

Package ‘hsphase’

April 6, 2025

Type Package

Title Phasing, Pedigree Reconstruction, Sire Imputation and
Recombination Events Identification of Half-sib Families Using
SNP Data

Version 2.0.4

Date 2025-04-05

Depends snowfall, R (\geq 3.1.0)

LinkingTo RcppArmadillo (\geq 0.4.300.8.0), Rcpp (\geq 0.11.2)

Imports Rcpp

Description Identification of recombination events, haplotype reconstruction, sire imputation and pedigree reconstruction using half-sib family SNP data.

License GPL-3

NeedsCompilation yes

Repository CRAN

RoxygenNote 7.3.1

Author Mohammad Ferdosi [aut, cre],
Cedric Gondro [aut]

Maintainer Mohammad Ferdosi <mhferdosi@yahoo.com>

Date/Publication 2025-04-06 03:00:02 UTC

Contents

hsphase-package	2
.fastdist	4
.maf	4
.simulateHalfsib	5
aio	6
bmh	7
co	8
cs	9
genotypes	10

hbp	10
hh	11
hss	12
imageplot	13
impute	14
map	15
ohd	15
ohg	16
ohplot	17
para	18
pedigree	19
pedigreeNaming	20
phf	20
pm	21
pogc	22
readGenotype	23
recombinations	24
rplot	25
rpoh	26
ssp	27

Index	29
--------------	-----------

hsphase-package	<i>Phasing, Pedigree Reconstruction, Sire Imputation and Recombination Events Identification for Half-sib Families</i>
-----------------	--

Description

Identification of recombination events, haplotype reconstruction and sire imputation using half-sib family SNP data.

Details

Package: hsphase
 Type: Package
 Version: 2.0.1
 Date: 2014-6-17
 License: GPL 3

Main Functions:

[bmh](#): Block partitioning
[ssp](#): Sire inference
[ai](#): Phasing
[imageplot](#): Image plot of the block structure

rpoh: Reconstruct pedigree based on opposing homozygote

Auxiliary Functions

hss: Half-sib family splitter

cs: Chromosome splitter

para: Parallel data analysis

Note: These functions can be used to analyse large datasets.

Author(s)

Mohammad H. Ferdosi <mferdosi@une.edu.au>, Cedric Gondro <cgondro2@une.edu.au> Maintainer: Mohammad H. Ferdosi <mferdosi@myune.edu.au>

References

Ferdosi, M. H., Kinghorn, B. P., van der Werf, J. H., & Gondro, C (2013). Effect of genotype and pedigree error on detection of recombination events, sire imputation and haplotype inference using the hsphase algorithm. In Proc. Assoc. Advmt. Anim. Breed. Genet (Vol. 20, pp. 546-549). AAABG; Napier, New Zealand.

Ferdosi, M. H., Kinghorn, B. P., van der Werf, J. H., & Gondro, C. (2014). Detection of recombination events, haplotype reconstruction and imputation of sires using half-sib SNP genotypes. Genetics, selection, evolution: GSE, 46(1), 11.

Ferdosi, M. H., Kinghorn, B. P., van der Werf, J. H., Lee, S. H., & Gondro, C. (2014). hsphase: an R package for pedigree reconstruction, detection of recombination events, phasing and imputation of half-sib family groups. BMC Bioinformatics, 15(1), 172.

Ferdosi, M. H., & Boerner, V. (2014). A fast method for evaluating opposing homozygosity in large SNP data sets. Livestock Science.

Examples

```
genotype <- matrix(c(
  0,0,0,0,1,2,2,2,0,0,2,0,0,0,
  2,2,2,2,1,0,0,0,2,2,2,2,2,2,
  2,2,2,2,1,2,2,2,0,0,2,2,2,2,
  2,2,2,2,0,0,0,0,2,2,2,2,2,2,
  0,0,0,0,0,2,2,2,2,2,0,0,0), ncol = 14, byrow = TRUE)
ssp(bmh(genotype), genotype)
aio(genotype)
imageplot(bmh(genotype), title = "ImagePlot example")
rplot(genotype, c(1:14))
```

.fastdist *Calculate Genotypic Distances*

Description

Calculates a symmetric matrix of distances between genotypes, based on a given genotype matrix. Each row in the 'GenotypeMatrix' represents a genotype, and each column represents a marker. The genotype is coded as 0 for AA, 1 for AB, and 2 for BB. Use 9 to represent missing data.

Usage

```
.fastdist(GenotypeMatrix)
```

Arguments

GenotypeMatrix A matrix where each row represents a genotype and each column represents a marker. Genotypes should be coded as 0 for AA, 1 for AB, and 2 for BB, with 9 representing missing data.

Value

Returns a symmetric matrix of distances between the genotypes specified in the 'GenotypeMatrix'. Row and column names of the returned matrix correspond to the row names of the 'GenotypeMatrix'.

Examples

```
# Simulate genotype data for 40 individuals across 1000 SNPs
# genotypes <- simulateHalfSib(numInd = 40, numSNP = 1000, recbound = 0:6, type = "genotype")
# Calculate the distance matrix
# dist_matrix <- fastdist(genotypes)
# Display the distance matrix
# print(dist_matrix)
```

.maf *Calculate Minor Allele Frequency (MAF)*

Description

This function calculates the minor allele frequency (MAF) for a given single nucleotide polymorphism (SNP) data. The SNP data should be coded numerically: 0 for homozygous for the first allele (AA), 1 for heterozygous (AB), and 2 for homozygous for the second allele (BB). Missing data should be coded as 9.

Usage

```
.maf(snp)
```

Arguments

snp A numeric vector representing the genotype of individuals for a single SNP. The genotype should be coded as 0 for AA, 1 for AB, and 2 for BB. Use 9 to represent missing data.

Value

A numeric value representing the minor allele frequency (MAF) for the SNP data provided.

Examples

```
snp_data <- c(0, 0, 1, 2, 2, 9)
maf_value <- .maf(snp_data)
print(maf_value)
```

.simulateHalfSib *Simulate Half-Sibling Genotypes*

Description

This function simulates genotypes for a set of half-siblings based on specified parameters, including the number of individuals, the number of SNPs, recombination boundaries, and the type of data to return. It generates a sire genotype, maternal half-sib genotypes, and combines these to simulate offspring genotypes, optionally returning phased genotypes based on recombination events.

Usage

```
.simulateHalfSib(
  numInd = 40,
  numSNP = 10000,
  recbound = 0:6,
  type = "genotype"
)
```

Arguments

numInd Integer, the number of half-siblings to simulate.

numSNP Integer, the number of SNPs to simulate for each individual.

recbound Numeric vector, specifying the range of possible recombination events to simulate.

type Character string, specifying the type of data to return: "genotype" for genotypic data or any other string for phased genotypic data.

Value

Depending on the type parameter, this function returns a matrix of simulated genotypic data for half-siblings. If type is "genotype", it returns unphased genotypic data; otherwise, it returns phased genotypic data.

Examples

```
sim_genotypes <- .simulateHalfSib(numInd = 40, numSNP = 10000, recbound = 0:6, type = "genotype")
dim(sim_genotypes) # Should return 40 rows (individuals) and 100 columns (SNPs)
```

aio

*All-in-one Phasing***Description**

Phasing of a half-sib family group.

Usage

```
aio(genotypeMatrix, bmh_forwardVectorSize = 30, bmh_excludeFP = TRUE,
    bmh_nsap = 3, output = "phase")
```

Arguments

genotypeMatrix	matrix half-sib genotypes (one half-sib per row, with SNP ordered by mapping position in the columns. Data should be numeric. Use 0, 1 and 2 respectively for AA, AB and BB. Use 9 for missing data)
bmh_forwardVectorSize	integer number of heterozygous sites used to validate recombination events or check for genotyping errors
bmh_excludeFP	logical excludes SNPs that may cause heterozygous sites in the sire due to genotyping errors or map errors
bmh_nsap	integer number of SNP per block to validate recombinations
output	character if equal to the phase the 'aio' will only return the phasing results

Details

This function calls the [bmh](#), [ssp](#) and [phf](#) functions.

Value

Returns a list of matrices. The first element (phasedHalfSibs) is a matrix with two rows (phased haplotypes) per individual (first paternal and second maternal). Data in format 0 (A), 1 (B) and 9 (unphased or missing). The second (sireHaplotype) and third (blockStructure) elements are the same as the output of [ssp](#) and [bmh](#).

Note

Only this function needs to be called to phase a half-sib family. The genotype's matrix must contain individuals from only one half-sib family and one ordered chromosome.

See Also

[bmh](#), [ssp](#) and [phf](#)

Examples

```
genotype <- matrix(c(      # Define a Half-sib Genotype Matrix
  2,1,0,                  # Individual 1
  2,0,0,                  # Individual 2
  0,0,2,                  # Individual 3
), byrow = TRUE, ncol = 3) # There are 3 individuals with three SNPs

aio(genotype)             # The genotypes must include only one half-sib family and one chromosome
```

 bmh

Block Partitioning

Description

Identifies the block structure (chromosome segments) in the half-sib family that each individual inherited from its sire.

Usage

```
bmh(GenotypeMatrix, forwardVectorSize = 30, excludeFP = TRUE, nsap = 3)
```

Arguments

GenotypeMatrix	matrix half-sib genotypes (one half-sib per row, with SNP ordered by mapping position in the columns. Data should be numeric. Use 0, 1 and 2 respectively for AA, AB and BB. Use 9 for missing data)
forwardVectorSize	integer number of heterozygous sites used to validate recombination events or check for genotyping errors (50k -> 30, 700k -> 120)
excludeFP	logical excludes SNPs that may cause heterozygous sites in the sire due to genotyping errors or map errors
nsap	integer number of SNP per block to validate recombinations (50k -> 3, 700k -> 10)

Value

Returns a matrix of the blocking structure that contains 1s, 2s and 0s. 1s and 2s are the two sire strands. The choice of strand is arbitrary for each chromosome and not consistent across chromosomes. 0s indicate regions of unknown origin.

Note

The genotype's matrix must contain individuals from only one half-sib family and one ordered chromosome.

See Also

[ssp](#), [phf](#), [aio](#) and [imageplot](#)

Examples

```
genotype <- matrix(c(
  0,2,1,1,1,
  2,0,1,2,2,
  2,2,1,0,2,
  2,2,1,1,1,
  0,0,2,1,0), ncol = 5, byrow = TRUE)

(result <- bmh(genotype))
```

 co

Crossover Detection

Description

Detect all possible crossover events.

Usage

```
co(genotypeMatrix)
```

Arguments

`genotypeMatrix` matrix half-sib genotypes (one half-sib per row, with SNP ordered by mapping position in the columns. Data should be numeric. Use 0, 1 and 2 for respectively AA, AB and BB. Use 9 for missing data)

Value

Returns a matrix with the number of crossover events for each site.

Examples

```
genotype <- matrix(c(          # Define a Half-sib Genotype Matrix
  2,1,0,                      # Individual 1
  2,0,2,                      # Individual 2
  0,0,2,                      # Individual 3
), byrow = TRUE, ncol = 3)   # There are 3 individuals with three SNPs
```



```
co(genotype)
```

cs

Chromosome Splitter

Description

This function splits the genotypes list generated by [hss](#) into the different chromosomes based on a map file and orders SNP based on chromosomal position.

Usage

```
cs(halfsib, mapPath, separator = " ")
```

Arguments

halfsib	list list with matrices of half-sib genotypes, one family per list item
mapPath	character path to the map file (column 1 -> SNP names, column 2 -> chromosome name and column 3 -> SNP position in base pairs) or, alternatively, the name of a dataframe with the mapping information (in the same format)
separator	character separator character used in the the map file

Details

The map file should include only the chromosomes that will be analyzed. For example, the Y and X chromosomes should be excluded (and others optionally). Names of each element in the list can be used for further categorization. The header must be "Name Chr Position".

Value

Returns a list of matrices, the number of elements in this list is the number of half-sib families multiplied by the number of chromosomes.

Examples

```
# Please run demo(hsphase)
```

 genotypes

Example of Genotype Data Set

Description

This data set serves as an example of a genotype matrix intended for use with the hspbase package.

Usage

```
data(genotypes)
```

Format

The data set is a genotype matrix with specific structure, including:

- **Columns:** Represent Single Nucleotide Polymorphisms (SNPs). Each column corresponds to a specific SNP.
- **Rows:** Represent individual animals. Each row corresponds to the genotypic data for a single animal across various SNPs.

 hbp

Haplotype Blocks of Phased Data

Description

Creates a blocking structure matrix of the half-sib family based on phased data of the sire and half-sib family.

Usage

```
hbp(PhasedGenotypeMatrix, PhasedSireGenotype, strand = "auto")
```

Arguments

PhasedGenotypeMatrix

matrix haplotypes for a half-sib family (two rows per individual)

PhasedSireGenotype

matrix haplotypes of sire

strand

character method for identification of paternal strand (1 and 2 for strand one and two of the offsprings)

Value

Returns a matrix where 3 or 4 stands for the SNP originating in, respectively, strands 1 and 2. 0 indicates that the source strand for the SNP is unknown.

Note

The input matrices must only contain individuals from one half-sib family and one ordered chromosome. The strand option should be set to "auto" (default value).

See Also

[aio](#), [ssp](#)

Examples

```
sire <- matrix(c(
  0,0,0,0,0,1,      # Haplotype one of the sire
  0,1,1,1,1,0      # Haplotype two of the sire
), byrow = TRUE, ncol = 6)

haplotypeHalfsib <- matrix(c(
  1,0,1,1,1,1,      # Individual one, haplotype one
  0,1,0,0,0,0,      # Individual one, haplotype two
  0,1,1,0,1,1,      # Individual two, haplotype one
  1,0,0,1,0,0,      # Individual two, haplotype two
), byrow = TRUE, ncol = 6) # 0s and 1s are allele a and b

hbp(haplotypeHalfsib, sire)
```

hh *Heatmap of Half-sibs*

Description

The hh function creates a heatmap of the half-sib families using the matrix of opposing homozygotes.

Usage

```
hh(oh, inferredPedigree, realPedigree, pedOnly = TRUE)
```

Arguments

```
oh          matrix Opposing homozygotes matrix (output of ohg)
inferredPedigree
              matrix inferred pedigree (output of rpoh)
realPedigree matrix original pedigree
pedOnly     logical Consider only individuals that are exist in the real pedigree
```

Value

Returns the heatmap of the matrix of opposing homozygotes with sidebars colour coded by sires from the inferred and original pedigree.

Author(s)

The function uses the colour generated by *getcol* function in the *made4* package (Aedin Culhane).

See Also

[ohg](#) and [rpoh](#)

Examples

```
c1h1 <- .simulateHalfSib(numInd = 62, numSNP = 5000)
c1h2 <- .simulateHalfSib(numInd = 38, numSNP = 5000)
Genotype <- rbind(c1h1, c1h2)
oh <- ohg(Genotype) # creating the Opposing Homozygote matrix
hh(oh)
```

hss

Half-sib Family Splitter

Description

Splits the dataset into half-sib family groups based on a pedigree.

Usage

```
hss(pedigree, genotype, check = TRUE)
```

Arguments

pedigree	matrix the pedigree matrix should contain at least two columns, the first column with the half-sib IDs and the second column with the sires IDs
genotype	matrix genotype matrix with SNP ordered by mapping position in the columns. Data should be numeric. Use 0, 1 and 2 respectively for AA, AB and BB. Use 9 for missing data
check	logical check the genotype file for the possible errors

Details

Only half-sib groups that have more than 3 individuals will be returned.

Value

Returns a list of numeric matrices, each matrix is a half-sib family.

Note

Pedigree must have at least two columns with sample ids (Column 1) and sire ids (Column 2).

Examples

```
# Please run demo(hsphase)
```

imageplot	<i>Image Plot of Blocking Structure</i>
-----------	---

Description

Create an imageplot of the blocking structure.

Usage

```
imageplot(x, title, rv = FALSE, ...)
```

Arguments

x	matrix blocking structure (output of bmh or hbp functions)
title	character title of imageplot
rv	logical reverse the colour
...	Can be used to set xLabels and yLabels axis .

Details

White indicates regions of unknown origin, red and blue correspond to the two sire strands.

Author(s)

This is a modified version of a function written by Chris Seidel.
http://www.phaget4.org/R/image_matrix.html

See Also

[bmh](#) and [aio](#)

Examples

```
genotype <- matrix(c(
  0,2,1,1,1,
  2,0,1,2,2,
  2,2,1,0,2,
  2,2,1,1,1,
  0,0,2,1,0), ncol = 5, byrow = TRUE) # each row contains the SNP of individuals
imageplot(bmh(genotype))
```

impute	<i>Impute of Low Density SNP Marker to High Density (Paternal Strand)</i>
--------	---

Description

Impute the paternal strand from low density to high density utilising high density sire haplotype.

Usage

```
impute(halfsib_genotype_ld, sire_hd, bmh_forwardVectorSize = 30,  
       bmh_excludeFP = TRUE, bmh_nsap = 3)
```

Arguments

halfsib_genotype_ld	matrix half-sib genotypes with low density marker (one half-sib per row, with SNP ordered by mapping position in the columns. Data should be numeric. Use 0, 1 and 2 respectively for AA, AB and BB. Use 9 for missing data)
sire_hd	matrix haplotype of sire (this parameter can be sequence data or any phased sire - the matrix should have rownames which are the sample IDs and colnames which are the SNP names)
bmh_forwardVectorSize	integer number of heterozygous sites used to validate recombination events or check for genotyping errors
bmh_excludeFP	logical exclude SNPs that may cause heterozygous sites in the sire due to genotyping errors or map errors
bmh_nsap	integer number of SNPs per block

Value

Return an imputed half-sib matrix.

See Also

[bmh](#), [ssp](#) and [phf](#)

 map

Example Map File for Genetic Data

Description

This data set is an example of a map file used within the `hsphase` package to demonstrate the mapping of SNPs to their respective locations on chromosomes.

Usage

```
data(map)
```

Format

The data set is formatted as a data frame with the following columns, providing essential information about each SNP:

- **Name:** The unique identifier or name of the SNP.
- **Chr:** The chromosome on which the SNP is located.
- **Position:** The position of the SNP on the chromosome, expressed in base pairs.

 ohd

Opposing Homozygote Detection

Description

Counts the number of opposing homozygotes for each animal that caused a heterozygous site in the sire.

Usage

```
ohd(genotypeMatrix, unique_check = FALSE, SNPs = 6000)
```

Arguments

<code>genotypeMatrix</code>	matrix half-sib genotypes (one half-sib per row, with SNP ordered by mapping position in the columns. Data should be numeric. Use 0, 1 and 2 respectively for AA, AB and BB. Use 9 for missing data)
<code>unique_check</code>	logical check if samples uniquely originate an opposing homozygote at a locus
<code>SNPs</code>	integer number of SNP to use

Value

Returns a vector with the number of heterozygous sites that each sample caused.

Note

This function can be used to identify pedigree errors; i.e., the outliers.

Author(s)

This method is suggested by Bruce Tier <btier@une.edu.au> to identify pedigree errors.

Examples

```
genotype <- matrix(c(
  2,1,0,
  2,0,0,
  0,0,2
), byrow = TRUE, ncol = 3)

ohd(genotype)
```

ohg

Matrix of Opposing Homozygotes

Description

Creates a matrix of opposing homozygotes from the genotype matrix.

Usage

```
ohg(genotypeMatrix)
```

Arguments

genotypeMatrix matrix genotype (Data should be numeric. Use 0, 1 and 2 respectively for AA, AB and BB. Use 9 for missing data)

Value

Returns a square matrix (sample X sample) with the pairwise counts of opposing homozygotes.

Note

This function can be slow with a large data set. The fast version of this function will be available after publish of the related manuscript.

Author(s)

Ferdosi, M. H., & Boerner, V. (2014). A fast method for evaluating opposing homozygosity in large SNP data sets. *Livestock Science*.

See Also[rpoh](#)**Examples**

```
genotype <- matrix(c(
  2,1,0,
  2,0,0,
  0,0,2
), byrow = TRUE, ncol = 3)

ohg(genotype)
```

ohplot*Opposing Homozygotes Plot*

Description

Plot the sorted vectorized matrix of Opposing Homozygotes.

Usage

```
ohplot(oh, genotype, pedigree, check = FALSE)
```

Arguments

oh	integer Opposing homozygotes matrix (Output of ohg)
genotype	matrix genotype of one chromosome (data should be numeric. Use 0, 1 and 2 for respectively AA, AB and BB. Use 9 for missing data)
pedigree	matrix the pedigree matrix should contain at least two columns, the first column with the half-sib IDs and the second column with the sires IDs. This argument is optional.
check	logical check the genotype file for the possible errors

Details

The cut off line shows the edge of most different groups.

See Also[ohg](#) and [rpoh](#)

Examples

```

set.seed(100)
chr <- list()
sire <- list()
set.seed(1)
chr <- list()
for(i in 1:5)
{
chr[[i]] <- .simulateHalfsib(numInd = 20, numSNP = 5000, recbound = 1:10)
sire[[i]] <- ssp(bmh(chr[[i]]), chr[[i]])
sire[[i]] <- sire[[i]][1,] + sire[[i]][2,]
sire[[i]][sire[[i]] == 18] <- 9
}

Genotype <- do.call(rbind, chr)
rownames(Genotype) <- 6:(nrow(Genotype) + 5)
sire <- do.call(rbind, sire)
rownames(sire) <- 1:5
Genotype <- rbind(sire, Genotype)
oh <- ohg(Genotype) # creating the Opposing Homozygote matrix
pedigree <- as.matrix(data.frame(c(1:5, 6:(nrow(Genotype))),
rep = c(rep(0,5), rep(1:5, rep(20,5))))))
ohplot(oh, Genotype, pedigree, check = TRUE)

```

para

*Parallel Analysis of Data***Description**

This function uses the list of matrices (the output of `cs`) and runs one of the options, on each element of the list, in parallel.

Usage

```
para(halfsibs, cpus = 1, option = "bmh", type = "SOCK", bmh_forwardVectorSize = 30,
    bmh_excludeFP = TRUE, bmh_nsap = 3, pmMethod = "constant")
```

Arguments

halfsibs	list list of matrices of half-sibs (can be generated with hss and cs functions)
cpus	numeric number of CPUs (thread)
option	character type of analysis
type	character type of cluster for parallel analysis
bmh_forwardVectorSize	integer number of heterozygous sites used to validate recombination events or check for genotyping errors

<code>bmh_excludeFP</code>	logical exclude SNPs that may cause heterozygous sites in the sire due to genotyping errors or map errors
<code>bmh_nsap</code>	integer number of SNPs per block
<code>pmMethod</code>	character method for creating the recombination matrix

Details

Type of analysis can be `bmh`, `ssp`, `aio`, `pm`, or `rec` (refer to `pm`, `rplot` and vignette for more information about `rec`).

Value

Returns a list of matrices with the results (formats specific to the option selected).

Examples

```
# Please run demo(hsphase)
```

pedigree

Example Pedigree Data Set

Description

This dataset provides an example of a pedigree, specifically designed for use with the `hsphase` package.

Usage

```
data(pedigree)
```

Format

The dataset is structured as a data frame with detailed familial relationships, including:

- **First Column:** Identifiers for half-sibs.
- **Second Column:** Identifiers for sires.

pedigreeNaming	<i>Fix Pedigree Errors</i>
----------------	----------------------------

Description

Tries to link the inferred pedigree from [rpoh](#) with the sire IDs in the original pedigree and fix pedigree errors.

Usage

```
pedigreeNaming(inferredPedigree, realPedigree)
```

Arguments

```
inferredPedigree      matrix inferred pedigree (output of rpoh )
realPedigree          matrix original pedigree
```

Details

This function calls the [bmh](#) and [recombinations](#) functions to count the number of recombinations in each half-sib group.

Value

Returns the inferred pedigree with the best fit to the sire names used in the original pedigree file.

See Also

[rpoh](#) and [ohg](#)

Examples

```
# Please run demo(hsphase)
```

phf	<i>Half-Sib Family Phasing</i>
-----	--------------------------------

Description

Phases the half-sib family by using the blocking structure and imputed sire matrices.

Usage

```
phf(GenotypeMatrix, blockMatrix, sirePhasedMatrix)
```

Arguments

GenotypeMatrix matrix half-sib genotypes (one half-sib per row, with SNP ordered by mapping position in the columns. Data should be numeric. Use 0, 1 and 2 respectively for AA, AB and BB. Use 9 for missing data)

blockMatrix matrix blocking structure (output of [bmh](#))

sirePhasedMatrix matrix imputed sire (output of [ssp](#))

Value

Returns a matrix that contains the phased parental haplotypes of the half-sibs. It uses 1, 0 and 9 for A, B and missing.

Note

The genotype matrix must only contain individuals from one half-sib family and one ordered chromosome. This function is used by the [aio](#) function for complete phasing of a half-sib group.

See Also

[aio](#)

Examples

```
genotype <- matrix(c(
  2,1,0,
  2,0,0,
  0,0,2), byrow = TRUE, ncol = 3)
block <- bmh(genotype)
phf(genotype, block, ssp(block, genotype))
```

pm

Probability Matrix

Description

Creates a recombination matrix based on the blocking structure.

Usage

```
pm(blockMatrix, method = "constant")
```

Arguments

blockMatrix matrix blocking structure (Output of [bmh](#))

method character method for creating the recombination matrix

Details

This function finds the recombination between two consecutive sites, and marks the recombination site with a 1; if there are unknown sites between two blocks it will also mark these sites with a 1 (constant method) or 1 divided by number of unknown site (relative method).

Examples

```
genotype <- matrix(c(
  0,2,0,1,0,
  2,0,1,2,2,
  2,2,1,0,2,
  2,2,1,1,1,
  0,0,2,1,0), ncol = 5, byrow = TRUE) # each row contains the SNP of individuals

(result <- bmh(genotype))
pm(result)
```

pogc

Parent Offspring Group Constructor

Description

Assign offsprings to the parents.

Usage

```
pogc(oh, genotypeError)
```

Arguments

oh integer opposing homozygotes matrix (Output of [ohg](#))
genotypeError integer number of genotypeing error allowed in the oh matrix

Value

Return a data frame with two columns. The first column is the animal ID and the second column is the parent ID.

See Also

[ohg](#), [hss](#) and [rpoh](#)

Examples

```

set.seed(100)
chr <- list()
sire <- list()
set.seed(1)
chr <- list()
for(i in 1:5)
{
chr[[i]] <- .simulateHalfsib(numInd = 20, numSNP = 5000, recbound = 1:10)
sire[[i]] <- ssp(bmh(chr[[i]]), chr[[i]])
sire[[i]] <- sire[[i]][1,] + sire[[i]][2,]
sire[[i]][sire[[i]] == 18] <- 9
}

Genotype <- do.call(rbind, chr)
rownames(Genotype) <- 6:(nrow(Genotype) + 5)
sire <- do.call(rbind, sire)
rownames(sire) <- 1:5
Genotype <- rbind(sire, Genotype)
oh <- ohg(Genotype) # creating the Opposing Homozygote matrix
pogc(oh, 5)

```

readGenotype

Read and Check the Genotype File

Description

This function reads and checks genotype files.

Usage

```
readGenotype(genotypePath, separatorGenotype = " ", check = TRUE)
```

Arguments

genotypePath	character genotype path (animals (rows) and SNP (columns), SNP should be coded as 0, 1 and 2 for respectively AA, AB and BB. Use 9 for missing data. please refer to vignette for more information)
separatorGenotype	character separator character for genotype
check	logical check the genotype file for possible errors

Value

Returns the genotype matrix.

Note

Please refer to vignette for more information.

Examples

```
# A comprehensive demo and example dataset is available from  
# http://www-personal.une.edu.au/~cgondro2/hsphase.html
```

recombinations	<i>Recombination Number</i>
----------------	-----------------------------

Description

Counts the number of recombinations for each individual.

Usage

```
recombinations(blockMatrix)
```

Arguments

blockMatrix matrix block structure (Output of [bmh](#))

Value

Returns a vector of recombinations. The number of elements in this vector is equal to the number of individuals, i.e. each element holds the number of recombinations identified for each sample.

See Also

[bmh](#)

Examples

```
genotype <- matrix(c(  
  2,1,0,0,  
  2,0,2,2,  
  0,0,2,2,  
  0,2,0,0  
), byrow = TRUE, ncol = 4)  
  
recombinations(bmh(genotype))
```

rplot	<i>Recombination Plot</i>
-------	---------------------------

Description

This function creates a plot which shows the sum of all recombination events across a half-sib family.

Usage

```
rplot(x, distance, start = 1, end = ncol(x), maximum = 100,  
      overwrite = FALSE, method = "constant")
```

Arguments

x	matrix of half-sib genotypes (one half-sib per row, with SNP ordered by mapping position in the columns. Data should be numeric. Use 0, 1 and 2 for respectively AA, AB and BB. Use 9 for missing data).
distance	integer of physical distances between markers
start	integer first marker selected for the plot
end	integer last marker selected for the plot
maximum	integer maximum number of recombinations to show (higher recombination rates will be omitted from the plot)
overwrite	logical draw a diagram over the current diagram (default FALSE)
method	character please refer to the pm document

Examples

```
genotype <- matrix(c(  
  0,2,0,1,0,  
  2,0,1,2,2,  
  2,2,1,0,2,  
  2,2,1,1,1,  
  0,0,2,1,0), ncol = 5, byrow = TRUE) # each row contains the SNP of individuals  
  
rplot(genotype, c(1,2,3,4,8))
```

rpoh *Reconstruct Pedigree Based on Matrix of Opposing Homozygotes*

Description

Reconstructs a half-sib pedigree based on a matrix of opposing homozygotes.

Usage

```
rpoh(genotypeMatrix, oh, forwardVectorSize = 30, excludeFP = TRUE, nsap = 3,
maxRec = 15, intercept = 26.3415, coefficient = 77.3171, snpnooh, method, maxsnpnooh)
```

Arguments

genotypeMatrix	matrix genotype of one chromosome (data should be numeric. Use 0, 1 and 2 for respectively AA, AB and BB. Use 9 for missing data)
oh	integer Opposing homozygotes matrix (Output of ohg)
forwardVectorSize	integer number of heterozygous sites used to validate recombination events or check for genotyping errors
excludeFP	logical excludes SNPs that may cause heterozygous sites in the sire due to genotyping errors or map errors
nsap	integer number of SNP per block to validate recombinations
maxRec	integer maximum number of expected recombinations per individual
intercept	integer intercept of fitted model
coefficient	integer coefficient of fitted model
snpnooh	integer number of SNPs used to create <i>oh</i> matrix (this number must be divided by 1000)
method	character pedigree reconstruction method
maxsnpnooh	numeric the maximum number of allowing opposing homozygote in a half-sib family

Details

Four methods *simple*, *recombinations*, *calus* and *manual* can be utilized to reconstruct the pedigree.

The following examples show the arguments require for each method.

```
pedigree1 <- rpoh(oh = oh, snpnooh = 732, method = "simple")
pedigree2 <- rpoh(genotypeMatrix = genotypeChr1, oh = ohg(genotype), maxRec = 10, method =
"recombinations")
pedigree3 <- rpoh(genotypeMatrix = genotype, oh = oh, method = "calus")
pedigree4 <- rpoh(oh = oh, maxsnpnooh = 31662, method = "manual")
```

Value

Returns a data frame with two columns, the first column is animals' ID and the second column is sire identifiers (randomly generated).

Note

Method can be *recombinations*, *simple*, *calus* or *manual*. Please refer to vignette for more information.

The sire genotype should be removed before using this function utilizing [pogc](#) function.

See Also

[bmh](#) and [recombinations](#)

Examples

```
# Please run demo(hsphase)
```

ssp

Sire Imputation and Phasing

Description

Infer (impute) and phase sire's genotype based on the block structure matrix (recombination blocks) and homozygous sites of the half-sib genotype matrix.

Usage

```
ssp(blockMatrix, genotypeMatrix)
```

Arguments

`blockMatrix` matrix block structure (Output of [bmh](#))

`genotypeMatrix` matrix half-sibs genotype (each row includes the SNP of individuals, 0, 1 and 2 for respectively AA, AB and BB. Use 9 for missing data)

Value

Returns a matrix (Imputed Sire) with two rows one for each haplotype of the sire (columns are SNP in the order of the genotype matrix). Alleles are coded as 0 (A) and 1 (B). Alleles that could not be imputed are coded as 9.

See Also

[phf](#), [aio](#) and [imageplot](#)

Examples

```
genotype <- matrix(c(
  0,2,1,1,1,
  2,0,1,2,2,
  2,2,1,0,2,
  2,2,1,1,1,
  0,0,2,1,0), ncol = 5, byrow = TRUE) # each row contains the SNP of individuals

(result <- ssp(bmh(genotype), genotype))
```

Index

- * **Chromosome**
 - cs, 9
- * **High_Density**
 - impute, 14
- * **Low_Density**
 - impute, 14
- * **Opposing_Homozygotes**
 - ohplot, 17
- * **SNP**
 - hbp, 10
- * **Splitter**
 - cs, 9
- * **block**
 - hbp, 10
 - hsphase-package, 2
 - imageplot, 13
 - recombinations, 24
- * **crossover**
 - co, 8
- * **datasets**
 - genotypes, 10
 - map, 15
 - pedigree, 19
- * **error**
 - recombinations, 24
- * **genoytpe**
 - ohg, 16
- * **half-sib**
 - hsphase-package, 2
- * **halfsib**
 - bmh, 7
 - rplot, 25
 - ssp, 27
- * **haplotype**
 - aio, 6
 - phf, 20
- * **heatmap**
 - hh, 11
- * **image**
 - imageplot, 13
- * **impute**
 - impute, 14
- * **inference**
 - aio, 6
 - phf, 20
- * **infer**
 - pedigreeNaming, 20
- * **opposing homozygote**
 - ohd, 15
- * **opposing-homozygote**
 - ohg, 16
 - para, 18
 - rpoh, 26
- * **parentage**
 - pogc, 22
- * **pedigree**
 - hh, 11
 - pedigreeNaming, 20
 - pogc, 22
 - recombinations, 24
 - rpoh, 26
- * **phase**
 - aio, 6
 - bmh, 7
 - hbp, 10
 - hsphase-package, 2
 - phf, 20
 - rplot, 25
 - ssp, 27
- * **plot**
 - ohplot, 17
- * **recombination**
 - recombinations, 24
 - rplot, 25
- * **reconstruction**
 - aio, 6
 - phf, 20
 - rpoh, 26

- * **sire inference**
 - hsphase-package, 2
- * **snp**
 - aio, 6
 - bmh, 7
 - ohg, 16
 - phf, 20
 - rplot, 25
 - rpoh, 26
 - ssp, 27
 - .fastdist, 4
 - .maf, 4
 - .simulateHalfsib, 5
- aio, 2, 6, 8, 11, 13, 19, 21, 27
- axis, 13
- bmh, 2, 6, 7, 7, 13, 14, 19–21, 24, 27
- co, 8
- cs, 3, 9, 18
- genotypes, 10
- hbp, 10, 13
- hh, 11
- hsphase (hsphase-package), 2
- hsphase-package, 2
- hss, 3, 9, 12, 18, 22
- imageplot, 2, 8, 13, 27
- impute, 14
- map, 15
- ohd, 15
- ohg, 12, 16, 17, 20, 22, 26
- ohplot, 17
- para, 3, 18
- pedigree, 19
- pedigreeNaming, 20
- phf, 6–8, 14, 20, 27
- pm, 19, 21, 25
- pogc, 22, 27
- readGenotype, 23
- recombinations, 20, 24, 27
- rplot, 19, 25
- rpoh, 3, 12, 17, 20, 22, 26
- ssp, 2, 6–8, 11, 14, 19, 21, 27