Package 'QTLEMM'

May 7, 2026

Type Package

Title QTL EM Algorithm Mapping and Hotspots Detection

Version 3.1.0

Description For QTL mapping, this package comprises several functions designed to execute diverse tasks, such as simulating or analyzing data, calculating significance thresholds, and visualizing QTL mapping results. The single-QTL or multiple-QTL method, which enables the fitting and comparison of various statistical models, is employed to analyze the data for estimating QTL parameters. The models encompass linear regression, permutation tests, normal mixture models, and truncated normal mixture models. The Gaussian stochastic process is utilized to compute significance thresholds for QTL detection on a genetic linkage map within experimental populations. Two types of data, complete genotyping, and selective genotyping data from various experimental populations, including backcross, F2, recombinant inbred (RI) populations, and advanced intercrossed (AI) populations, are considered in the QTL mapping analysis. For QTL hotspot detection, statistical methods can be developed based on either utilizing individual-level data or summarized data. We have proposed a statistical framework capable of handling both individual-level data and summarized QTL data for QTL hotspot detection. Our statistical framework can overcome the underestimation of thresholds resulting from ignoring the correlation structure among traits. Additionally, it can identify different types of hotspots with minimal computational cost during the detection process. Here, we endeavor to furnish the R codes for our QTL mapping and hotspot detection methods, intended for general use in genes, genomics, and genetics studies. The QTL mapping methods for the complete and selective genotyping designs are based on the multiple interval mapping (MIM) model proposed by Kao, C.-H. , Z.-B. Zeng and R. D. Teasdale (1999) <doi:10.1534/genetics.103.021642> and H.-I Lee, H.-A. Ho and C.-H. Kao (2014) <doi:10.1534/genetics.114.168385>, respectively. The QTL hotspot detection analysis is based on the method by Wu, P.-Y., M.-.H. Yang, and C.-H. Kao (2021) <doi:10.1093/g3journal/jkab056>.

Imports mvtnorm, utils, stats, graphics, grDevices, gtools

URL https://github.com/py-chung/QTLEMM

BugReports https://github.com/py-chung/QTLEMM/issues

License GPL-2

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Ping-Yuan Chung [cre],
Chen-Hung Kao [aut],
Y.-T. Guo [aut],
H.-N. Ho [aut],
H.-I. Lee [aut],
P.-Y. Wu [aut],
M.-H. Yang [aut],
M.-H. Zeng [aut]

Maintainer Ping-Yuan Chung <pychung@webmail.stat.sinica.edu.tw>

Repository CRAN

Date/Publication 2025-09-12 08:40:14 UTC

Contents

D.make 2
EM.MIM 4
EM.MIMv 8
EQF.permu 11
EQF.plot 12
IM.search 13
LOD.QTLdetect 16
LRTthre 18
MIM.points 19
MIM.search 23
progeny 26
Q.make 28
Qhot 29
Qhot.EQF 31

Index **33**

`D.make` *Generate D Matrix*

Description

Generate the genetic design matrix of specified QTL number and effects.

Usage

```
D.make(
  nQTL,
  type = "RI",
  a = TRUE,
  d = TRUE,
```

```
  aa = FALSE,
  dd = FALSE,
  ad = FALSE
)
```

Arguments

nQTL	integer. The number of QTLs.
type	character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".
a	integer or vector. A integer or vector to determines which additive effects of QTLs will be considered in this design matrix. If a=TRUE, the additive effect of all QTLs will be considered. If a=FALSE, no additive effect will be considered.
d	integer or vector. A integer or vector to determines which dominant effects of QTLs will be considered in this design matrix. If d=TRUE, the dominant effect of all QTLs will be considered.If d=FALSE, no dominant effect will be considered.
aa	vector or matrix. The additive-by-additive interaction. Two format can be used in this parameter. One format is a vector, where every two elements indicate a combination of additive-by-additive interaction. The other format is a 2*i matrix, where i is the number of combinations of interaction, and each column indicates the two interacting QTLs. Additionally, if aa=TRUE, all combinations of additive-by-additive interaction will be considered. If aa=FALSE, no additive-by-additive interaction will be considered.
dd	vector or matrix. The dominant-by-dominant interaction. The format is the same as that in aa.
ad	vector or matrix. The additive-by-dominant interaction. The format is the same as that in aa. Note that in each pair of QTLs, the first element indicates the additive effect, and the second element indicates the dominant effect.

Value

The genetic design matrix, where the elements represent the coded variables of the QTL effects. It is a g*p matrix, where g is the number of possible QTL genotypes, and p is the number of effects in the MIM model.

Note

For the 'type' parameter, if type="BC", the design matrix exclusively contains additive effects and additive-by-additive interactions. However, if type="AI" or type="RI", it encompasses additive and dominance effects along with all interactions.

For instance, when aa=c(1,3,2,4,5,6), it denotes that the interaction between QTL1 and QTL3, the interaction between QTL2 and QTL4, and that between QTL5 and QTL6 will be considered in the design matrix. Furthermore, the matrix format can be expressed as aa=matrix(c(1,3,2,4,5,6),2,3). Similarly, parameters DD and AD are also expressed in the same format.

References

KAO, C.-H. and Z.-B. ZENG 1997 General formulas for obtaining the maximum likelihood estimates and the asymptotic variance-covariance matrix in QTL mapping when using the EM algorithm. Biometrics 53, 653-665. <doi: 10.2307/2533965.>

KAO, C.-H., Z.-B. ZENG and R. D. TEASDALE 1999 Multiple interval mapping for Quantitative Trait Loci. Genetics 152: 1203-1216. <doi: 10.1093/genetics/152.3.1203>

Examples

```
D.make(4, d = c(1,3,4), aa = c(1,2,2,3), dd = c(1,3,1,4), ad = c(1,2,2,1,2,3,3,4))

aa <- matrix(c(1,2,3,4,4,5), 2, 3)
aa
D.make(5, type = "BC", a = c(1,3,4,5), aa = aa)
```

`EM.MIM` *EM Algorithm for QTL MIM*

Description

Expectation-maximization algorithm for QTL multiple interval mapping. It can handle genotype data which is selective genotyping too.

Usage

```
EM.MIM(
  QTL = NULL,
  marker = NULL,
  geno = NULL,
  D.matrix = NULL,
  cp.matrix = NULL,
  y = NULL,
  yu = NULL,
  sele.g = "n",
  tL = NULL,
  tR = NULL,
  type = "RI",
  ng = 2,
  cM = TRUE,
  E.vector0 = NULL,
  X = NULL,
  beta0 = NULL,
  variance0 = NULL,
  crit = 10^-5,
  stop = 1000,
  conv = TRUE,
  console = interactive(),
```

```
  IMresult = NULL,
  MIMresult = NULL
)
```

Arguments

QTL	matrix. A q*2 matrix contains the QTL information, where the row dimension 'q' represents the number of QTLs in the chromosomes. The first column labels the chromosomes where the QTLs are located, and the second column labels the positions of QTLs (in morgan (M) or centimorgan (cM)).
marker	matrix. A k*2 matrix contains the marker information, where the row dimension 'k' represents the number of markers in the chromosomes. The first column labels the chromosomes where the markers are located, and the second column labels the positions of markers (in morgan (M) or centimorgan (cM)). It's important to note that chromosomes and positions must be sorted in order.
geno	matrix. A n*k matrix contains the genotypes of k markers for n individuals. The marker genotypes of P1 homozygote (MM), heterozygote (Mm), and P2 homozygote (mm) are coded as 2, 1, and 0, respectively, with NA indicating missing values.
D.matrix	matrix. The design matrix of QTL effects is a g*p matrix, where g is the number of possible QTL genotypes, and p is the number of effects considered in the MIM model. This design matrix can be conveniently generated using the function D.make().
cp.matrix	matrix. The conditional probability matrix is an n*g matrix, where n is the number of genotyped individuals, and g is the number of possible genotypes of QTLs. If cp.matrix=NULL, the function will calculate the conditional probability matrix for selective genotyping.
y	vector. A vector that contains the phenotype values of individuals with genotypes.
yu	vector. A vector that contains the phenotype values of individuals without genotypes.
sele.g	character. Determines the type of data being analyzed: If sele.g="n", it considers the data as complete genotyping data. If sele.g="f", it treats the data as selective genotyping data and utilizes the proposed corrected frequency model (Lee 2014) for analysis; If sele.g="t", it considers the data as selective genotyping data and uses the truncated model (Lee 2014) for analysis; If sele.g="p", it treats the data as selective genotyping data and uses the population frequency-based model (Lee 2014) for analysis. Note that the 'yu' argument must be provided when sele.g="f" or "p".
tL	numeric. The lower truncation point of phenotype value when sele.g="t". When sele.g="t" and tL=NULL, the 'yu' argument must be provided. In this case, the function will consider the minimum of 'yu' as the lower truncation point.
tR	numeric. The upper truncation point of phenotype value when sele.g="t". When sele.g="t" and tR=NULL, the 'yu' argument must be provided. In this case, the function will consider the maximum of 'yu' as the upper truncation point.

type	character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".
ng	integer. The generation number of the population type. For instance, in a BC1 population where type="BC", ng=1; in an AI F3 population where type="AI", ng=3.
cM	logical. Specify the unit of marker position. If cM=TRUE, it denotes centimorgan; if cM=FALSE, it denotes morgan.
E.vector0	vector. The initial value for QTL effects. The number of elements corresponds to the column dimension of the design matrix. If E.vector0=NULL, the initial value for all effects will be set to 0.
X	matrix. The design matrix of the fixed factors except for QTL effects. It is an n*k matrix, where n is the number of individuals, and k is the number of fixed factors. If X=NULL, the matrix will be an n*1 matrix where all elements are 1.
beta0	vector. The initial value for effects of the fixed factors. The number of elements corresponds to the column dimension of the fixed factor design matrix. If beta0=NULL, the initial value will be set to the average of y.
variance0	numeric. The initial value for variance. If variance0=NULL, the initial value will be set to the variance of phenotype values.
crit	numeric. The convergence criterion of EM algorithm. The E and M steps will iterate until a convergence criterion is met. It must be a value between 0 and 1.
stop	numeric. The stopping criterion of EM algorithm. The E and M steps will halt when the iteration number reaches the stopping criterion, treating the algorithm as having failed to converge.
conv	logical. If set to False, it will disregard the failure to converge and output the last result obtained during the EM algorithm before reaching the stopping criterion.
console	logical. Determines whether the process of the algorithm will be displayed in the R console or not.
IMresult	list. The data list of the output from IM.search(). The required parameters for this function will be extracted from the data list.
MIMresult	list. The data list of the output from MIM.search() or MIM.points(). The required parameters for this function will be extracted from the data list.

Value

QTL	The QTL imformation of this analysis.
E.vector	The QTL effects are calculated by the EM algorithm.
beta	The effects of the fixed factors are calculated by the EM algorithm.
variance	The variance is calculated by the EM algorithm.
PI.matrix	The posterior probabilities matrix after the process of the EM algorithm.
log.likelihood	The log-likelihood value of this model.
LRT	The LRT statistic of this model.

R2	The coefficient of determination of this model. This can be used as an estimate of heritability.
y.hat	The fitted values of trait values with genotyping are calculated by the estimated values from the EM algorithm.
yu.hat	The fitted values of trait values without genotyping are calculated by the estimated values from the EM algorithm.
iteration.number	The iteration number of the EM algorithm.
model	The model of this analysis, contains complete a genotyping model, a proposed model, a truncated model, and a frequency-based model.

References

KAO, C.-H. and Z.-B. ZENG 1997 General formulas for obtaining the maximum likelihood estimates and the asymptotic variance-covariance matrix in QTL mapping when using the EM algorithm. Biometrics 53, 653-665. <doi: 10.2307/2533965.>

KAO, C.-H., Z.-B. ZENG and R. D. TEASDALE 1999 Multiple interval mapping for Quantitative Trait Loci. Genetics 152: 1203-1216. <doi: 10.1093/genetics/152.3.1203>

H.-I LEE, H.-A. HO and C.-H. KAO 2014 A new simple method for improving QTL mapping under selective genotyping. Genetics 198: 1685-1698. <doi: 10.1534/genetics.114.168385.>

See Also

D.make Q.make IM.search MIM.search MIM.points

Examples

```
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# run and result
D.matrix <- D.make(3, type = "RI", aa = c(1, 3, 2, 3), dd = c(1, 2, 1, 3), ad = c(1, 2, 2, 3))
cp.matrix <- Q.make(QTL, marker, geno, type = "RI", ng = 2)$cp.matrix
result <- EM.MIM(D.matrix = D.matrix, cp.matrix = cp.matrix, y = y)
result$E.vector

## Not run:
# Example for selective genotyping data
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# make the seletive genotyping data
ys <- y[y > quantile(y)[4] | y < quantile(y)[2]]
yu <- y[y >= quantile(y)[2] & y <= quantile(y)[4]]
geno.s <- geno[y > quantile(y)[4] | y < quantile(y)[2],]

# run and result
D.matrix <- D.make(3, type = "RI", aa = c(1, 3, 2, 3), dd = c(1, 2, 1, 3), ad = c(1, 2, 2, 3))
result <- EM.MIM(QTL, marker, geno.s, D.matrix, y = ys, yu = yu, sele.g = "f")
```

```
result$E.vector

## End(Not run)
```

EM.MIMv *EM Algorithm for QTL MIM with Asymptotic Variance-Covariance Matrix*

Description

Expectation-maximization algorithm for QTL multiple interval mapping. It can obtain the asymptotic variance-covariance matrix of the result from the EM algorithm and the approximate solution of variances of parameters.

Usage

```
EM.MIMv(
  QTL = NULL,
  marker = NULL,
  geno = NULL,
  D.matrix = NULL,
  cp.matrix = NULL,
  y = NULL,
  type = "RI",
  ng = 2,
  cM = TRUE,
  E.vector0 = NULL,
  X = NULL,
  beta0 = NULL,
  variance0 = NULL,
  crit = 10^-5,
  stop = 1000,
  conv = TRUE,
  var.pos = TRUE,
  console = interactive(),
  IMresult = NULL,
  MIMresult = NULL
)
```

Arguments

QTL	matrix. A q*2 matrix contains the QTL information, where the row dimension 'q' represents the number of QTLs in the chromosomes. The first column labels the chromosomes where the QTLs are located, and the second column labels the positions of QTLs (in morgan (M) or centimorgan (cM)).

`marker` matrix. A k*2 matrix contains the marker information, where the row dimension 'k' represents the number of markers in the chromosomes. The first column labels the chromosomes where the markers are located, and the second column labels the positions of markers (in morgan (M) or centimorgan (cM)). It's important to note that chromosomes and positions must be sorted in order.

`geno` matrix. A n*k matrix contains the genotypes of k markers for n individuals. The marker genotypes of P1 homozygote (MM), heterozygote (Mm), and P2 homozygote (mm) are coded as 2, 1, and 0, respectively, with NA indicating missing values.

`D.matrix` matrix. The design matrix of QTL effects is a g*p matrix, where g is the number of possible QTL genotypes, and p is the number of effects considered in the MIM model. The design matrix can be easily generated by the function D.make().

`cp.matrix` matrix. The conditional probability matrix is an n*g matrix, where n is the number of genotyped individuals, and g is the number of possible genotypes of QTLs. If cp.matrix=NULL, the function will calculate the conditional probability matrix, and markers whose positions are the same as QTLs will be skipped.

`y` vector. A vector with n elements that contain the phenotype values of individuals.

`type` character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".

`ng` integer. The generation number of the population type. For instance, in a BC1 population where type="BC", ng=1; in an AI F3 population where type="AI", ng=3.

`cM` logical. Specify the unit of marker position. If cM=TRUE, it denotes centimorgan; if cM=FALSE, it denotes morgan.

`E.vector0` vector. The initial value for QTL effects. The number of elements corresponds to the column dimension of the design matrix. If E.vector0=NULL, the initial value for all effects will be set to 0.

`X` matrix. The design matrix of the fixed factors except for QTL effects. It is an n*k matrix, where n is the number of individuals, and k is the number of fixed factors. If X=NULL, the matrix will be an n*1 matrix where all elements are 1.

`beta0` vector. The initial value for effects of the fixed factors. The number of elements corresponds to the column dimension of the fixed factor design matrix. If beta0=NULL, the initial value will be set to the average of y.

`variance0` numeric. The initial value for variance. If variance0=NULL, the initial value will be the variance of phenotype values.

`crit` numeric. The convergence criterion of EM algorithm. The E and M steps will iterate until a convergence criterion is met. It must be a value between 0 and 1.

`stop` numeric. The stopping criterion of EM algorithm. The E and M steps will halt when the iteration number reaches the stopping criterion, treating the algorithm as having failed to converge.

`conv` logical. If set to False, it will disregard the failure to converge and output the last result obtained during the EM algorithm before reaching the stopping criterion.

var.pos	logical. Determines whether the variance of the position of QTLs will be considered in the asymptotic variance-covariance matrix or not.
console	logical. Determines whether the process of the algorithm will be displayed in the R console or not.
IMresult	list. The data list of the output from IM.search(). The required parameters for this function will be extracted from the data list.
MIMresult	list. The data list of the output from MIM.search() or MIM.points(). The required parameters for this function will be extracted from the data list.

Value

E.vector	The QTL effects are calculated by the EM algorithm.
beta	The effects of the fixed factors are calculated by the EM algorithm.
variance	The error variance is calculated by the EM algorithm.
PI.matrix	The posterior probabilities matrix after the process of the EM algorithm.
log.likelihood	The log-likelihood value of this model.
LRT	The LRT statistic of this model.
R2	The coefficient of determination of this model. This can be used as an estimate of heritability.
y.hat	The fitted values of trait values are calculated by the estimated values from the EM algorithm.
iteration.number	The iteration number of the EM algorithm.
avc.matrix	The asymptotic variance-covariance matrix contains position of QTLs, QTL effects, variance, and fix effects.
EMvar	The asymptotic approximate variances include the position of QTLs, QTL effects, variance, and fixed effects. Parameters for which the approximate variance cannot be calculated will be shown as 0. The approximate variance of the position of QTLs is calculated in morgan.

References

KAO, C.-H. and Z.-B. ZENG 1997 General formulas for obtaining the maximum likelihood estimates and the asymptotic variance-covariance matrix in QTL mapping when using the EM algorithm. Biometrics 53, 653-665. <doi: 10.2307/2533965.>

KAO, C.-H., Z.-B. ZENG and R. D. TEASDALE 1999 Multiple interval mapping for Quantitative Trait Loci. Genetics 152: 1203-1216. <doi: 10.1093/genetics/152.3.1203>

See Also

D.make Q.make EM.MIM IM.search MIM.search MIM.points

Examples

```
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# run and result
D.matrix <- D.make(3, type = "RI", aa = c(1, 3, 2, 3), dd = c(1, 2, 1, 3), ad = c(1, 2, 2, 3))
result <- EM.MIMv(QTL, marker, geno, D.matrix, cp.matrix = NULL, y)
result$EMvar
```

EQF.permu *EQF Permutation*

Description

The EQF matrix cluster permutation process for QTL hotspot detection.

Usage

```
EQF.permu(
  LOD.QTLdetect.result,
  npermu = 1000,
  alpha = 0.05,
  Q = TRUE,
  console = interactive()
)
```

Arguments

LOD.QTLdetect.result	list. The data list of the output from LOD.QTLdetect().
npermu	integer. The number of permutations.
alpha	numeric. The type 1 error rate of detecting the hotspot.
Q	logical. When set to TRUE, the function will additionally carry out the permutation of the Q method as the control group, which will be indicated as 'B' in the output.
console	logical. Determines whether the process of the algorithm will be displayed in the R console or not.

Value

EQF.matrix	The matrix denotes the EQF value of each bin.
bin	The bin information matrix used in this analysis.
LOD.threshold	The LOD threshold used in this analysis.
cluster.number	The number of QTLs in each cluster group.
cluster.id	The serial number of traits in each cluster group.

`cluster.matrix` The new EQF matrix after the clustering process.

`permu.matrix.cluster` The permutation result of the clustering method, which has been sorted by order.

`permu.matrix.Q` The permutation result of the Q method, which has been sorted by order.

`EQF.threshold` The EQF threshold is calculated from the permutation process.

References

Wu, P.-Y., M.-.H. Yang, and C.-H. KAO 2021 A Statistical Framework for QTL Hotspot Detection. G3: Genes, Genomes, Genetics: jkab056. <doi: 10.1093/g3journal/jkab056>

See Also

`LOD.QTLdetect` `EQF.plot`

Examples

```
# load the example data
load(system.file("extdata", "LODexample.RDATA", package = "QTLEMM"))

# run and result
result <- EQF.permu(LOD.QTLdetect.result, npermu = 50)
result$cluster.number
```

EQF.plot *EQF plot*

Description

Generate an EQF plot based on the result of the permutation process used to detect the QTL hotspot.

Usage

```
EQF.plot(result, plot.all = TRUE, plot.chr = TRUE, plot.main = TRUE)
```

Arguments

`result`	list. The data list of the output from LOD.QTLdetect(), EQF.permu(), or Qhot.EQF().
`plot.all`	logical. When set to TRUE, it directs the function to output one figure of the EQF values over the bins.
`plot.chr`	logical. When set to TRUE, it instructs the function to output the figures of the EQF values over the bins for each chromosome.
`plot.main`	logical of character. When set to TRUE, it will use the default title on the plot. When set to FALSE, it will be no title on the plot. Users can also use this argument to set their own title.

Value

One or several EQF plots.

References

Wu, P.-Y., M.-.H. Yang, and C.-H. KAO 2021 A Statistical Framework for QTL Hotspot Detection. G3: Genes, Genomes, Genetics: jkab056. <doi: 10.1093/g3journal/jkab056>

See Also

LOD.QTLdetect EQF.permu Qhot.EQF

Examples

```
# load the example data
load(system.file("extdata", "LODexample.RDATA", package = "QTLEMM"))

# run and result
EQF.plot(LOD.QTLdetect.result)
EQF.plot(EQF.permu.result)
```

IM.search *QTL search by IM*

Description

Expectation-maximization algorithm for QTL interval mapping to search for possible positions of QTL in all chromosomes. It can handle genotype data which is selective genotyping too.

Usage

```
IM.search(
  marker,
  geno,
  y,
  yu = NULL,
  sele.g = "n",
  tL = NULL,
  tR = NULL,
  method = "EM",
  type = "RI",
  D.matrix = NULL,
  ng = 2,
  cM = TRUE,
  speed = 1,
  crit = 10^-5,
  d.eff = FALSE,
```

```
  LRT.thre = TRUE,
  simu = 1000,
  alpha = 0.05,
  detect = TRUE,
  QTLdist = 15,
  plot.all = TRUE,
  plot.chr = TRUE,
  console = interactive()
)
```

Arguments

marker	matrix. A k*2 matrix contains the marker information, where the row dimension 'k' represents the number of markers in the chromosomes. The first column labels the chromosomes where the markers are located, and the second column labels the positions of markers (in morgan (M) or centimorgan (cM)). It's important to note that chromosomes and positions must be sorted in order.
geno	matrix. A n*k matrix contains the genotypes of k markers for n individuals. The marker genotypes of P1 homozygote (MM), heterozygote (Mm), and P2 homozygote (mm) are coded as 2, 1, and 0, respectively, with NA indicating missing values.
y	vector. A vector that contains the phenotype values of individuals with genotypes.
yu	vector. A vector that contains the phenotype values of individuals without genotypes.
sele.g	character. Determines the type of data being analyzed: If sele.g="n", it considers the data as complete genotyping data. If sele.g="f", it treats the data as selective genotyping data and utilizes the proposed corrected frequency model (Lee 2014) for analysis; If sele.g="t", it considers the data as selective genotyping data and uses the truncated model (Lee 2014) for analysis; If sele.g="p", it treats the data as selective genotyping data and uses the population frequency-based model (Lee 2014) for analysis. Note that the 'yu' argument must be provided when sele.g="f" or "p".
tL	numeric. The lower truncation point of phenotype value when sele.g="t". When sele.g="t" and tL=NULL, the 'yu' argument must be provided. In this case, the function will consider the minimum of 'yu' as the lower truncation point.
tR	numeric. The upper truncation point of phenotype value when sele.g="t". When sele.g="t" and tR=NULL, the 'yu' argument must be provided. In this case, the function will consider the maximum of 'yu' as the upper truncation point.
method	character. When method="EM", it indicates that the interval mapping method by Lander and Botstein (1989) is used in the analysis. Conversely, when method="REG", it indicates that the approximate regression interval mapping method by Haley and Knott (1992) is used in the analysis.
type	character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".

D.matrix	matrix. The design matrix of the IM model. If D.matrix=NULL, the design matrix will be constructed using Cockerham's model: In BC population, it is a 2*1 matrix with values 0.5 and -0.5 for the additive effect; In RI or AI population, it is a 3*2 matrix. The first column consists of 1, 0, and -1 for the additive effect, and the second column consists of 0.5, -0.5, and 0.5 for the dominant effect.
ng	integer. The generation number of the population type. For instance, in a BC1 population where type="BC", ng=1; in an AI F3 population where type="AI", ng=3.
cM	logical. Specify the unit of marker position. If cM=TRUE, it denotes centimorgan; if cM=FALSE, it denotes morgan.
speed	numeric. The walking speed of the QTL search (in cM).
crit	numeric. The convergence criterion of EM algorithm. The E and M steps will iterate until a convergence criterion is met. It must be a value between 0 and 1.
d.eff	logical. Specifies whether the dominant effect will be considered in the parameter estimation for AI or RI population.
LRT.thre	logical or numeric. If set to TRUE, the LRT threshold will be computed based on the Gaussian stochastic process (Kao and Ho 2012). Alternatively, users can input a numerical value as the LRT threshold to evaluate the significance of QTL detection.
simu	integer. Determines the number of simulation samples that will be used to compute the LRT (Likelihood Ratio Test) threshold using the Gaussian process. It must be a value between 50 and 10^8.
alpha	numeric. The type I error rate for the LRT threshold.
detect	logical. Determines whether the significant QTL, whose LRT statistic is larger than the LRT threshold, will be displayed in the output dataset or not.
QTLdist	numeric. The minimum distance (in cM) among different linked significant QTL.
plot.all	logical. When set to TRUE, it directs the function to output the profile of LRT statistics for the genome in one figure.
plot.chr	logical. When set to TRUE, it instructs the function to output the profile of LRT statistics for the chromosomes.
console	logical. Determines whether the process of the algorithm will be displayed in the R console or not.

Value

effect	The estimated effects and LRT statistics of all positions.
LRT.threshold	The LRT threshold value computed for the data using the Gaussian stochastic process (Kuo 2011; Kao and Ho 2012).
detect.QTL	The positions, effects and LRT statistics of the detected QTL significant using the obtained LRT threshold value.
model	The model of selective genotyping data in this analyze.
inputdata	The input data of this analysis. It contains marker, geno, y, yu, sele.g, type, ng, cM, and d.eff. The parameters not provided by the user will be output with default values.

Graphical outputs including LOD value and effect of each position.

References

KAO, C.-H. and Z.-B. ZENG 1997 General formulas for obtaining the maximum likelihood estimates and the asymptotic variance-covariance matrix in QTL mapping when using the EM algorithm. Biometrics 53, 653-665. <doi: 10.2307/2533965.>

KAO, C.-H., Z.-B. ZENG and R. D. TEASDALE 1999 Multiple interval mapping for Quantitative Trait Loci. Genetics 152: 1203-1216. <doi: 10.1093/genetics/152.3.1203>

H.-I LEE, H.-A. HO and C.-H. KAO 2014 A new simple method for improving QTL mapping under selective genotyping. Genetics 198: 1685-1698. <doi: 10.1534/genetics.114.168385.>

KAO, C.-H. and H.-A. Ho 2012 A score-statistic approach for determining threshold values in QTL mapping. Frontiers in Bioscience. E4, 2670-2682. <doi: 10.2741/e582>

See Also

EM.MIM LRTthre

Examples

```
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# run and result
result <- IM.search(marker, geno, y, type = "RI", ng = 2, speed = 7, crit = 10^-3, LRT.thre = 10)
result$detect.QTL

## Not run:
# Example for selective genotyping data
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# make the seletive genotyping data
ys <- y[y > quantile(y)[4] | y < quantile(y)[2]]
yu <- y[y >= quantile(y)[2] & y <= quantile(y)[4]]
geno.s <- geno[y > quantile(y)[4] | y < quantile(y)[2],]

# run and result
result <- IM.search(marker, geno.s, ys, yu, sele.g = "f", type = "RI", ng = 2,
speed = 7, crit = 10^-3, LRT.thre = 10)
result$detect.QTL

## End(Not run)
```

LOD.QTLdetect *QTL Detect by LOD*

Description

Detect QTL by the likelihood of odds (LOD) matrix.

Usage

```
LOD.QTLdetect(LOD, bin, thre = 3, QTLdist = 20, console = interactive())
```

Arguments

LOD	matrix. The LOD matrix, which is a t*p matrix, where t is the number of traits and p is the number of bins on the chromosomes. Missing values should be denoted as NA in the matrix.
bin	matrix. An n*2 matrix that represents the number of bins on each chromosome, where n is the number of chromosomes. The first column denotes the chromosome number, and the second column denotes the number of bins on that chromosome. It's important to ensure that chromosomes are divided in order.
thre	numeric. The LOD threshold. Any LOD score under this threshold will be calculated as 0.
QTLdist	numeric. The minimum distance (in bins) among different linked significant QTL.
console	logical. Determines whether the process of the algorithm will be displayed in the R console or not.

Value

detect.QTL.number	The number of detected QTL in each trait.
QTL.matrix	The QTL position matrix. Where the elements 1 donates the position of QTL; elements 0 donate the bins whose LOD score is under the LOD threshold; other positions are shown as NA.
EQF.matrix	The matrix denotes the EQF value of each bin.
linkage.QTL.number	The linkage QTL number of all detected QTL. In other words, it is the table that denote how many QTL are on one chromosome.
LOD.threshold	The LOD threshold used in this analysis.
bin	The bin information matrix used in this analysis.

References

Wu, P.-Y., M.-.H. Yang, and C.-H. KAO 2021 A Statistical Framework for QTL Hotspot Detection. G3: Genes, Genomes, Genetics: jkab056. <doi: 10.1093/g3journal/jkab056>

See Also

EQF.permu EQF.plot

Examples

```
# load the example data
load(system.file("extdata", "LODexample.RDATA", package = "QTLEMM"))
dim(LODexample) # 100 traits, 633 bins on chromosome

# run and result
result <- LOD.QTLdetect(LODexample, bin, thre = 3, QTLdist = 10)
result$detect.QTL.number
```

LRTthre *LRT Threshold*

Description

The LRT threshold for QTL interval mapping based on the Gaussian stochastic process (Kao and Ho 2012).

Usage

```
LRTthre(
  marker,
  type = "RI",
  ng = 2,
  cM = TRUE,
  ns = 200,
  gv = 25,
  speed = 1,
  simu = 1000,
  d.eff = FALSE,
  alpha = 0.05,
  console = interactive()
)
```

Arguments

marker	matrix. A k*2 matrix contains the marker information, where the row dimension 'k' represents the number of markers in the chromosomes. The first column labels the chromosomes where the markers are located, and the second column labels the positions of markers (in morgan (M) or centimorgan (cM)). It's important to note that chromosomes and positions must be sorted in order.
type	character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".
ng	integer. The generation number of the population type. For instance, in a BC1 population where type="BC", ng=1; in an AI F3 population where type="AI", ng=3.

cM	logical. Specify the unit of marker position. If cM=TRUE, it denotes centimorgan; if cM=FALSE, it denotes morgan.
ns	integer. The number of individuals for generating the individual trait values. Changes in this value do not significantly affect the outcome of the LRT threshold value.
gv	numeric. The genetic variance for generating the individual trait values. Changes in this value do not significantly affect the outcome of the LRT threshold value.
speed	numeric. The walking speed of the QTL analysis (in cM).
simu	integer. Determines the number of simulation samples that will be used to compute the LRT threshold using the Gaussian process.
d.eff	logical. Specifies whether the dominant effect will be considered in the parameter estimation for AI or RI population.
alpha	numeric. The type I error rate for the LRT threshold.
console	logical. Determines whether the process of the algorithm will be displayed in the R console or not.

Value

The LRT threshold for QTL interval mapping.

References

KAO, C.-H. and H.-A. Ho 2012 A score-statistic approach for determining threshold values in QTL mapping. Frontiers in Bioscience. E4, 2670-2682. <doi: 10.2741/e582>

See Also

rmvnorm

Examples

```
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# run and result
LRTthre(marker, type = "RI", ng = 2, speed = 2, simu = 60)
```

MIM.points *QTL Short Distance Correction by MIM*

Description

Expectation-maximization algorithm for QTL multiple interval mapping to find the best QTL position near the designated QTL position. It can handle genotype data which is selective genotyping too.

Usage

```
MIM.points(
  QTL = NULL,
  marker = NULL,
  geno = NULL,
  y = NULL,
  yu = NULL,
  sele.g = "n",
  tL = NULL,
  tR = NULL,
  method = "EM",
  type = "RI",
  D.matrix = NULL,
  ng = 2,
  cM = TRUE,
  scope = 5,
  speed = 1,
  crit = 10^-3,
  console = interactive(),
  IMresult = NULL,
  MIMresult = NULL
)
```

Arguments

QTL	matrix. A q*2 matrix contains the QTL information, where the row dimension 'q' represents the number of QTLs in the chromosomes. The first column labels the chromosomes where the QTLs are located, and the second column labels the positions of QTLs (in morgan (M) or centimorgan (cM)).
marker	matrix. A k*2 matrix contains the marker information, where the row dimension 'k' represents the number of markers in the chromosomes. The first column labels the chromosomes where the markers are located, and the second column labels the positions of markers (in morgan (M) or centimorgan (cM)). It's important to note that chromosomes and positions must be sorted in order.
geno	matrix. A n*k matrix contains the genotypes of k markers for n individuals. The marker genotypes of P1 homozygote (MM), heterozygote (Mm), and P2 homozygote (mm) are coded as 2, 1, and 0, respectively, with NA indicating missing values.
y	vector. A vector that contains the phenotype values of individuals with genotypes.
yu	vector. A vector that contains the phenotype values of individuals without genotypes.
sele.g	character. Determines the type of data being analyzed: If sele.g="n", it considers the data as complete genotyping data. If sele.g="f", it treats the data as selective genotyping data and utilizes the proposed corrected frequency model (Lee 2014) for analysis; If sele.g="t", it considers the data as selective genotyping data and uses the truncated model (Lee 2014) for analysis; If sele.g="p", it treats the data

as selective genotyping data and uses the population frequency-based model (Lee 2014) for analysis. Note that the 'yu' argument must be provided when sele.g="f" or "p".

`tL` numeric. The lower truncation point of phenotype value when sele.g="t". When sele.g="t" and tL=NULL, the 'yu' argument must be provided. In this case, the function will consider the minimum of 'yu' as the lower truncation point.

`tR` numeric. The upper truncation point of phenotype value when sele.g="t". When sele.g="t" and tR=NULL, the 'yu' argument must be provided. In this case, the function will consider the maximum of 'yu' as the upper truncation point.

`method` character. When method="EM", it indicates that the interval mapping method by Lander and Botstein (1989) is used in the analysis. Conversely, when method="REG", it indicates that the approximate regression interval mapping method by Haley and Knott (1992) is used in the analysis.

`type` character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".

`D.matrix` matrix. The design matrix of QTL effects is a g*p matrix, where g is the number of possible QTL genotypes, and p is the number of effects considered in the MIM model. This design matrix can be easily generated by the function D.make(). If set to NULL, it will automatically generate a design matrix with all additive and dominant effects and without any epistasis effect.

`ng` integer. The generation number of the population type. For instance, in a BC1 population where type="BC", ng=1; in an AI F3 population where type="AI", ng=3.

`cM` logical. Specify the unit of marker position. If cM=TRUE, it denotes centimorgan; if cM=FALSE, it denotes morgan.

`scope` numeric vector. During the MIM process, it will search forward and backward for the corresponding centimorgan (cM). Users can assign a numeric number for every QTL or a numeric vector for each QTL. Note that 0 denotes that the corresponding QTL position is fixed, and the positions of its surrounding intervals will not be searched.

`speed` numeric. The walking speed of the QTL search (in cM).

`crit` numeric. The convergence criterion of EM algorithm. The E and M steps will iterate until a convergence criterion is met. It must be a value between 0 and 1.

`console` logical. Determines whether the process of the algorithm will be displayed in the R console or not.

`IMresult` list. The data list of the output from IM.search(). The required parameters for this function will be extracted from the data list.

`MIMresult` list. The data list of the output from MIM.search() or MIM.points(). The required parameters for this function will be extracted from the data list.

Value

`effect` The estimated effects, log likelihood value, and LRT statistics of all searched positions.

QTL.best	The positions of the best QTL combination.
effect.best	The estimated effects and LRT statistics of the best QTL combination.
model	The model of selective genotyping data in this analysis.
inputdata	The input data of this analysis. It contains marker, geno, y, yu, sele.g, type, ng, cM, and D.matrix. The parameters not provided by the user will be output with default values.

Note

When IMresult and MIMresult are entered simultaneously, only IMresult will be processed.

If an error occurs, please check whether the original output data of IM.search(), MIM.result(), or MIM.points() is used.

References

KAO, C.-H. and Z.-B. ZENG 1997 General formulas for obtaining the maximum likelihood estimates and the asymptotic variance-covariance matrix in QTL mapping when using the EM algorithm. Biometrics 53, 653-665. <doi: 10.2307/2533965.>

KAO, C.-H., Z.-B. ZENG and R. D. TEASDALE 1999 Multiple interval mapping for Quantitative Trait Loci. Genetics 152: 1203-1216. <doi: 10.1093/genetics/152.3.1203>

H.-I LEE, H.-A. HO and C.-H. KAO 2014 A new simple method for improving QTL mapping under selective genotyping. Genetics 198: 1685-1698. <doi: 10.1534/genetics.114.168385.>

See Also

EM.MIM IM.search MIM.search

Examples

```
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# run and result
result <- MIM.points(QTL, marker, geno, y, type = "RI", ng = 2, scope = c(0,1,2), speed = 2)
result$QTL.best
result$effect.best

## Not run:
# Example for selective genotyping data
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# make the seletive genotyping data
ys <- y[y > quantile(y)[4] | y < quantile(y)[2]]
yu <- y[y >= quantile(y)[2] & y <= quantile(y)[4]]
geno.s <- geno[y > quantile(y)[4] | y < quantile(y)[2],]

# run and result
result <- MIM.points(QTL, marker, geno.s, ys, yu, sele.g = "f",
```

```
 type = "RI", ng = 2, scope = c(0,1,2), speed = 2)
result$QTL.best
result$effect.best

## End(Not run)
```

MIM.search *QTL search by MIM*

Description

Expectation-maximization algorithm for QTL multiple interval mapping to find one more QTL in the presence of some known QTLs. It can handle genotype data which is selective genotyping too.

Usage

```
MIM.search(
  QTL = NULL,
  marker = NULL,
  geno = NULL,
  y = NULL,
  yu = NULL,
  sele.g = "n",
  tL = NULL,
  tR = NULL,
  method = "EM",
  type = "RI",
  D.matrix = NULL,
  ng = 2,
  cM = TRUE,
  speed = 1,
  QTLdist = 15,
  link = TRUE,
  crit = 10^-3,
  console = interactive(),
  IMresult = NULL,
  MIMresult = NULL
)
```

Arguments

QTL	matrix. A q*2 matrix contains the QTL information, where the row dimension 'q' represents the number of QTLs in the chromosomes. The first column labels the chromosomes where the QTLs are located, and the second column labels the positions of QTLs (in morgan (M) or centimorgan (cM)).

marker	matrix. A k*2 matrix contains the marker information, where the row dimension 'k' represents the number of markers in the chromosomes. The first column labels the chromosomes where the markers are located, and the second column labels the positions of markers (in morgan (M) or centimorgan (cM)). It's important to note that chromosomes and positions must be sorted in order.
geno	matrix. A n*k matrix contains the genotypes of k markers for n individuals. The marker genotypes of P1 homozygote (MM), heterozygote (Mm), and P2 homozygote (mm) are coded as 2, 1, and 0, respectively, with NA indicating missing values.
y	vector. A vector that contains the phenotype values of individuals with genotypes.
yu	vector. A vector that contains the phenotype values of individuals without genotypes.
sele.g	character. Determines the type of data being analyzed: If sele.g="n", it considers the data as complete genotyping data. If sele.g="f", it treats the data as selective genotyping data and utilizes the proposed corrected frequency model (Lee 2014) for analysis; If sele.g="t", it considers the data as selective genotyping data and uses the truncated model (Lee 2014) for analysis; If sele.g="p", it treats the data as selective genotyping data and uses the population frequency-based model (Lee 2014) for analysis. Note that the 'yu' argument must be provided when sele.g="f" or "p".
tL	numeric. The lower truncation point of phenotype value when sele.g="t". When sele.g="t" and tL=NULL, the 'yu' argument must be provided. In this case, the function will consider the minimum of 'yu' as the lower truncation point.
tR	numeric. The upper truncation point of phenotype value when sele.g="t". When sele.g="t" and tR=NULL, the 'yu' argument must be provided. In this case, the function will consider the maximum of 'yu' as the upper truncation point.
method	character. When method="EM", it indicates that the interval mapping method by Lander and Botstein (1989) is used in the analysis. Conversely, when method="REG", it indicates that the approximate regression interval mapping method by Haley and Knott (1992) is used in the analysis.
type	character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".
D.matrix	matrix. The design matrix of QTL effects is a g*p matrix, where g is the number of possible QTL genotypes, and p is the number of effects considered in the MIM model. It's important to note that the QTL number of the design matrix must be the original QTL number plus one. The design matrix can be easily generated by the function D.make(). If set to NULL, it will automatically generate a design matrix with all additive and dominant effects and without any epistasis effect.
ng	integer. The generation number of the population type. For instance, in a BC1 population where type="BC", ng=1; in an AI F3 population where type="AI", ng=3.
cM	logical. Specify the unit of marker position. If cM=TRUE, it denotes centimorgan; if cM=FALSE, it denotes morgan.

speed	numeric. The walking speed of the QTL search (in cM).
QTLdist	numeric. The minimum distance (in cM) among different linked significant QTL. Positions near the known QTLs within this distance will not be considered as candidate positions in the search process.
link	logical. When set to False, positions on the same chromosomes as the known QTLs will not be searched.
crit	numeric. The convergence criterion of EM algorithm. The E and M steps will iterate until a convergence criterion is met. It must be a value between 0 and 1.
console	logical. Determines whether the process of the algorithm will be displayed in the R console or not.
IMresult	list. The data list of the output from IM.search(). The required parameters for this function will be extracted from the data list.
MIMresult	list. The data list of the output from MIM.search() or MIM.points(). The required parameters for this function will be extracted from the data list.

Value

effect	The estimated effects, log-likelihood value, and LRT statistics of all searched positions.
QTL.best	The positions of the best QTL combination.
effect.best	The estimated effects and LRT statistics of the best QTL combination.
model	The model of selective genotyping data in this analyze.
inputdata	The input data of this analysis. It contains marker, geno, y, yu, sele.g, type, ng, cM, and D.matrix. The parameters not provided by the user will be output with default values.

Note

When IMresult and MIMresult are entered simultaneously, only IMresult will be processed.

If an error occurs, please check whether the original output data of IM.search(), MIM.result(), or MIM.points() is used.

References

KAO, C.-H. and Z.-B. ZENG 1997 General formulas for obtaining the maximum likelihood estimates and the asymptotic variance-covariance matrix in QTL mapping when using the EM algorithm. Biometrics 53, 653-665. <doi: 10.2307/2533965.>

KAO, C.-H., Z.-B. ZENG and R. D. TEASDALE 1999 Multiple interval mapping for Quantitative Trait Loci. Genetics 152: 1203-1216. <doi: 10.1093/genetics/152.3.1203>

H.-I LEE, H.-A. HO and C.-H. KAO 2014 A new simple method for improving QTL mapping under selective genotyping. Genetics 198: 1685-1698. <doi: 10.1534/genetics.114.168385.>

See Also

EM.MIM IM.search MIM.points

Examples

```
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# run and result
QTL <- c(1, 23)
result <- MIM.search(QTL, marker, geno, y, type = "RI", ng = 2, speed = 15, QTLdist = 50)
result$QTL.best
result$effect.best

## Not run:
# Example for selective genotyping data
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# make the seletive genotyping data
ys <- y[y > quantile(y)[4] | y < quantile(y)[2]]
yu <- y[y >= quantile(y)[2] & y <= quantile(y)[4]]
geno.s <- geno[y > quantile(y)[4] | y < quantile(y)[2],]

# run and result
QTL <- c(1, 23)
result <- MIM.search(QTL, marker, geno.s, ys, yu, sele.g = "f",
 type = "RI", ng = 2, speed = 15, QTLdist = 50)
result$QTL.best
result$effect.best

## End(Not run)
```

---

progeny *Progeny Simulation*

Description

Generate simulated phenotype and genotype data for a specified generation from various breeding schemes.

Usage

```
progeny(
  QTL,
  marker,
  type = "RI",
  ng = 2,
  cM = TRUE,
  E.vector = NULL,
  h2 = 0.5,
  size = 200
)
```

Arguments

QTL	matrix. A q*2 matrix contains the QTL information, where the row dimension 'q' represents the number of QTLs in the chromosomes. The first column labels the chromosomes where the QTLs are located, and the second column labels the positions of QTLs (in morgan (M) or centimorgan (cM)).
marker	matrix. A k*2 matrix contains the marker information, where the row dimension 'k' represents the number of markers in the chromosomes. The first column labels the chromosomes where the markers are located, and the second column labels the positions of markers (in morgan (M) or centimorgan (cM)). It's important to note that chromosomes and positions must be sorted in order.
type	character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".
ng	integer. The generation number of the population type. For instance, in a BC1 population where type="BC", ng=1; in an AI F3 population where type="AI", ng=3.
cM	logical. Specify the unit of marker position. If cM=TRUE, it denotes centimorgan; if cM=FALSE, it denotes morgan.
E.vector	vector. Set the effect of QTLs. It should be a named vector, where the names of elements represent the effects of QTLs and their interactions. For example: the additive effect of QTL1 is coded as "a1"; the dominant effect of QTL2 is coded as "d2"; the interaction of the additive effect of QTL2 and the dominant effect of QTL1 is coded as "a2:d1". So, if the additive effect of QTL1 is 2, the dominant effect of QTL2 is 5, and the interaction of the additive effect of QTL2 and the dominant effect of QTL1 is 3, the user should input: E.vector = c("a1"=2, "d2"=5, "a2:d1"=3). If E.vector=NULL, the phenotypic value will not be simulated.
h2	numeric. Set the heritability for simulated phenotypes. It should be a number between 0 and 1.
size	numeric. The population size of simulated progeny.

Value

phe	The phenotypic value of each simulated progeny.
E.vector	The effect vector used in this simulation.
marker.prog	The marker genotype of each simulated progeny.
QTL.prog	The QTL genotype of each simulated progeny.
VG	The genetic variance of this population.
VE	The environmental variance of this population.
genetic.value	The genetic value of each simulated progeny.

References

Haldane J.B.S. 1919. The combination of linkage values and the calculation of distance between the loci for linked factors. Genetics 8: 299–309.

Examples

```
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# run and result
result <- progeny(QTL, marker, type = "RI", ng = 5, E.vector = c("a1" = 2, "d2" = 5, "a2:d1" = 3),
h2 = 0.5, size = 200)
result$phe
```

`Q.make` *Generate Q Matrix*

Description

Generate the conditional probability matrix using the information of QTL and marker, along with the genotype data.

Usage

```
Q.make(
  QTL,
  marker,
  geno = NULL,
  interval = FALSE,
  type = "RI",
  ng = 2,
  cM = TRUE
)
```

Arguments

`QTL`	matrix. A q*2 matrix contains the QTL information, where the row dimension 'q' represents the number of QTLs in the chromosomes. The first column labels the chromosomes where the QTLs are located, and the second column labels the positions of QTLs (in morgan (M) or centimorgan (cM)).
`marker`	matrix. A k*2 matrix contains the marker information, where the row dimension 'k' represents the number of markers in the chromosomes. The first column labels the chromosomes where the markers are located, and the second column labels the positions of markers (in morgan (M) or centimorgan (cM)). It's important to note that chromosomes and positions must be sorted in order.
`geno`	matrix. A n*k matrix contains the genotypes of k markers for n individuals. The marker genotypes of P1 homozygote (MM), heterozygote (Mm), and P2 homozygote (mm) are coded as 2, 1, and 0, respectively, with NA indicating missing values.
`interval`	logical. When set to interval=TRUE, if a QTL shares the same position as a marker, the marker will be skipped and not considered as a flanking marker.

type	character. The population type of the dataset. Includes backcross (type="BC"), advanced intercross population (type="AI"), and recombinant inbred population (type="RI"). The default value is "RI".
ng	integer. The generation number of the population type. For instance, in a BC1 population where type="BC", ng=1; in an AI F3 population where type="AI", ng=3.
cM	logical. Specify the unit of marker position. If cM=TRUE, it denotes centimorgan; if cM=FALSE, it denotes morgan.

Value

The output contains k conditional probability matrices for the k flanking marker pairs (the k Q-matrices) and a conditional probability matrix of each QTL for all individuals (the cp-matrix) provided the genotype data of the testing population is input..

Note

If geno=NULL, the function can still be executed, and the output will contain k Q-matrices but no cp-matrix.

References

KAO, C.-H. and Z.-B. ZENG 1997 General formulas for obtaining the maximum likelihood estimates and the asymptotic variance-covariance matrix in QTL mapping when using the EM algorithm. Biometrics 53, 653-665. <doi: 10.2307/2533965.>

KAO, C.-H., Z.-B. ZENG and R. D. TEASDALE 1999 Multiple interval mapping for Quantitative Trait Loci. Genetics 152: 1203-1216. <doi: 10.1093/genetics/152.3.1203>

Examples

```
# load the example data
load(system.file("extdata", "exampledata.RDATA", package = "QTLEMM"))

# run and result
result <- Q.make(QTL, marker, geno)
head(result$cp.matrix)
```

Qhot *QTL Hotspot*

Description

This function generates both numerical and graphical summaries of the QTL hotspot detection in the genomes, including information about the flanking markers of QTLs.

Usage

```
Qhot(
  DataQTL,
  DataCrop,
  ScanStep = 1,
  NH = 100,
  NP = 1000,
  save.pdf = interactive()
)
```

Arguments

DataQTL	data.frame. A data frame with 5 columns for QTL information. The columns represent the serial number of QTLs, the trait names, the chromosome numbers, the left flanking marker positions(in cM) of QTLs, and the right flanking marker positions(in cM) of QTLs.
DataCrop	data.frame. A data frame with 3 columns for chromosome information consists of the chromosome names, the center positions(in cM) and the lengths of chromosomes.
ScanStep	numeric. A value for the length(cM) of every bin.
NH	integer. A value for the number of spurious hotspots in the proposed method.
NP	integer. A value for permutation times to calculate the threshold.
save.pdf	logical. When set to TRUE, the PDF file of plots will be saved in the working directory instead of being displayed in the console.

Value

EQF	The expected QTL frequency(EQF) in every bin per chromosome.
P.threshold	The EQF thresholds for proposed method.
Q.threshold	The EQF thresholds for the Q method.
nHot	The numbers of detected hotspots per chromosome for the proposed method and Q method.

Graphical outputs for visualizing the summarized results includes the expected QTL frequency of scan steps, and the composition of QTLs for different traits in the detected hotspots.

Note

This program may generate a large amount of graphical output. To manage this, it's recommended to save the results in a PDF file using the "save.pdf" argument.

References

Wu, P.-Y., M.-.H. Yang, and C.-H. KAO 2021 A Statistical Framework for QTL Hotspot Detection. G3: Genes, Genomes, Genetics: jkab056. <doi: 10.1093/g3journal/jkab056>

Examples

```
# load the example data
load(system.file("extdata", "QHOTexample.RDATA", package = "QTLEMM"))

# run and result
result <- Qhot(QTL.example, crop.example, 5, 20, 100, save.pdf = FALSE)
```

Qhot.EQF *EQF Matrix Conversion*

Description

Convert the QTL flanking marker data to EQF matrix. And the EQF matrix cluster permutation process can be further carried out to detect QTL hotspots.

Usage

```
Qhot.EQF(
  DataQTL,
  Datachr,
  bin.size = 0.5,
  permu = TRUE,
  ptime = 1000,
  alpha = 0.05,
  Q = TRUE,
  console = interactive()
)
```

Arguments

DataQTL	data.frame. A data frame with 5 columns for QTL information. The columns represent the serial number of QTLs, the trait names, the chromosome numbers, the left flanking marker positions(in cM) of QTLs, and the right flanking marker positions(in cM) of QTLs.
Datachr	vector. The length of each chromosome(in cM).
bin.size	numeric. The bin size(in cM) for QTL detection. If the distance of flanking marker of a QTL is lass than the bin size, it will be mark in the EQF matrix and will participate in the cluster grouping process.
permu	logical. When set to TRUE, the function will carry out the cluster grouping process and cluster group permutation.
ptime	integer. The permutation times.
alpha	numeric. The type 1 error rate of detecting the hotspot.
Q	logical. When set to TRUE, the function will additionally carry out the permutation of the Q method as the control group, which will be indicated as 'B' in the output.
console	logical. Determines whether the process of the algorithm will be displayed in the R console or not.

Value

`EQF.matrix`	The matrix denotes the EQF value of each bin for every QTL in this database.
`bin`	The bin information matrix whose first column denotes the chromosome number and the second column denotes the number of bins on that chromosome.
`bin.size`	The bin size set in this analysis.
`EQF.trait`	The matrix denotes the EQF value of each bin for every trait of this database.
`EQF.detect`	The matrix denotes the EQF value of each bin for the trait that have the QTL detected in the set bin size.
`EQF.nondetect`	The matrix denotes the EQF value of each bin for the trait that have no QTL detected in the set bin size.
`cluster.matrix`	The new EQF matrix after the clustering process.
`permu.matrix.cluster`	The permutation result of the clustering method, which has been sorted by order.
`permu.matrix.Q`	The permutation result of the Q method, which has been sorted by order.
`EQF.threshold`	The EQF threshold is calculated from the permutation process.

References

Wu, P.-Y., M.-.H. Yang, and C.-H. KAO 2021 A Statistical Framework for QTL Hotspot Detection. G3: Genes, Genomes, Genetics: jkab056. <doi: 10.1093/g3journal/jkab056>

See Also

`Qhot` `EQF.plot`

Examples

```
# load the example data
load(system.file("extdata", "QHOTEQFexample.RDATA", package = "QTLEMM"))

#' # run and result
result <- Qhot.EQF(QTL.example, chr.example, bin.size = 2, permu = TRUE,
ptime = 100, alpha = 0.05, Q = FALSE)
```

Index

D.make, 2, *7*, *10*

EM.MIM, 4, *10*, *16*, *22*, *25*
EM.MIMv, 8
EQF.permu, 11, *13*, *17*
EQF.plot, *12*, 12, *17*, *32*

IM.search, *7*, *10*, 13, *22*, *25*

LOD.QTLdetect, *12, 13*, 16
LRTthre, *16*, 18

MIM.points, *7*, *10*, 19, *25*
MIM.search, *7*, *10*, *22*, 23

progeny, 26

Q.make, *7*, *10*, 28
Qhot, 29, *32*
Qhot.EQF, *13*, 31

rmvnorm, *19*