Package ‘hahmmr’

May 8, 2026

Title Haplotype-Aware Hidden Markov Model for RNA

Version 1.0.0

Description Haplotype-aware Hidden Markov Model for RNA (HaHMMR) is a method for detecting copy number variations (CNVs) from bulk RNA-seq data. Additional examples, documentations, and details on the method are available at <`https://github.com/kharchenkolab/hahmmr/`>.

Depends R (>= 4.1.0)

Imports data.table, dplyr, GenomicRanges, ggplot2, glue, IRanges, methods, patchwork, Rcpp, stringr, tibble, zoo

Suggests ggrastr, testthat

LinkingTo Rcpp, RcppArmadillo, roptim

NeedsCompilation yes

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

LazyData true

Author Teng Gao [aut, cre] (ORCID: <`https://orcid.org/0000-0002-0196-689X`>),
Evan Biederstedt [aut],
Peter Kharchenko [aut]

Maintainer Teng Gao <`tgaoteng@gmail.com`>

Repository CRAN

Date/Publication 2023-10-25 18:00:10 UTC

Contents

`acen_hg19` *centromere regions (hg19)*

Description

centromere regions (hg19)

Usage

```
acen_hg19
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 22 rows and 3 columns.

`acen_hg38` *centromere regions (hg38)*

Description

centromere regions (hg38)

Usage

```
acen_hg38
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 22 rows and 3 columns.

`analyze_allele` *Analyze allele profile*

Description

Analyze allele profile

Usage

```
analyze_allele(
  bulk,
  t = 1e-05,
  theta_min = 0.08,
  gamma = 20,
  nu = 0.5,
  r = 0.015,
  hmm = "S5",
  fit_theta = FALSE,
  fit_gamma = FALSE,
  theta_start = 0.05,
  verbose = TRUE
)
```

Arguments

`bulk`	dataframe Bulk allele profile
`t`	numeric Transition probability
`theta_min`	numeric Minimum allele fraction
`gamma`	numeric Overdispersion parameter

`nu`	numeric Phase switch rate
`r`	numeric Alternative allele count bias
`hmm`	character HMM model to use (S3 or S5)
`fit_theta`	logical Whether to fit theta_min
`fit_gamma`	logical Whether to fit gamma
`theta_start`	numeric Starting value for theta_min
`verbose`	logical Whether to print progress

Value

dataframe Bulk allele profile with CNV states

Examples

```
bulk_example = analyze_allele(bulk_example, hmm = 'S5')
```

`analyze_joint` *Analyze allele and expression profile*

Description

Analyze allele and expression profile

Usage

```
analyze_joint(
  bulk,
  t = 1e-05,
  gamma = 20,
  theta_min = 0.08,
  logphi_min = 0.25,
  hmm = "S15",
  nu = 1,
  min_genes = 10,
  r = 0.015,
  theta_start = 0.05,
  exclude_neu = TRUE,
  fit_gamma = FALSE,
  fit_theta = FALSE,
  verbose = TRUE
)
```

Arguments

`bulk`	dataframe Bulk allele and expression profile
`t`	numeric Transition probability
`gamma`	numeric Overdispersion parameter
`theta_min`	numeric Minimum allele fraction
`logphi_min`	numeric Minimum log2 fold change
`hmm`	character HMM model to use (S7 or S15)
`nu`	numeric Phase switch rate
`min_genes`	integer Minimum number of genes per segment
`r`	numeric Alternative allele count bias
`theta_start`	numeric Starting value for theta_min
`exclude_neu`	logical Whether to exclude neutral segments in retest
`fit_gamma`	logical Whether to fit gamma
`fit_theta`	logical Whether to fit theta_min
`verbose`	logical Whether to print progress

Value

dataframe Bulk allele and expression profile with CNV states

Examples

```
bulk_example = analyze_joint(bulk_example, hmm = 'S15')
```

`bulk_example` *example pseudobulk dataframe*

Description

example pseudobulk dataframe

Usage

```
bulk_example
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 10321 rows and 58 columns.

`chrom_sizes_hg19` *chromosome sizes (hg19)*

Description

chromosome sizes (hg19)

Usage

```
chrom_sizes_hg19
```

Format

An object of class `data.table` (inherits from `data.frame`) with 22 rows and 2 columns.

`chrom_sizes_hg38` *chromosome sizes (hg38)*

Description

chromosome sizes (hg38)

Usage

```
chrom_sizes_hg38
```

Format

An object of class `data.table` (inherits from `data.frame`) with 22 rows and 2 columns.

`dbbinom` *Beta-binomial distribution density function A distribution is beta-binomial if p, the probability of success, in a binomial distribution has a beta distribution with shape parameters alpha > 0 and beta > 0 For more details, see extraDistr::dbbinom*

Description

Beta-binomial distribution density function A distribution is beta-binomial if p, the probability of success, in a binomial distribution has a beta distribution with shape parameters alpha > 0 and beta > 0 For more details, see extraDistr::dbbinom

Usage

```
dbbinom(x, size, alpha = 1, beta = 1, log = FALSE)
```

Arguments

`x`	vector of quantiles
`size`	number of trials (zero or more)
`alpha`	numeric (default=1)
`beta`	numeric (default=1)
`log`	boolean (default=FALSE)

Value

numeric Probability density values

Examples

```
dbbinom(1, 1, 1, 1)
```

`df_allele_example` *example allele count dataframe*

Description

example allele count dataframe

Usage

```
df_allele_example
```

Format

An object of class `data.table` (inherits from `data.frame`) with 9957 rows and 11 columns.

`dpoilog` *Returns the density for the Poisson lognormal distribution with parameters mu and sig*

Description

Returns the density for the Poisson lognormal distribution with parameters mu and sig

Usage

```
dpoilog(x, mu, sig, log = FALSE)
```

Arguments

`x`	vector of integers, the observations
`mu`	mean of lognormal distribution
`sig`	standard deviation of lognormal distribution
`log`	boolean Return the log density if TRUE (default=FALSE)

Value

numeric Probability density values

Examples

```
p = dpoilog(1, 1, 1)
```

`fit_lnpois_cpp` *Fit MLE of log-normal Poisson model*

Description

Fit MLE of log-normal Poisson model

Usage

```
fit_lnpois_cpp(Y_obs, lambda_ref, d)
```

Arguments

`Y_obs`	Vector of observed counts
`lambda_ref`	Vector of reference rates
`d`	integer Total depth

Value

NumericVector MLE estimates of mu and sigma

`forward_back_allele` *Forward-backward algorithm for allele HMM*

Description

Forward-backward algorithm for allele HMM

Usage

```
forward_back_allele(hmm)
```

Arguments

`hmm`	HMM object; expect variables x (allele depth), d (total depth), logPi (log transition prob matrix), delta (prior for each state), alpha (alpha for each state), beta (beta for each state), states (states), p_s (phase switch probs)

Value

numeric matrix; posterior probabilities

Examples

```
forward_back_allele(pre_likelihood_hmm)
```

`gaps_hg19` *genome gap regions (hg19)*

Description

genome gap regions (hg19)

Usage

```
gaps_hg19
```

Format

An object of class `data.table` (inherits from `data.frame`) with 28 rows and 3 columns.

`gaps_hg38` *genome gap regions (hg38)*

Description

genome gap regions (hg38)

Usage

```
gaps_hg38
```

Format

An object of class `data.table` (inherits from `data.frame`) with 30 rows and 3 columns.

`gene_counts_example` *example gene expression counts matrix*

Description

example gene expression counts matrix

Usage

```
gene_counts_example
```

Format

An object of class `matrix` (inherits from `array`) with 1758 rows and 1 columns.

`get_allele_bulk` *Aggregate into pseudobulk alelle profile*

Description

Aggregate into pseudobulk alelle profile

Usage

```
get_allele_bulk(df_allele, gtf, genetic_map = NULL, nu = 0.5, min_depth = 0)
```

Arguments

`df_allele`	dataframe Single-cell allele counts
`gtf`	dataframe Transcript gtf
`genetic_map`	dataframe Genetic map
`nu`	numeric Phase switch rate
`min_depth`	integer Minimum coverage to filter SNPs

Value

dataframe Pseudobulk allele profile

Examples

```
bulk_example = get_allele_bulk(
    df_allele = df_allele_example,
    gtf = gtf_hg38)
```

`get_bulk` *Produce combined bulk expression and allele profile*

Description

Produce combined bulk expression and allele profile

Usage

```
get_bulk(
  count_mat,
  lambdas_ref,
  df_allele,
  gtf,
  genetic_map = NULL,
  min_depth = 0,
  nu = 1,
  verbose = TRUE
)
```

Arguments

`count_mat`	matrix Gene expression counts
`lambdas_ref`	matrix Reference expression profiles
`df_allele`	dataframe Allele counts
`gtf`	dataframe Transcript gtf
`genetic_map`	dataframe Genetic map
`min_depth`	integer Minimum coverage to filter SNPs
`nu`	numeric Phase switch rate
`verbose`	logical Whether to print progress

Value

dataframe Pseudobulk gene expression and allele profile

Examples

```
bulk_example = get_bulk(
    count_mat = gene_counts_example,
    lambdas_ref = ref_hca,
    df_allele = df_allele_example,
    gtf = gtf_hg38)
```

`gtf_hg19` *gene model (hg19)*

Description

gene model (hg19)

Usage

```
gtf_hg19
```

Format

An object of class `data.table` (inherits from `data.frame`) with 26841 rows and 5 columns.

`gtf_hg38` *gene model (hg38)*

Description

gene model (hg38)

Usage

```
gtf_hg38
```

Format

An object of class `data.table` (inherits from `data.frame`) with 26807 rows and 5 columns.

`gtf_mm10` *gene model (mm10)*

Description

gene model (mm10)

Usage

```
gtf_mm10
```

Format

An object of class `data.table` (inherits from `data.frame`) with 30336 rows and 5 columns.

`likelihood_allele` *Only compute total log likelihood from an allele HMM*

Description

Only compute total log likelihood from an allele HMM

Usage

```
likelihood_allele(hmm)
```

Arguments

`hmm`	HMM object; expect variables x (allele depth), d (total depth), logPi (log transition prob matrix), delta (prior for each state), alpha (alpha for each state), beta (beta for each state), states (states), p_s (phase switch probs)

Value

numeric; total log likelihood

Examples

```
likelihood_allele(pre_likelihood_hmm)
```

`logSumExp` *logSumExp function*

Description

logSumExp function

Usage

```
logSumExp(x)
```

Arguments

`x`	NumericVector

Value

double logSumExp of x

`l_bbinom` *calculate joint likelihood of allele data*

Description

calculate joint likelihood of allele data

Usage

```
l_bbinom(AD, DP, alpha, beta)
```

Arguments

`AD`	numeric vector Variant allele depth
`DP`	numeric vector Total allele depth
`alpha`	numeric Alpha parameter of Beta-Binomial distribution
`beta`	numeric Beta parameter of Beta-Binomial distribution

Value

numeric Joint log likelihood

Examples

```
l_bbinom(c(1, 2), c(1, 2), 1, 1)
```

`l_lnpois` *calculate joint likelihood of a PLN model*

Description

calculate joint likelihood of a PLN model

Usage

```
l_lnpois(Y_obs, lambda_ref, d, mu, sig, phi = 1)
```

Arguments

`Y_obs`	numeric vector Gene expression counts
`lambda_ref`	numeric vector Reference expression levels
`d`	numeric Total library size
`mu`	numeric Global mean expression
`sig`	numeric Global standard deviation of expression
`phi`	numeric Fold change of expression

Value

numeric Joint log likelihood

Examples

```
l_lnpois(c(1, 2), c(1, 2), 1, 1, 1)
```

`plot_bulks` *Plot a group of pseudobulk HMM profiles*

Description

Plot a group of pseudobulk HMM profiles

Usage

```
plot_bulks(bulks, ..., ncol = 1, title = TRUE, title_size = 8)
```

Arguments

`bulks`	dataframe Pseudobulk profiles annotated with "sample" column
`...`	additional parameters passed to plot_psbulk()
`ncol`	integer Number of columns
`title`	logical Whether to add titles to individual plots
`title_size`	numeric Size of titles

Value

a ggplot object

Examples

```
p = plot_bulks(bulk_example)
```

`plot_psbulk` *Plot a pseudobulk HMM profile*

Description

Plot a pseudobulk HMM profile

Usage

```
plot_psbulk(
  bulk,
  use_pos = TRUE,
  allele_only = FALSE,
  min_LLR = 5,
  min_depth = 8,
  exp_limit = 2,
  phi_mle = TRUE,
  theta_roll = FALSE,
  dot_size = 0.8,
  dot_alpha = 0.5,
  legend = TRUE,
  exclude_gap = TRUE,
  genome = "hg38",
  text_size = 10,
  raster = FALSE
)
```

Arguments

`bulk`	dataframe Pseudobulk profile
`use_pos`	logical Use marker position instead of index as x coordinate
`allele_only`	logical Only plot alleles
`min_LLR`	numeric LLR threshold for event filtering
`min_depth`	numeric Minimum coverage depth for a SNP to be plotted
`exp_limit`	numeric Expression logFC axis limit
`phi_mle`	logical Whether to plot estimates of segmental expression fold change
`theta_roll`	logical Whether to plot rolling estimates of allele imbalance

`dot_size`	numeric Size of marker dots
`dot_alpha`	numeric Transparency of the marker dots
`legend`	logical Whether to show legend
`exclude_gap`	logical Whether to mark gap regions and centromeres
`genome`	character Genome build, either 'hg38' or 'hg19'
`text_size`	numeric Size of text in the plot
`raster`	logical Whether to raster images

Value

ggplot Plot of pseudobulk HMM profile

Examples

```
p = plot_psbulk(bulk_example)
```

`pre_likelihood_hmm` *HMM object for unit tests*

Description

HMM object for unit tests

Usage

```
pre_likelihood_hmm
```

Format

An object of class `list` of length 10.

`ref_hca` *reference expression magnitudes from HCA*

Description

reference expression magnitudes from HCA

Usage

```
ref_hca
```

Format

An object of class `matrix` (inherits from `array`) with 24756 rows and 12 columns.

`ref_hca_counts` *reference expression counts from HCA*

Description

reference expression counts from HCA

Usage

```
ref_hca_counts
```

Format

An object of class `matrix` (inherits from `array`) with 24857 rows and 12 columns.

`run_allele_hmm_s5` *Run a 5-state allele-only HMM - two theta levels*

Description

Run a 5-state allele-only HMM - two theta levels

Usage

```
run_allele_hmm_s5(
  pAD,
  DP,
  p_s,
  t = 1e-05,
  theta_min = 0.08,
  gamma = 20,
  prior = NULL,
  ...
)
```

Arguments

`pAD`	integer vector Paternal allele counts
`DP`	integer vector Total alelle counts
`p_s`	numeric vector Phase switch probabilities
`t`	numeric Transition probability between copy number states
`theta_min`	numeric Minimum haplotype frequency deviation threshold
`gamma`	numeric Overdispersion in the allele-specific expression
`prior`	numeric vector Prior probabilities for each state
`...`	Additional parameters

Value

character vector Decoded states

Examples

```
with(bulk_example, {
    run_allele_hmm_s5(pAD = pAD, DP = DP, R = R, p_s = p_s, theta_min = 0.08, gamma = 30)
})
```

`run_joint_hmm_s15` *Run 15-state joint HMM on a pseudobulk profile*

Description

Run 15-state joint HMM on a pseudobulk profile

Usage

```
run_joint_hmm_s15(
  pAD,
  DP,
  p_s,
  Y_obs = 0,
  lambda_ref = 0,
  d_total = 0,
  theta_min = 0.08,
  theta_neu = 0,
  bal_cnv = TRUE,
  phi_del = 2^(-0.25),
  phi_amp = 2^(0.25),
  phi_bamp = phi_amp,
  phi_bdel = phi_del,
  mu = 0,
  sig = 1,
  r = 0.015,
  t = 1e-05,
  gamma = 18,
  prior = NULL,
  exp_only = FALSE,
  allele_only = FALSE,
  classify_allele = FALSE,
  debug = FALSE,
  ...
)
```

Arguments

`pAD`	integer vector Paternal allele counts
`DP`	integer vector Total alelle counts
`p_s`	numeric vector Phase switch probabilities
`Y_obs`	numeric vector Observed gene counts
`lambda_ref`	numeric vector Reference expression rates
`d_total`	integer Total library size for expression counts
`theta_min`	numeric Minimum haplotype imbalance threshold
`theta_neu`	numeric Haplotype imbalance threshold for neutral state
`bal_cnv`	logical Whether to include balanced CNV states
`phi_del`	numeric Expected fold change for deletion
`phi_amp`	numeric Expected fold change for amplification
`phi_bamp`	numeric Expected fold change for balanced amplification
`phi_bdel`	numeric Expected fold change for balanced deletion
`mu`	numeric Global expression bias
`sig`	numeric Global expression variance
`r`	numeric Variant mapping bias
`t`	numeric Transition probability between copy number states
`gamma`	numeric Overdispersion in the allele-specific expression
`prior`	numeric vector Prior probabilities for each state
`exp_only`	logical Whether to only use expression data
`allele_only`	logical Whether to only use allele data
`classify_allele`	logical Whether to classify allele states
`debug`	logical Whether to print debug messages
`...`	Additional parameters

Value

character vector Decoded states

Examples

```
with(bulk_example, {
   run_joint_hmm_s15(pAD = pAD, DP = DP, p_s = p_s, Y_obs = Y_obs, lambda_ref = lambda_ref,
   d_total = na.omit(unique(d_obs)), mu = mu, sig = sig, t = 1e-5, gamma = 30, theta_min = 0.08)
})
```

`segs_example` *example CNV segments dataframe*

Description

example CNV segments dataframe

Usage

```
segs_example
```

Format

An object of class `data.table` (inherits from `data.frame`) with 27 rows and 30 columns.

`vcf_meta` *example VCF header*

Description

example VCF header

Usage

```
vcf_meta
```

Format

An object of class `character` of length 65.

Index