_by_chromosomes(data, type = c("450K", "EPIC", "user"), annotation = NULL)
```

Arguments

data	a numeric data matrix with with samples in rows and variables (probes) in named columns.
type	a type of data, 450K or EPIC. Type is used to split CpGs across chromosomes. Match of CpGs to chromosomes is taken from ChAMPdata package. If you wish to provide your own match, specify "user" in this argument and provide a data frame in the next argument.
annotation	a data frame, user provided match between CpG sites and chromosomes. Must contain two columns: cpg and chr. Choose "user" in the previous argument to be able to provide user annotation.

Value

A list of numeric data matrices where each matrix contains probes from one chromosome.

Index