Package ‘hibayes’

May 25, 2022

Title Individual-Level, Summary-Level and Single-Step Bayesian Regression Model

Version 1.1.0

Date 2022-05-23

Description A user-friendly tool to fit Bayesian regression models. It can fit 3 types of Bayesian models using individual-level, summary-level, and individual plus pedigree-level (single-step) data for both Genomic prediction/selection (GS) and Genome-Wide Association Study (GWAS), it was designed to estimate joint effects and genetic parameters for a complex trait, including:

(1) fixed effects and coefficients of covariates,
(2) environmental random effects, and its corresponding variance,
(3) genetic variance,
(4) residual variance,
(5) heritability,
(6) genomic estimated breeding values (GEBV) for both genotyped and non-genotyped individuals,
(7) SNP effect size,
(8) phenotype/genetic variance explained (PVE) for single or multiple SNPs,
(9) posterior probability of association of the genomic window (WPPA),
(10) posterior inclusive probability (PIP).

The functions are not limited, we will keep on going in enriching it with more features.


License GPL-3

Maintainer Lilin Yin <ylilin@163.com>

URL https://github.com/YinLiLin/hibayes

BugReports https://github.com/YinLiLin/hibayes/issues

Encoding UTF-8

Imports utils, stats, methods, Rcpp
Bayes model

Description

Bayes linear regression model using individual level data

\[ y = X\beta + Rr + M\alpha + e \]

where \( \beta \) is a vector of estimated coefficient for covariates, and \( r \) is a vector of environmental random effects. \( M \) is a matrix of genotype covariate, \( \alpha \) is a vector of estimated marker effect size. \( e \) is a vector of residuals.

Usage

bayes(
  y,
  M,
  X = NULL,
  R = NULL,
  model = c("BayesCpi", "BayesA", "BayesL", "BSLMM", "BayesR", "BayesB", "BayesC", 
            "BayesBpi", "BayesRR"),
  map = NULL,
)
```r
Pi = NULL,
fold = NULL,
niter = 20000,
nburn = 12000,
windsize = NULL,
windnum = NULL,
vg = NULL,
dfvg = NULL,
s2vg = NULL,
ve = NULL,
dfve = NULL,
s2ve = NULL,
lambda = 0,
outfreq = NULL,
seed = 666666,
threads = 4,
verbose = TRUE
)
```

**Arguments**

- `y`: vector of phenotype, use 'NA' for the missings. The number and order of individuals of `y`, `M`, `X`, `R` should be exactly the same.
- `M`: numeric matrix of genotype with individuals in rows and markers in columns, NAs are not allowed.
- `X`: (optional) covariate matrix of all individuals, all values should be in digits, characters are not allowed, please use `model.matrix.lm` function to prepare it.
- `R`: (optional) environmental random effects matrix of all individuals, NAs are not allowed for the individuals with phenotypic value.
  - "BayesRR": Bayes Ridge Regression, all SNPs have non-zero effects and share the same variance, equals to RRBLUP or GBLUP.
  - "BayesA": all SNPs have non-zero effects, and take different variance which follows an inverse chi-square distribution.
  - "BayesB": only a small proportion of SNPs (1-Pi) have non-zero effects, and take different variance which follows an inverse chi-square distribution.
  - "BayesBpi": the same with "BayesB", but 'Pi' is not fixed.
  - "BayesC": only a small proportion of SNPs (1-Pi) have non-zero effects, and share the same variance.
  - "BayesCpi": the same with "BayesC", but 'Pi' is not fixed.
  - "BayesL": BayesLASSO, all SNPs have non-zero effects, and take different variance which follows an exponential distribution.
  - "BSLMM": all SNPs have non-zero effects, and take the same variance, but a small proportion of SNPs have additional shared variance.
  - "BayesR": only a small proportion of SNPs have non-zero effects, and the SNPs are allocated into different groups, each group has the same variance.
map (optional, only for GWAS) the map information of genotype, at least 3 columns are: SNPs, chromosome, physical position.

Pi vector, the proportion of zero effect and non-zero effect SNPs, the first value must be the proportion of non-effect markers.

fold proportion of variance explained for groups of SNPs, the default is c(0, 0.0001, 0.001, 0.01).

niter the number of MCMC iteration.

nburn the number of iterations to be discarded.

windsize window size in bp for GWAS, the default is NULL.

windnum fixed number of SNPs in a window for GWAS, if it is specified, 'windsize' will be invalid, the default is NULL.

vg prior value of genetic variance.

dfvg the number of degrees of freedom for the distribution of genetic variance.

s2vg scale parameter for the distribution of genetic variance.

ve prior value of residual variance.

dfve the number of degrees of freedom for the distribution of residual variance.

s2ve scale parameter for the distribution of residual variance.

lambda value of ridge regression for inverting a matrix.

outfreq frequency of collecting the estimated parameters and printing on console. Note that smaller frequency may have higher accuracy of estimated parameters, but would result in more time and memory for collecting process, on contrary, bigger frequency may have an negative effect on accuracy of estimations.

seed seed for random sample.

threads number of threads used for OpenMP.

verbose whether to print the iteration information on console.

Value

the function returns a list containing

$mu the regression intercept

$pi estimated proportion of zero effect and non-zero effect SNPs

$beta estimated coefficients for all covariates

$sr estimated environmental random effects

$vr estimated variance for all environmental random effect

$vg estimated genetic variance

$ve estimated residual variance

$h2 estimated heritability (h2 = Vg / (Vr + Vg + Ve))

$alpha estimated effect size of all markers

$g genomic estimated breeding value

$e residuals of the model
$\textbf{pip}$ the frequency for markers to be included in the model during MCMC iteration, known as posterior inclusive probability (PIP)

$\textbf{gwas}$ WPPA is defined to be the window posterior probability of association, it is estimated by counting the number of MCMC samples in which $
\alpha$
is nonzero for at least one SNP in the window

$\textbf{SMCMCsamples}$ the collected samples of posterior estimation for all the above parameters across MCMC iterations

**References**


**Examples**

```
# Load the example data attached in the package
pheno_file_path = system.file("extdata", "pheno.txt", package = "hibayes")
pheno = read.table(pheno_file_path, header=TRUE)
bfile_path = system.file("extdata", "geno", package = "hibayes")
data = read_plink(bfile_path, out=tempfile())
fam = data$fam
gen = data$geno
map = data$map

# Adjust the order of phenotype by genotype id
geno.id = fam[, 2]
pheno = pheno[match(geno.id, pheno[, 1]), ]

# Add fixed effects, covariates, and random effect
X <- model.matrix.lm(~as.numeric(scale)+as.factor(sex), data=pheno, na.action = "na.pass")
X <- X[, -1] #remove the intercept
# then fit the model as: fit = bayes(..., X=X, R=pheno[,c("group")], ...)

# For GS/GP
fit = bayes(y=pheno[, 2], M=geno, model="BayesR", niter=200, nburn=100)
```
# For GWAS
fit = bayes(y=pheno[, 2], M=geno, map=map, windsize=1e6, model="BayesCpi")

# The standard deviation of unknown parameters can be obtained from the list 'MCMCsamples':
# get the SD of estimated SNP effects for markers
snp_effect_sd = apply(fit$MCMCsamples$alpha, 1, sd)
# get the prediction error variance (PEV) of estimated breeding values
gebv_pev = apply(fit$MCMCsamples$g, 1, var)

### ldmat

**LD variance-covariance matrix calculation**

**Description**
To calculate density or sparse LD variance-covariance matrix with genotype in bigmemory format.

**Usage**

```r
ldmat(
geno,
map = NULL,
gwas.geno = NULL,
gwas.map = NULL,
chisq = NULL,
ldchr = FALSE,
threads = 4,
verbose = TRUE
)
```

**Arguments**

- **geno**
  the reference genotype panel in bigmemory format.
- **map**
  the map information of reference genotype panel, columns are: SNPs, chromosome, physical position.
- **gwas.geno**
  (optional) the genotype of gwas samples which were used to generate the summary data.
- **gwas.map**
  (optional) the map information of the genotype of gwas samples, columns are: SNPs, chromosome, physical position.
- **chisq**
  chi-square value for generating sparse matrix, if n*r2 < chisq, it would be set to zero.
- **ldchr**
  logical, whether to calculate the LD between chromosomes.
- **threads**
  the number of threads used in computation.
- **verbose**
  whether to print the information.
Value

For full ld matrix, it returns a standard R matrix, for sparse matrix, it returns a ‘dgCMatrix’.

Examples

```r
bfile_path = system.file("extdata", "geno", package = "hibayes")
data = read_plink(bfile_path, out=tempfile())
genotype = data$geno
map = data$map

xx = ldmat(geno, threads=4)  # chromosome wide full ld matrix
xx = ldmat(geno, chisq=5, threads=4)  # chromosome wide sparse ld matrix
xx = ldmat(geno, map, ldchr=FALSE, threads=4)  # chromosome block ld matrix
xx = ldmat(geno, map, ldchr=FALSE, chisq=5, threads=4)  # chromosome block + sparse ld matrix
```

Description

To load plink binary data

Usage

```r
read_plink(
  bfile = "", 
  maxLine = 10000, 
  impute = TRUE, 
  mode = c("A", "D"), 
  out = NULL, 
  threads = 4
)
```

Arguments

- `bfile`: character, prefix of Plink binary format data.
- `maxLine`: number, set the number of lines to read at a time.
- `impute`: logical, whether to impute missing values in genotype by major alleles.
- `mode`: “A” or “D”, additive effect or dominant effect.
- `out`: character, path and prefix of output file.
- `threads`: number, the number of used threads for parallel process.
Value

hibayes will code the genotype A1A1 as 2, A1A2 as 1, and A2A2 as 0, where A1 is the first allele of each marker in *.bim file, therefore the estimated effect size is on A1 allele, users should pay attention to it when a process involves marker effect.

Examples

```r
bfile_path = system.file("extdata", "geno", package = "hibayes")
data = read_plink(bfile_path, out=tempfile(), mode="A")
  fam = data$fam
geno = data$geno
map = data$map
```

Description

Bayes linear regression model using summary level data

Usage

```r
sbayes(
  sumstat,
  ldm,
  model = c("BayesB", "BayesA", "BayesL", "BayesRR", "BayesBpi", "BayesC", "BayesCpi",
            "BayesR", "CG"),
  map = NULL,
  Pi = NULL,
  lambda = NULL,
  fold = NULL,
  niter = 20000,
  nburn = 12000,
  windsize = NULL,
  windnum = NULL,
  vg = NULL,
  dfvg = NULL,
  s2vg = NULL,
  ve = NULL,
  dfve = NULL,
  s2ve = NULL,
  outfreq = NULL,
  seed = 666666,
  threads = 4,
  verbose = TRUE
)
```
**Arguments**

**sumstat** matrix of summary data, details refer to https://cnsgenomics.com/software/gcta/#COJO.

**ldm** dense or sparse matrix, ld for reference panel (m * m, m is the number of SNPs). NOTE that the order of SNPs should be consistent with summary data.

**model** bayes model including: "BayesB", "BayesA", "BayesL", "BayesRR", "BayesBpi", "BayesC", "BayesCpi", "BayesR", "CG".

  - "BayesRR": Bayes Ridge Regression, all SNPs have non-zero effects and share the same variance, equals to RRBLUP or GBLUP.
  - "BayesA": all SNPs have non-zero effects, and take different variance which follows an inverse chi-square distribution.
  - "BayesB": only a small proportion of SNPs (1-Pi) have non-zero effects, and take different variance which follows an inverse chi-square distribution.
  - "BayesBpi": the same with "BayesB", but 'Pi' is not fixed.
  - "BayesC": only a small proportion of SNPs (1-Pi) have non-zero effects, and share the same variance.
  - "BayesCpi": the same with "BayesC", but 'Pi' is not fixed.
  - "BayesL": BayesLASSO, all SNPs have non-zero effects, and take different variance which follows an exponential distribution.
  - "BayesR": only a small proportion of SNPs have non-zero effects, and the SNPs are allocated into different groups, each group has the same variance.
  - "CG": conjugate gradient algorithm with assigned lambda.

**map** (optional, only for GWAS) the map information of genotype, at least 3 columns are: SNPs, chromosome, physical position.

**Pi** vector, the proportion of zero effect and non-zero effect SNPs, the first value must be the proportion of non-effect markers.

**lambda** value or vector, the ridge regression value for each SNPs.

**fold** percentage of variance explained for groups of SNPs, the default is c(0, 0.0001, 0.001, 0.01).

**niter** the number of MCMC iteration.

**nburn** the number of iterations to be discarded.

**windsize** window size in bp for GWAS, the default is 1e6.

**windnum** fixed number of SNPs in a window for GWAS, if it is specified, 'windsize' will be invalid, the default is NULL.

**vg** prior value of genetic variance.

**dfvg** the number of degrees of freedom for the distribution of genetic variance.

**s2vg** scale parameter for the distribution of genetic variance.

**ve** prior value of residual variance.

**dfve** the number of degrees of freedom for the distribution of residual variance.

**s2ve** scale parameter for the distribution of residual variance.

**outfreq** frequency of collecting the estimated parameters and printing on console. Note that smaller frequency may have higher accuracy of estimated parameters, but would result in more time and memory for collecting process, on contrary, bigger frequency may have an negative effect on accuracy of estimations.
seed for random sample.
threads number of threads used for OpenMP.
verbose whether to print the iteration information on console.

Value

the function returns a list containing

$\pi$ estimated proportion of zero effect and non-zero effect SNPs
$Vg$ estimated genetic variance
$Ve$ estimated residual variance
$h2$ estimated heritability ($h2 = \frac{Vg}{(Vg + Ve)}$)
$\alpha$ estimated effect size of all markers
$\alpha$ the frequency for markers to be included in the model during MCMC iteration, also known as posterior inclusive probability (PIP)
$\omega$ WPPA is defined to be the window posterior probability of association, it is estimated by counting the number of MCMC samples in which

is nonzero for at least one SNP in the window

$\alpha$ MCMC samples the collected samples of posterior estimation for all the above parameters across MCMC iterations

References


Examples

```r
bfile_path = system.file("extdata", "geno", package = "hibayes")
data = read_plink(bfile_path, out=tempfile())
genon = data$genon
mapon = data$map
headon(mapon)
sumstat_path = system.file("extdata", "geno.ma", package = "hibayes")
sumstat = read.table(sumstat_path, header=TRUE)
head(sumstat)

# compute ld variance covariance matrix
ldm = ldmat(genon, threads=4)  # chromosome wide full ld matrix

# if the order of SNPs in genotype is not consistent with the order in sumstat file,
# prior adjusting is necessary.
indx = match(mapon[, 1], sumstat[, 1])
sumstat = sumstat[indx, ]
```
# fit model
fit = sbayes(sumstat=sumstat, ldm=ldm1, model="BayesR")

# The standard deviation of unknown parameters can be obtained from the list 'MCMCsamples':
# get the SD of estimated SNP effects for markers
snp_effect_sd = apply(fit$MCMCsamples$alpha, 1, sd)

---

**ssbayes**

**Single-step Bayes model**

**Description**

Single-step Bayes linear regression model using individual level data and pedigree information

\[
y = X\beta + Rr + M\alpha + U\epsilon + e
\]

where \(y\) is the vector of phenotypic values for both genotyped and non-genotyped individuals, \(\beta\) is a vector of estimated coefficient for covariates, \(M\) contains the genotype \((M_2)\) for genotyped individuals and the imputed genotype \((M_1 = A_{12}A_{22}M_2)\) for non-genotyped individuals, \(\epsilon\) is the vector of genotype imputation error, \(e\) is a vector of residuals.

**Usage**

ssbayes(
  y,
  y.id,
  M,
  M.id,
  P,
  X = NULL,
  R = NULL,
  model = c("BayesCpi", "BayesA", "BayesL", "BayesR", "BayesB", "BayesC", "BayesBpi", "BayesRR"),
  map = NULL,
  Pi = NULL,
  fold = NULL,
  niter = 20000,
  nburn = 12000,
  windsize = NULL,
  windnum = NULL,
  maf = 0.01,
  vg = NULL,
  dfvg = NULL,
  s2vg = NULL,
  ve = NULL,
Arguments

- **y**: vector of phenotype, use 'NA' for the missings.
- **y.id**: vector of id for phenotype.
- **M**: numeric matrix of genotype with individuals in rows and markers in columns, NAs are not allowed.
- **M.id**: vector of id for genotype.
- **P**: matrix of pedigree, 3 columns limited, the order of columns should be "id", "sir", "dam".
- **X**: (optional) covariate matrix of all individuals, all values should be in digits, characters are not allowed, please use `model.matrix.lm` function to prepare it.
- **R**: (optional) environmental random effects matrix of all individuals, NAs are not allowed for the individuals with phenotypic value.
- **model**: bayes model including: "BayesB", "BayesA", "BayesL", "BayesRR", "BayesBpi", "BayesC", "BayesCpi", "BayesR", "BSLMM".
  - "BayesRR": Bayes Ridge Regression, all SNPs have non-zero effects and share the same variance, equals to RRBLUP or GBLUP.
  - "BayesA": all SNPs have non-zero effects, and take different variance which follows an inverse chi-square distribution.
  - "BayesB": only a small proportion of SNPs (1-Pi) have non-zero effects, and take different variance which follows an inverse chi-square distribution.
  - "BayesBpi": the same with "BayesB", but 'Pi' is not fixed.
  - "BayesC": only a small proportion of SNPs (1-Pi) have non-zero effects, and share the same variance.
  - "BayesCpi": the same with "BayesC", but 'Pi' is not fixed.
  - "BayesL": BayesLASSO, all SNPs have non-zero effects, and take different variance which follows an exponential distribution.
  - "BayesR": only a small proportion of SNPs have non-zero effects, and the SNPs are allocated into different groups, each group has the same variance.
- **map**: (optional, only for GWAS) the map information of genotype, at least 3 columns are: SNPs, chromosome, physical position.
- **Pi**: vector, the proportion of zero effect and non-zero effect SNPs, the first value must be the proportion of non-effect markers.
- **fold**: proportion of variance explained for groups of SNPs, the default is c(0, 0.0001, 0.001, 0.01).
- **niter**: the number of MCMC iteration.
**ssbayes**

- **nburn**  the number of iterations to be discarded.
- **windsize**  window size in bp for GWAS, the default is NULL.
- **windnum**  fixed number of SNPs in a window for GWAS, if it is specified, ’windsize’ will be invalid, the default is NULL.
- **maf**  the effects of markers whose MAF are lower than the threshold will be not estimated.
- **vg**  prior value of genetic variance.
- **dfvg**  the number of degrees of freedom for the distribution of genetic variance.
- **s2vg**  scale parameter for the distribution of genetic variance.
- **ve**  prior value of residual variance.
- **dfve**  the number of degrees of freedom for the distribution of residual variance.
- **s2ve**  scale parameter for the distribution of residual variance.
- **outfreq**  frequency of collecting the estimated parameters and printing on console. Note that smaller frequency may have higher accuracy of estimated parameters, but would result in more time and memory for collecting process, on contrary, bigger frequency may have an negative effect on accuracy of estimations.
- **seed**  seed for random sample.
- **threads**  number of threads used for OpenMP.
- **verbose**  whether to print the iteration information on console.

**Value**

the function returns a list containing

- **$J**  coefficient for genotype imputation residuals
- **$Ve**  estimated variance of genotype imputation residuals
- **$epsilon**  genotype imputation residuals
- **$mu**  the regression intercept
- **$pi**  estimated proportion of zero effect and non-zero effect SNPs
- **$beta**  estimated coefficients for all covariates
- **$r**  estimated environmental random effects
- **$Vr**  estimated variance for all environmental random effect
- **$Vg**  estimated genetic variance
- **$Ve**  estimated residual variance
- **$h2**  estimated heritability ($h2 = Vg / (Vr + Vg + Ve)$)
- **$g**  data.frame, the first column is the list of individual id, the second column is the genomic estimated breeding value for all individuals, including genotyped and non-genotyped.
- **$alpha**  estimated effect size of all markers
- **$e**  residuals of the model
- **$pip**  the frequency for markers to be included in the model during MCMC iteration, also known as posterior inclusive probability (PIP)
## gwass

WPPA is defined to be the window posterior probability of association, it is estimated by counting the number of MCMC samples in which

\[ \alpha \]

is nonzero for at least one SNP in the window

### MCMC samples

the collected samples of posterior estimation for all the above parameters across MCMC iterations

### References


### Examples

```r
# Load the example data attached in the package
pheno_file_path = system.file("extdata", "pheno.txt", package = "hibayes")
pheno = read.table(pheno_file_path, header=TRUE)
pedigree_file_path = system.file("extdata", "ped.txt", package = "hibayes")
ped = read.table(pedigree_file_path, header=TRUE)
bfile_path = system.file("extdata", "geno", package = "hibayes")
data = read_plink(bfile_path, out=tempfile())
fam = data$fam
geno = data$geno
map = data$map

# NOTE: for ssbayes model, there is no NEED to adjust the order of id in different files
genoid = fam[, 2]
pheno.id = pheno[, 1]

# Add fixed effects, covariates, and random effect
X <- model.matrix.lm(~as.numeric(scale)+as.factor(sex), data=pheno, na.action = "na.pass")
X <- X[, -1] #remove the intercept

# then fit the model as: fit = ssbayes(..., X=X, R=pheno[,c("group")], ...)

# For GS/GP
fit = ssbayes(y=pheno[, 2], y.id=pheno.id, M=geno, M.id=geno.id, P=ped,
model="BayesR", niter=200, nburn=100)

# For GWAS
fit = ssbayes(y=pheno[, 2], y.id=pheno.id, M=geno, M.id=geno.id, P=ped,
map=map, windsize=1e6, model="BayesCpi")

# The standard deviation of unknow parameters can be obtained from the list 'MCMCsamples':
# get the SD of estimated SNP effects for markers
snp_effect_sd = apply(fit$MCMCsamples$alpha, 1, sd)
# get the prediction error variance (PEV) of estimated breeding values
gebv_pev = apply(fit$MCMCsamples$g, 1, var)
```
Index

bayes, 2
ldmat, 6
read_plink, 7
sabayes, 8
ssbayes, 11