Package 'HAPTRACE'

September 10, 2026

Title Haplotype-Based Tracking of Admixed Population for Breed Composition Estimation

Version 0.1.2

Maintainer Shweta Sahoo <queryhap01@gmail.com>

Description Simulate populations and track haplotypes over generations to evaluate population admixture. The 'HAPTRACE' supports customisable population parameters, including size, number of markers, mutation rates and recombination.

License GPL (>= 3)

Encoding UTF-8

Imports data.table, gdata, hsphase (>= 3.0.1), Rcpp, RcppArmadillo

Suggests knitr, rmarkdown

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

Config/roxygen2/version 8.1.0

NeedsCompilation yes

Author Shweta Sahoo [aut, cre] (ORCID: <https://orcid.org/0000-0002-5154-5719>), Sara de las Heras-Saldana [aut] (ORCID: <https://orcid.org/0000-0002-8665-6160>), Julius H. J. van der Werf [aut] (ORCID: <https://orcid.org/0000-0003-2512-1696>), Mohammad H. Ferdosi [aut] (ORCID: <https://orcid.org/0000-0001-5385-4913>)

Repository CRAN

Date/Publication 2026-09-10 04:10:02 UTC

Contents

Admixplot 2
AlleleFreq 3
BreedEff_Haplo 4

BreedPopCross 4
BreedPopSelect 7
CalQTLeffect 9
Coded_to_Haplo 10
DamSample 10
Ext_Pheno 11
GenBreedPop 12
generateHP 14
generateMAP 15
GenOnePopulation 16
MakeGeno 18
mapQTLfile 19
MatingPop 19
MutationRate 20
Ped_Pheno 21
plinkped 22
plotHaplo 22
PopSelect 23
PopulationID 25
QMSim_PlinkMAP 26
QTLeffects 26
QTL_densityplot 27
Random_gametes 28
ReadQTL 29
read_qmsim_geno 29
Recode_Haplotype 31
RecombinationPoint 31
recProbplot 32
Reformat_BLUPF90 33
Rescale 34
SireSample 35
TBV_Haplo 35
trueBreedComp 36
TwoPopCross 37
Var_PhenoRecords 39

Index **41**

`Admixplot` *Plot for True Breed Composition*

Description

Generates barplot of true breed composition of the animals with haplotype information coded according to the breed of origin.

Usage

```
Admixplot(trueBC, color, legendlabels)
```

Arguments

trueBC	True breed composition estimated by the haplotype information coded according to the breed of origin.
color	Colors specifying the number of populations contributing to the true breed composition.
legendlabels	Population label to add as a legend to the plot.

Value

Returns an admixture plot of animals where x axis shows number of animals and y axis shows the proportion of different breeds in different colors.

See Also

trueBreedComp()

Examples

```
popAdm <- matrix(c(-1, 2, 3, -4, 2,
1,-1,2,2,3,
1,2,-3,-4,-5,
2,-3,2,4,-5), byrow = TRUE, nrow = 4, ncol = 5)
TBC <- trueBreedComp(Haplotype = popAdm)
Admixplot(trueBC = TBC, color = c("#FFC107", "#2E7D32", "#FB8072", "#386CB0", "#4d4a49"),
legendlabels = c("A", "B", "C", "D", "E"))
```

AlleleFreq *Calculate Allele Frequency*

Description

Function to calculate allele frequency at each locus of a population.

Usage

```
AlleleFreq(df)
```

Arguments

df	A matrix consisting of a genotype information of the animals in (0,1,2) format.

Value

Returns the value of allele frequency at each locus.

Examples

```
popC <- matrix(data = sample(c(0,1,2), 60, replace = TRUE), nrow = 6, ncol = 10)
allelefreq <- AlleleFreq(df = popC)
```

`BreedEff_Haplo` *True Breeding Value (TBV) estimation according to breed of origin*

Description

Function to estimate TBV for animals having different SNP effects according to breed of origin.

Usage

```
BreedEff_Haplo(Haplotype, Breed_Effect)
```

Arguments

`Haplotype`	A matrix comprising a coded haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
`Breed_Effect`	A matrix comprising the breed effect defined according to the breed of origin such that row depicts the breeds and columns depicts the SNP effects.

Value

Returns a matrix of coded Haplotype with TBV estimated according to the breed of origin.

Examples

```
pop_BH <- matrix(sample(c(-1,1,-2,2,-3,3), 16, replace = TRUE), nrow = 4, byrow = TRUE)
Breed_Eff <- matrix(runif(4 * 4, 0, 1), nrow = 4, byrow = TRUE)
Breed_Hap <- BreedEff_Haplo(Haplotype = pop_BH, Breed_Effect = Breed_Eff)
```

`BreedPopCross` *Function for crossing two population with breed effects according to breed of origin*

Description

Function to cross two population and estimation of TBV with breed effect according to the breed of origin.

Usage

```
BreedPopCross(
  map = map,
  populationSim_A,
  populationSim_B,
  nSireA = 54,
  nDamA = 1000,
  nSireB = 26,
  nDamB = 1000,
  Sire_From,
  mutationRate = 2.5 * 10^-5,
  SelType,
  h2 = 0.3,
  trait_mean,
  VarE_PopA,
  VarE_PopB,
  Breed_Effect_A,
  Breed_Effect_B,
  IndStartVal = 1,
  prefixID = "Cr",
  nProgeny = 1000,
  recL,
  nChr,
  recProb,
  minLength
)
```

Arguments

map	Map file for the population comprising of marker ID, chromosome number and marker position.
populationSim_A	Coded haplotype information of the population A where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
populationSim_B	Coded haplotype information of the population B where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
nSireA	Number of sires to be selected as parents for the simulation from population A.
nDamA	Number of dams to be selected as parents for the simulation from population A.
nSireB	Number of sires to be selected as parents for the simulation from population B.
nDamB	Number of dams to be selected as parents for the simulation from population B.
Sire_From	The population from which sires should being selected for cross population. If sire should be selected from population A, then use "A" and if sire should be selected from population B, then use "B".

mutationRate	Mutation rate to added to the simulated population.
SelType	Selection Type. It can be based on phenotype, random or true breeding value. The default selection type is "random". For phenotypic selection, use "Pheno" and for true breeding value, use "TBV".
h2	Heritability of the trait to be simulated.
trait_mean	Mean of the trait to be simulated.
VarE_PopA	Error Variance obtained after rescaling the QTL effects for PopA.
VarE_PopB	Error Variance obtained after rescaling the QTL effects for PopB.
Breed_Effect_A	A matrix comprising breed effect according to the breed of origin for population A.
Breed_Effect_B	A matrix comprising breed effect according to the breed of origin for population B.
IndStartVal	Starting value for the animal ID.
prefixID	Prefix to be added to the ID of the animal.
nProgeny	Number of progeny to be simulated in the population.
recL	Average number of recombinations to be introduced per animal.
nChr	Number of haploid chromosomes (n) defined in the population.
recProb	A vector of recombination probability between the markers. Its length should be total number of markers minus 1. User can define the recombination probability between markers helping in simulating a desired population.
minLength	Minimum length between recombination points beyond which recombination is allowed. The default value is 0. If a user define minimum length, it will ensure that difference between two recombination points less than minimum length won't allow recombination.

Value

Returns a list of Haplotype information, ID of the simulated animals, population with TBV and phenotype, recombination point list of sire and dam.

See Also

BreedEff_Haplo(), BreedPopSelect()

Examples

```
crossmap <- generateMAP(nChr = 5, n_markers = 2, len_Chr = 20)
QTL_cross <- matrix(data = c(-0.001, 0.0003, 0, 0, 0, 0.0032, 0.0023, -0.0012,0, 0)
, nrow = 1, ncol = 10)
P1cross <- matrix(data = sample(c(-1, 1), 40, replace = TRUE), nrow = 4, ncol = 10)
P2cross <- matrix(data = sample(c(-2, 2), 40, replace = TRUE), nrow = 4, ncol = 10)
P1cross <- PopulationID(Population = P1cross, PopCode = "P1")
P2cross <- PopulationID(Population = P2cross, PopCode = "P2")
F2cross <- BreedPopCross(map = crossmap, populationSim_A = P1cross,
populationSim_B = P2cross, nSireA = 1, nDamA = 1, nSireB = 1, nDamB = 1,
Sire_From = "A",mutationRate = 2.5 * 10^-5, SelType = "TBV", h2 = 0.3,
```

```
Breed_Effect_A = QTL_cross, Breed_Effect_B = QTL_cross, IndStartVal = 1,
prefixID = "Cr", nProgeny = 10, recL = 1, nChr = 2, minLength = 0,
trait_mean = 5, VarE_PopA = 0.6, VarE_PopB = 0.6)
```

`BreedPopSelect` *Selecting Population with breed effect according to breed of origin*

Description

Function to simulate a population with different breed effect according to breed of origin.

Usage

```
BreedPopSelect(
  map,
  Haplotype,
  nSire,
  nDam,
  rate = 2.5 * 10^-5,
  nProgeny,
  SelType = "random",
  Breed_Effect,
  h2,
  trait_mean,
  VarE,
  recL,
  nChr,
  recProb,
  minLength
)
```

Arguments

`map`	Map file for the population comprising of marker ID, chromosome number and marker position.
`Haplotype`	A matrix with coded haplotype information of the population where rows are number of animals and columns are number of markers. In case of haplotype, two rows provide information on genomic material of an animal.
`nSire`	Number of sires to be used for simulation.
`nDam`	Number of dams to be used for simulation.
`rate`	Mutation rate to introduce mutation in the simulated population.
`nProgeny`	Number of progeny to be simulated in each generation.
`SelType`	A character value specifying the selection criteria of the simulated population. Selection criteria can be based on phenotype, random or true breeding value. The default selection type is "random". For phenotypic selection, use "Pheno" and for true breeding value, use "TBV".

Breed_Effect	A matrix comprising breed effect according to the breed of origin where rows contain information on SNP effects.
h2	A numeric value describing the heritability of the trait to be simulated ranging from 0 to 1.
trait_mean	A positive integer value describing mean of the phenotype of the trait to be simulated.
VarE	A positive integer value depicting error variance obtained after rescaling the QTL effects.
recL	A numeric value describing the average number of recombinations to be introduced per animal.
nChr	A numeric value describing the number of haploid chromosomes (N) defined in the simulated population.
recProb	A vector of recombination probability between the markers. Its length should be total number of markers minus 1. User can define the recombination probability between markers helping in simulating a desired population.
minLength	Minimum length between recombination points beyond which recombination is allowed. The default value is 0. If a user define minimum length, it will ensure that difference between two recombination points less than minimum length won't allow recombination.

Value

Returns selected sire and dams along with with their ID, recombination points of sire and dam, and population coded haplotype with phenotype and TBV.

See Also

BreedEff_Haplo()

Examples

```
PopS <- matrix(data = sample(c(-1,1, -2,2), 200, replace = TRUE),
nrow = 20, ncol = 10)
PopS_map <- generateMAP(2, 5, 5)
PopS_QTL <-  matrix(c(-0.001, 0.0003, 0, 0, 0, 0.0032, 0.0023, -0.0012,0, 0,
 -0.002, 0.004, 0, 0, 0.002, 0, 0.002, 0,0, -0.0012), byrow = 2, nrow = 2, ncol = 10)
Selpop <- BreedPopSelect(map = PopS_map, Haplotype = PopS, nSire = 2, nDam = 2, rate = 2.5 * 10^-5,
nProgeny = 6, SelType = "random", Breed_Effect = PopS_QTL, trait_mean = 5,VarE = 0.6,
h2 = 0.2, recL = 1, nChr = 2, minLength = 0)
```

`CalQTLeffect` *Calculate QTL effect from file generated from ReadQTL (For QMSim generated files)*

Description

This function requires the file with information on markerID, chromosome and marker information with prefix "lm_mrk_qtl", total number of SNP markers ,and QTL effects generated from ReadQTL function. It generates SNP effects of length of total number of markers along with QTLs.

Usage

```
CalQTLeffect(filename, nSNP, QTLfile)
```

Arguments

`filename`	Requires the name of the file with markerID, chr, and marker position.
`nSNP`	Total number of SNP markers.
`QTLfile`	QTL effects generated from ReadQTL function.

Value

Returns a vector of the SNP effects with length equals to the total number of SNP markers.

See Also

`ReadQTL()`

Examples

```
## Not run:
# This example is not executed since it needs large files.
popQM <- read_qmsim_geno("p1_mrk_qtl_001.txt")
QTL <- ReadQTL(filename = "effect_qtl_001.txt")
QTLeffect <- CalQTLeffect(filename = "lm_mrk_qtl_001.txt",
nSNP = ncol(popQM),QTLfile = QTL)

## End(Not run)
```

Coded_to_Haplo — *Recode Coded haplotypes according to breed of origin into 0 and 1 format*

Description

Function to recode the coded haplotype back to its original form.

Usage

```
Coded_to_Haplo(df)
```

Arguments

df	A matrix consisting of coded haplotype information of the animals according to the breed of origin.

Value

Returns a matrix consisting of a haplotype information in (0,1) format.

Examples

```
popB <- matrix(data = sample(c(-1,1), 60, replace = TRUE), nrow = 6, ncol = 10)
coded_haplo <- Coded_to_Haplo(df = popB)
```

DamSample — *Select Dams*

Description

Function to select the dams.

Usage

```
DamSample(Haplotype, nDam, nProgeny)
```

Arguments

Haplotype	Haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
nDam	Number of Dams to be selected as parents for the simulation.
nProgeny	Number of progeny to be simulated in the population.

Value

Return a list of dam IDs and haplotype information of the dam.

Examples

```
pop_J <- matrix(data = sample(c(0,1), 60, replace = TRUE), nrow = 6, ncol = 10)
pop_J_dam <- DamSample(Haplotype = pop_J, nDam = 1, nProgeny = 4)
```

Ext_Pheno *Extract TBV and Phenotype*

Description

Extract TBV and Phenotype

Usage

```
Ext_Pheno(df)
```

Arguments

df	A matrix comprising the haplotype information along with TBV and Phenotype. Ensure that rownames of the matrix is Animal ID

Value

A datafile comprising Animal ID, TBV and phenotype.

Examples

```
popEP <- matrix(data= sample(c(0,1), 16, replace= TRUE), nrow = 4, ncol = 4)
popEP <- PopulationID(Population = popEP, PopCode = "EP")
QTLEff <- c(0.2, 0.1, 0.3, 0.1)
TBV_haplotype2 <- TBV_Haplo(Haplotype = popEP, Effect = QTLEff)
Var_Ph_R <- Var_PhenoRecords(Haplotype = TBV_haplotype2, h2 = 0.3,
trait_mean = 20, VarE = 0.6)
Haplo_TBV_Ph <- Var_Ph_R$Haplo_pheno
Datafile <- Ext_Pheno(Haplo_TBV_Ph)
```

GenBreedPop *Generate Population with specific breed effect*

Description

Generate Population of a crossbred with breed effect specific to the breed of origin.

Usage

```
GenBreedPop(
  ngenerations = 5,
  map = map,
  populationSim,
  nSire = 54,
  nDam = 1000,
  mutationRate = 2.5 * 10^-5,
  SelType = "random",
  h2 = 0.3,
  trait_mean,
  VarE,
  Breed_Effect,
  recL,
  nChr,
  recProb,
  minLength,
  IndStartVal = 1,
  prefixID = "A",
  nProgeny = 1000
)
```

Arguments

ngenerations	Number of generations to be simulated.
map	Map file for the population comprising of marker ID, chromosome number and marker position.
populationSim	Coded haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
nSire	Number of sires to be selected as parents for the simulation.
nDam	Number of Dams to be selected as parents for the simulation.
mutationRate	Mutation rate to added to the simulated population.
SelType	Selection Type. It can be based on phenotype, random or true breeding value. The default selection type is "random". For phenotypic selection, use "Pheno" and for true breeding value, use "TBV".
h2	Heritability of the trait to be simulated.

trait_mean	Mean of the trait to be simulated.
VarE	Error Variance obtained after rescaling the QTL effects.
Breed_Effect	A matrix comprising breed effect according to the breed of origin where rows contain information on SNP effects.
recL	Average number of recombinations to be introduced per animal.
nChr	Number of haploid chromosomes (n) defined in the population.
recProb	A vector of recombination probability between the markers. Its length should be total number of markers minus 1. User can define the recombination probability between markers helping in simulating a desired population.
minLength	Minimum length between recombination points beyond which recombination is allowed. The default value is 0. If a user define minimum length, it will ensure that difference between two recombination points less than minimum length won't allow recombination.
IndStartVal	Starting value for the Animal ID of the simulated population.
prefixID	Prefix to be added to the ID of the animal.
nProgeny	Number of progeny to be simulated in the population.

Value

Returns a list of Haplotype information, pedigree of the simulated animals, population with TBV and phenotype, recombination point list of sire and dam.

See Also

BreedEff_Haplo(), BreedPopSelect()

Examples

```
PopGen <- matrix(data = sample(c(-1, 1), 200, replace = TRUE), nrow = 20, ncol = 10)
PopGen <- PopulationID(Population = PopGen, PopCode = "PG1")
PopGen_map <- generateMAP(nChr = 2, n_markers = 5, len_Chr = 5)
QTL_PopGen <- matrix(data = c(-0.001, 0.0003, 0, 0, 0,
0.0032, 0.0023, -0.0012,0, 0), nrow = 1, ncol = 10)
PopF1 <- GenBreedPop(ngenerations = 2, map = PopGen_map, populationSim = PopGen,
nSire = 1, nDam = 1,mutationRate = 2.5 * 10^-5, SelType = "TBV", h2 = 0.3, trait_mean = 5,
VarE = 0.6, Breed_Effect = QTL_PopGen, recL = 1, nChr = 2, minLength = 0,
IndStartVal = 1,prefixID = "A", nProgeny = 10)
```

generateHP *Generate Historical Population*

Description

Function to simulate historical population.

Usage

```
generateHP(
  n_generations,
  n_animals,
  map,
  nSire,
  nDam,
  nProgeny,
  mutationRate = 2.5 * 10^-5,
  SelType = "random",
  Effect,
  h2 = 0.3,
  trait_mean = 10,
  VarE,
  recL = 5,
  nChr,
  recProb,
  minLength = 0,
  IndStartVal = 1,
  prefixID = "H"
)
```

Arguments

n_generations	Number of generations to be simulated.
n_animals	Number of animals to be simulated.
map	Map file for the population.
nSire	Number of sires to be selected as parents for the simulation.
nDam	Number of Dams to be selected as parents for the simulation.
nProgeny	Number of progeny to be simulated in the population.
mutationRate	Mutation rate to added to the simulated population.
SelType	Selection Type. It can be based on phenotype, random or true breeding value. The default selection type is random. For phenotypic selection, use "Pheno" and for true breeding value, use "TBV".
Effect	SNP effect for calculation of TBV.
h2	Heritability of the trait to be simulated.

trait_mean	Mean of the trait to be simulated.
VarE	Error Variance obtained after rescaling the QTL effects.
recL	Average number of recombinations to be introduced per animal.
nChr	Number of haploid chromosomes (n) defined in the population.
recProb	A vector of recombination probability between the markers. Its length should be total number of markers minus 1. User can define the recombination probability between markers helping in simulating a desired population.
minLength	Minimum length between recombination points beyond which recombination is allowed. The default value is 0. If a user define minimum length, it will ensure that difference between two recombination points less than minimum length won't allow recombination.
IndStartVal	Starting value for the Animal ID of the historical population.
prefixID	Prefix to be added to the ID of the animal of the historical population

Value

Returns haplotype information of the historical population.

See Also

MutationRate(), mapQTLfile(), generateMAP(), QTLeffects()

Examples

```
HP_map <- generateMAP(nChr = 2, n_markers = 5, len_Chr = 5)
QTL_HP <- QTLeffects(n_QTL_Chr = 2, nChr = 2, len_Chr = 5, shape_gamma = 0.4,
effect_distribution = "gamma")
HP_mapqtl <- mapQTLfile(map = HP_map, QTL = QTL_HP)
HP_effect <- as.numeric(HP_mapqtl$Effect)
H1 <- generateHP(n_generations = 5, n_animals = 20, map = HP_mapqtl,
nSire = 1, nDam = 1, nProgeny = 6, mutationRate = 2.5 * 10^-5, SelType = "random",
Effect = HP_effect, h2 = 0.3, trait_mean = 10, VarE = 0.6,recL = 2,nChr = 2, minLength = 0)
```

---

generateMAP *Generate MAP file*

Description

Function to generate MAP file with marker ID, chromosome and marker position columns.

Usage

```
generateMAP(nChr, n_markers, len_Chr)
```

Arguments

nChr	Number of haploid chromosomes (n) defined in the population.
n_markers	Number of SNP markers to be generated per chromosome. This parameter is independent of QTL effects at this stage.
len_Chr	Length of the chromosome. It can be a single value or vector specifying different length per chromosome in cM. If users wants to use base pairs(bp) as a measure to determine position, it can be obtained by bp = cM *10^6 .The base pair position should be calculated by users as it is not part of this function.

Value

Returns a map file of the population with marker ID, chromosome number and position of the markers on the chromosome.

See Also

QTLeffects(), mapQTLfile()

Examples

```
map <- generateMAP(nChr = 2, n_markers = 10, len_Chr = 5)
```

GenOnePopulation *Simulate Population*

Description

This function simulates a population for multiple number of generations as specified by the users. It generates coded Haplotype as per breed of origin, from which we can extract haplotype and genotype.It also generates pedigree file, true breeding value, phenotype, recombination points and recombination probability.

Usage

```
GenOnePopulation(
  ngenerations = 5,
  map = map,
  populationSim,
  nSire = 54,
  nDam = 1000,
  mutationRate = 2.5 * 10^-5,
  SelType = "random",
  h2 = 0.3,
  trait_mean,
  VarE,
  Effect,
```

```
  recL,
  nChr,
  recProb,
  minLength,
  IndStartVal = 1,
  prefixID = "A",
  nProgeny = 1000
)
```

Arguments

ngenerations	Number of generations to be simulated
map	Map file for the population comprising of marker ID, chromosome number and marker position.
populationSim	Coded haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
nSire	Number of sires to be selected as parents for the simulation.
nDam	Number of Dams to be selected as parents for the simulation.
mutationRate	Mutation rate to added to the simulated population.
SelType	Selection Type. It can be based on phenotype, random or true breeding value. The default selection type is "random". For phenotypic selection, use "Pheno" and for true breeding value, use "TBV".
h2	Heritability of the trait to be simulated.
trait_mean	Mean of the trait to be simulated.
VarE	Error Variance obtained after rescaling the QTL effects.
Effect	SNP effect for the calculation of TBV.
recL	Average number of recombinations to be introduced per animal.
nChr	Number of haploid chromosomes (n) defined in the population.
recProb	A vector of recombination probability between the markers. Its length should be total number of markers minus 1. User can define the recombination probability between markers helping in simulating a desired population.
minLength	Minimum length between recombination points beyond which recombination is allowed. The default value is 0. If a user define minimum length, it will ensure that difference between two recombination points less than minimum length won't allow recombination.
IndStartVal	Starting value for the Animal ID of the simulated population.
prefixID	Prefix to be added to the ID of the animal of simulated population.
nProgeny	Number of progeny to be simulated in the population.

Value

List of Haplotype information , pedigree of the simulated animals, population with TBV and phenotype, recombination point list of sire and dam.

See Also

generateMAP(), PopSelect()

Examples

```
PopGen <-  matrix(data = sample(c(-1, 1), 200, replace = TRUE), nrow = 20, ncol = 10)
PopGen <- PopulationID(Population = PopGen, PopCode = "A1")
PopGen_map <- generateMAP(nChr = 2, n_markers = 5, len_Chr = 5)
QTL_PopGen <- c(-0.001, 0.0003, 0, 0, 0, 0.0032, 0.0023, -0.0012,0, 0 )
PopF1 <- GenOnePopulation(ngenerations = 2, map = PopGen_map, populationSim = PopGen,
nSire = 1, nDam = 1, mutationRate = 2.5 * 10^-5, SelType = "TBV", h2 = 0.3,
trait_mean = 5,VarE = 0.6, Effect = QTL_PopGen, recL = 1, nChr = 2,
minLength = 0, IndStartVal = 1, prefixID = "A", nProgeny = 10)
```

MakeGeno *Make Genotype from Haplotype*

Description

Function to generate genotype information from haplotype (0,1) information.

Usage

```
MakeGeno(df)
```

Arguments

df	Haplotype (0,1) information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.

Value

Returns the genotype information of the animal.

Examples

```
popD <- matrix(data = sample(c(0,1), 60, replace = TRUE), nrow = 6, ncol = 10)
genotype <- MakeGeno(df = popD)
```

mapQTLfile *Create combined map and qtl file*

Description

This function combines marker and qtl file together. SNP markers are often observed to be near or away from the QTL in real life scenario. However, QTL position is included along with map file to get combined marker file for simulation purpose in this package. This step is crucial to calculate the true breeding value and simulate the phenotype with desired population parameters.

Usage

```
mapQTLfile(map, QTL)
```

Arguments

map	map file with marker ID, chromosome, and marker position
QTL	qtl file with QTL ID, chromosome, marker position and effect

Value

combined file with marker and QTL ID, chromosome, position and QTL effects.

See Also

generateMAP(), QTLeffects()

Examples

```
popmap <- generateMAP(nChr = 5, n_markers = 10,
len_Chr = c(10, 20, 40, 50, 60))
popQTL <- QTLeffects(n_QTL_Chr = 2, nChr = 5, len_Chr = c(10, 20, 40, 50, 60),
shape_gamma = 0.4, effect_distribution = "gamma")
pop_mapqtl <- mapQTLfile(map = popmap, QTL = popQTL)
```

MatingPop *Function for mating sire and dam*

Description

Function to generate progeny from selected sires and dams.

Usage

```
MatingPop(Sire, Dam)
```

Arguments

Sire	Haplotype information of the Sire in matrix format.
Dam	Haplotype information of the Dam in matrix format.

Value

Returns haplotype information of the progeny.

Examples

```
Sire_K <- matrix(data = sample(c(0,1), 30, replace = TRUE), nrow = 3, ncol = 10)
Dam_K <- matrix(data = sample(c(0,1), 30, replace = TRUE), nrow = 3, ncol = 10)
popK <- MatingPop(Sire = Sire_K, Dam = Dam_K)
```

MutationRate *Adding Mutation*

Description

Function to add mutation rate to the haplotype information of the animals.

Usage

```
MutationRate(Haplotype, rate)
```

Arguments

Haplotype	Haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
rate	Mutation rate. The default mutation rate is = 2.5*10^-5. However, user can also define mutation rate as per requirement.

Value

Returns a haplotype matrix with mutations.

Examples

```
popH <- matrix(data = sample(c(0,1), 60, replace = TRUE), nrow = 6, ncol = 10)
popH_Mutation <- MutationRate(Haplotype = popH, rate = 2.5*10^-5)
```

Ped_Pheno *Combine Pedigree and Phenotype data*

Description

Combine Pedigree and Phenotype data

Usage

```
Ped_Pheno(Pedigree, Phenotype)
```

Arguments

Pedigree	Pedigree file comprising AnimalID, Sire ID, Dam ID and Sex of the animal.
Phenotype	Phenotype file comprising Animal ID, TBV and phenotype of the animal

Value

A datafile comprising AnimalID, Sire ID, Dam ID, Sex, TBV and phenotype of the animal.

See Also

Ext_Pheno()

Examples

```
PopGen1 <-  matrix(data = sample(c(-1, 1), 200, replace = TRUE), nrow = 20, ncol = 10)
PopGen1 <- PopulationID(Population = PopGen1, PopCode = "PG")
PopGen1_map <-  generateMAP(2, 5, 5)
QTL_PopGen1  <-  c(-0.001, 0.0003, 0, 0, 0, 0.0032, 0.0023, -0.0012,0, 0)
PopPP <- GenOnePopulation(ngenerations=2, map=PopGen1_map,
PopGen1,nSire=1,nDam=1,mutationRate=2.5 * 10^-5,
SelType = "TBV", h2 = 0.3, trait_mean = 5 ,VarE = 0.6, Effect = QTL_PopGen1,
recL = 1, nChr = 2, minLength = 0, IndStartVal=1,prefixID="A",nProgeny=10)
Phenofile <- Ext_Pheno(df = PopPP$population)
Ped_Phenofile <- Ped_Pheno(Pedigree = PopPP$parents, Phenotype = Phenofile)
```

plinkped *Generate PLINK pedigree file*

Description

Function to generate ped file for PLINK analysis.

Usage

```
plinkped(Genotype, AnimalID, filename)
```

Arguments

Genotype	A matrix containing the genotype information of the animals where rows
AnimalID	Animal ID for the animals in the population
filename	Naming a ped file

Value

A ped file for PLINK analysis. As of now, it only saves animal ID and genotypes. Saving pedigree in ped file is in our plan for the update.

Examples

```
pop_plink <- matrix(data = sample(c(0,1,2), 60, replace = TRUE), nrow = 6, ncol = 10)
rownames(pop_plink) <- paste("PP", 1:nrow(pop_plink), sep = "_")
output_path <- file.path(tempdir(), "pop")
plinkped(Genotype = pop_plink, AnimalID = rownames(pop_plink), filename = output_path)
```

plotHaplo *Plot for Haplotype segments*

Description

Function to generate plot of haplotype segments inherited from different breeds.It utilises the coded haplotype information according to the breed of origin and generates plot at all chromosomes.

Usage

```
plotHaplo(Haplotype, colors, map, nChr, legendlabels)
```

Arguments

Haplotype	Haplotype information of the animals to be plotted.
colors	Colors to be defined for different populations.
map	Mapfile of the population comprising of marker ID, chromosome number and marker position.
nChr	Haploid number of chromosomes(n).
legendlabels	Population label to add as a legend to the plot.

Value

Haplotype plot showing blocks from different population for animals.

See Also

generateMAP()

Examples

```
Haplotype <- matrix(data = sample(c(-1, 1, -2, 2, -3, 3), 100, replace = TRUE),
nrow = 10, ncol = 10)
Haplotype_ID <- PopulationID(Population = Haplotype, PopCode = "H1")
map <- generateMAP(nChr = 2, n_markers = 5, len_Chr = 5)
colors = c("gold", "forestgreen", "steelblue")
plot <- plotHaplo(Haplotype = Haplotype_ID, colors = colors, map = map,
nChr = 2, legendlabels = c("A", "B", "C"))
```

PopSelect *Selecting Population with selection criteria random, phenotype and true breeding value*

Description

This function select sires and dams of the population using random, phenotypic or true breeding value.

Usage

```
PopSelect(
  map,
  Haplotype,
  nSire,
  nDam,
  rate = 2.5 * 10^-5,
  nProgeny,
  SelType = "random",
  Effect,
```

```
  h2,
  trait_mean,
  VarE,
  recL,
  nChr,
  recProb,
  minLength
)
```

Arguments

map	Map file for the population comprising of marker ID, chromosome number and marker position.
Haplotype	Coded haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
nSire	Number of Sires to be used for simulation.
nDam	Number of Dams to be used for simulation.
rate	Mutation rate to introduce mutation in the population.
nProgeny	Number of Progeny to be simulated in each generation.
SelType	Selection Type. It can be based on phenotype, random or true breeding value. The default selection type is "random". For phenotypic selection, use "Pheno" and for true breeding value, use "TBV".
Effect	SNP effect for the calculation of TBV.
h2	Heritability of the trait to be simulated.
trait_mean	Mean of the trait to be simulated.
VarE	Error Variance obtained after rescaling the QTL effects.
recL	Average number of recombinations to be introduced per animal.
nChr	Number of haploid chromosomes (n) defined in the population.
recProb	A vector of recombination probability between the markers. Its length should be total number of markers minus 1. User can define the recombination probability between markers helping in simulating a desired population.
minLength	Minimum length between recombination points beyond which recombination is allowed. The default value is 0. If a user define minimum length, it will ensure that difference between two recombination points less than minimum length won't allow recombination.

Value

It generates selected sire and dams along with with their ID, recombination points of sire and dam, and population coded haplotype with phenotype and TBV.

See Also

generateMAP()

Examples

```
PopS <- matrix(data = sample(c(0,1), 200, replace = TRUE),
nrow = 20, ncol = 10)
PopS_map <- generateMAP(nChr = 2, n_markers = 5, len_Chr = 5)
PopS_QTL <-  c(-0.001, 0.0003, 0, 0, 0, 0.0032, 0.0023, -0.0012,0, 0 )
Selpop <-  PopSelect(map = PopS_map, Haplotype = PopS, nSire = 2, nDam = 2,
rate = 2.5 * 10^-5, nProgeny = 6, SelType = "random", Effect = PopS_QTL,
trait_mean = 5, VarE = 0.6, h2 = 0.2, recL = 1, nChr = 2, minLength = 0)
```

PopulationID *Adding Population Code before simulating the population*

Description

Function to add Population code to the population which will help in generating parent IDs.

Usage

```
PopulationID(Population, PopCode)
```

Arguments

Population	Haplotype information of the base population to be used for simulating the population.
PopCode	Code to be assigned for the base population which will appear in IDs.

Value

Returns a population with provided PopCode, which will appear as PopulationID.

Examples

```
PopN_Check <- matrix(data = sample(c(0,1), 200, replace = TRUE), nrow = 10, ncol = 20)
PopN_CheckID <- PopulationID(Population = PopN_Check, PopCode = "P1")
```

QMSim_PlinkMAP *Format PLINK MAP file from QMSim*

Description

Function to format map file from QMSim for PLINK analysis.

Usage

```
QMSim_PlinkMAP(filepath, outputpath, filename)
```

Arguments

filepath	Map file generated from the QMSim.
outputpath	Path directory for the output file
filename	Name of the map file to be extracted

Value

Returns a map file that can be used for PLINK analysis.

Examples

```
## Not run:
# This example is not executed since it needs large files.
output_folder <- file.path(tempdir(), "map")
dir.create(output_folder, showWarnings = FALSE, recursive = TRUE)
QMSim_PlinkMAP(filepath = "lm_mrk_qtl_001.txt",
outputpath = output_folder, filename = "pop")

## End(Not run)
```

QTLeffects *Calculation QTL effects*

Description

Function to estimate QTL effects with output of QTL ID, chromosome, QTL position and QTL effects

Usage

```
QTLeffects(
  n_QTL_Chr,
  nChr,
  len_Chr,
  shape_gamma = 0.4,
  effect_distribution = "gamma"
)
```

Arguments

n_QTL_Chr	Number of QTL to be simulated per chromosome
nChr	Number of haploid chromosomes (n) defined in the population.
len_Chr	Length of the chromosome.It can be either provided single constant value or vector specifying varied length of the chromosomes.
shape_gamma	Shape of the QTL effects if simulated using gamma distribution.
effect_distribution	Define the distribution of QTL effects. It can be either gamma, normal or uniform distribution.

Value

file comprising of QTL ID, chromosomes, position of the QTL and QTL effects.

See Also

generateMAP(), mapQTLfile()

Examples

```
QTL <- QTLeffects(n_QTL_Chr = 1, nChr = 2, len_Chr = 10,
shape_gamma = 0.4, effect_distribution = "gamma")
```

QTL_densityplot *Plot for the QTL effect*

Description

Generates the density plot for the QTL effects.

Usage

```
QTL_densityplot(QTL_eff_file, color = "steelblue")
```

Arguments

QTL_eff_file	QTL file with information on QTL effects.
color	color of the density plot.

Value

shows density plot depicting the distribution of the QTL effects.

See Also

QTLeffects()

Examples

```
example_QTL <- QTLeffects(n_QTL_Chr = 1, nChr = 5, len_Chr = c(10, 8, 6, 4, 2),
shape_gamma = 0.4)
plot <- QTL_densityplot(QTL_eff_file = example_QTL, color = "forestgreen")
```

Random_gametes *Random selection of sire and dam gametes*

Description

Random selection of sire and dam gametes

Usage

```
Random_gametes(Gamete_Matrix)
```

Arguments

Gamete_Matrix	A matrix comprising haplotype information of sire or dam gametes after recombination

Value

Returns matrix comprising random selection of gametes

Examples

```
mat = matrix(data = sample(c(-1,1,-2,2), 100*10, replace = TRUE)
, nrow = 100, ncol = 10)
mat = PopulationID(Population = mat, PopCode = "M")
rev_mat = Random_gametes(Gamete_Matrix = mat)
```

ReadQTL *Read QTL file obtained from QMSim*

Description

This function reads the effect file generated from QMSim with the file prefix "effect_qtl". It generates the effects for the defined QTLs in QMSim.

Usage

```
ReadQTL(filename)
```

Arguments

filename	Requires the name of the QTL file with allele effect obtained from QMSim.

Value

Returns a readable QTL effect

See Also

CalQTLeffect()

Examples

```
## Not run:
# This example is not executed since it needs large files.
QTLfile <- ReadQTL(filename = "effect_qtl_001.txt")

## End(Not run)
```

read_qmsim_geno *Read QMSim Marker/QTL Genotype File*

Description

Efficiently reads a QMSim p1_mrk_qtl_*.txt genotype file using compiled C++ code (Rcpp) and returns a single integer matrix.

Usage

```
read_qmsim_geno(file_path, geno_mode = "haplotype", phase_seed = 42L)
```

Arguments

`file_path`	Character string. Path to the QMSim genotype file.
`geno_mode`	Character string controlling output layout. One of:
	`"haplotype"` (default) Two rows per individual (paternal haplotype then maternal haplotype), values 0/1. Matrix dimensions: `2*n_ind` rows x `n_loci` columns. Genotype codes are decoded as: `0` = 0|0, `2` = 1|1, `3` = 0|1, `4` = 1|0. Code `1` (unphased het) is resolved deterministically using `phase_seed`. Both rows carry the animal ID as rowname.
	`"allele"` One row per individual, raw genotype codes (0-4) as they appear in the file. Matrix dimensions: `n_ind` rows x `n_loci` columns.
`phase_seed`	Integer seed for deterministic phasing of unphased heterozygotes (code 1). Only used when `geno_mode = "haplotype"`. Default: 42.

Details

Each character in the QMSim genotype string represents one locus:

Code	Genotype	Paternal	Maternal
0	0|0	0	0
1	het (unphased)	?	?
2	1|1	1	1
3	0|1	0	1
4	1|0	1	0

Lines ending with > are continuation lines and are concatenated automatically.

Value

An integer matrix with rownames set to animal IDs.

Examples

```
## Not run:
# This example is not executed since it needs large files.
# Two 0/1 rows per individual (paternal / maternal haplotypes)
hap <- read_qmsim_geno("p1_mrk_qtl_001.txt")
dim(hap)        # 2*n_ind  x  n_loci

# Raw genotype codes (0-4) as in the file
geno <- read_qmsim_geno("p1_mrk_qtl_001.txt", geno_mode = "allele")
dim(geno)       # n_ind  x  n_loci

## End(Not run)
```

Recode_Haplotype *Recode Haplotype with different breed codes*

Description

Function to recode the haplotype according to different breed codes as specified by users.

Usage

```
Recode_Haplotype(df, popBase)
```

Arguments

df	A matrix consisting of haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
popBase	Code specified according to the specific breed.

Value

Returns a matrix consisting of coded haplotype information of the animals according to the defined popBase.

Examples

```
popA <- matrix(data = sample(c(0,1), 60, replace = TRUE), nrow = 6, ncol = 10)
PA_Haplo <- Recode_Haplotype(df = popA, popBase = 1)
```

RecombinationPoint *Add Recombination*

Description

Function to add recombination to the haplotype information of the animals.

Usage

```
RecombinationPoint(map, Haplotype, recL, nChr, recProb)
```

Arguments

map	Mapfile of the population comprising of marker ID, chromosome number and marker position.
Haplotype	Haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
recL	Average number of recombinations to be introduced per animal.
nChr	Number of haploid chromosomes (n) defined in the population.
recProb	A vector of recombination probability between the markers. Its length should be total number of markers minus 1. User can define the recombination probability between markers helping in simulating a desired population.

Value

Returns a list of recombination points and number of recombinations per animal.

Examples

```
popG <- matrix(data = sample(c(0,1), 40, replace = TRUE), nrow = 4, ncol = 10)
popGmap <- generateMAP(nChr = 2, n_markers = 5, len_Chr = 10)
popGRec <- RecombinationPoint(map = popGmap, Haplotype = popG, recL = 2, nChr = 2)
```

---

recProbplot *Plot for recombination points and recombination probability*

Description

Function to generate plot for recombination points and recombination probability.

Usage

```
recProbplot(recList, n_markers, col = "steelblue", map)
```

Arguments

recList	List of recombination points with each list containing information of recombination points of an animal.
n_markers	Total number of markers.
col	Color of the recombination probability plot.
map	Map file of the markers for the population.

Value

This function returns the plots for recombination points and recombination probability.

See Also

generateMAP(), RecombinationPoint()

Examples

```
map_rec <- generateMAP(nChr = 2, n_markers = 5, len_Chr = 10)
poprec <- matrix(data = sample(c(0,1), 1000, replace = TRUE), nrow = 100, ncol = 10)
RecPoint <- RecombinationPoint(map = map_rec, Haplotype = poprec, recL = 2, nChr = 2)
reclist <- RecPoint$recList
recom_plot <- recProbplot(recList = reclist, n_markers = 10, map = map_rec)
```

Reformat_BLUPF90 *Reformat genotype data for BLUPF90 analysis*

Description

Function to format genotype file for BLUPF90 analysis.

Usage

```
Reformat_BLUPF90(Genodata, savefile)
```

Arguments

Genodata	A matrix comprising genotype information (0,1,2) of the animals where rownames depicts the animal ID.
savefile	File name in which the genotype data will be saved.

Value

Returns a text file comprising of genotype data for its use in BLUPF90.

Examples

```
Geno <- matrix(data = sample(c(0,1,2), 100, replace = TRUE), nrow = 10, ncol = 10)
rownames(Geno) <- paste("G", 1:nrow(Geno), sep = "_")
output_path <- file.path(tempdir(), "genotype")
Reformat_BLUPF90(Genodata = Geno, savefile = output_path)
```

Rescale *Rescaling SNP Effect*

Description

This function allows scaling of the SNP effects based on the scaling factor derived from genotype of the base population and standard deviation of the phenotype. This allows users to scale the SNP effects to get desirable variance components.

Usage

```
Rescale(Haplotype, Effect, Phenosd, h2)
```

Arguments

Haplotype	A matrix where each row represents a haplotype (0,1) and each column represents a marker.
Effect	A vector with SNP effect of a population to be scaled before the simulation of a population
Phenosd	A value of phenotypic standard deviation which will help in estimating desirable variance components
h2	Heritability of the trait to be simulated for the population.

Value

This function returns a scaled SNP effect, scaled BV along with phenotypic, additive and residual variance.

Examples

```
popRescale <- matrix(c(0,1,1,0,
1,0,0,0,
1,1,1,1,
1,1,1,0), byrow = TRUE, nrow = 4, ncol = 4)
QTLEff = c(0.1, 0.2, 0, 0)
RescaleEff <- Rescale(Haplotype = popRescale, Effect = QTLEff,
Phenosd = 4, h2 = 0.2)
```

SireSample *Select Sires*

Description

Function to select the sire.

Usage

```
SireSample(Haplotype, nSire, nProgeny)
```

Arguments

Haplotype	Haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
nSire	Number of sires to be selected as parents for the simulation.
nProgeny	Number of progeny to be simulated in the population.

Value

Returns a list of sire IDs and haplotype information of the sire.

Examples

```
popH <- matrix(data = sample(c(0,1), 60, replace = TRUE), nrow = 6, ncol = 10)
popH_Sire <- SireSample(Haplotype = popH, nSire = 1, nProgeny = 4)
```

TBV_Haplo *Calculation of True breeding Value (TBV)*

Description

Function to estimate true breeding value.

Usage

```
TBV_Haplo(Haplotype, Effect)
```

Arguments

Haplotype	Haplotype (0,1) information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
Effect	A vector comprising the SNP effects of the population.

Value

Returns a haplotype information with TBV estimated from animal's haplotype information. The TBV mentioned in "TBV" column provides final TBV (sum of TBV from haplotype of the animal) for an animal.

See Also

Var_PhenoRecords()

Examples

```
popE <- matrix(data= sample(c(0,1), 16, replace= TRUE), nrow = 4, ncol = 4)
QTLEff <- c(0.2, 0.1, 0.3, 0.1)
TBV_haplotype <- TBV_Haplo(Haplotype = popE, Effect = QTLEff)
```

---

trueBreedComp *Calculate True Breed Proportion with coded haplotype*

Description

Function to calculate true breed proportion in an animal with coded haplotype information.

Usage

```
trueBreedComp(Haplotype)
```

Arguments

Haplotype	A matrix with coded haplotype information of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.

Value

Returns true breed proportion of population with coded haplotype.

See Also

Admixplot()

Examples

```
P1 <- matrix(data = sample(c(-1, 1, 2, -2), 20, replace = TRUE), nrow = 4, ncol = 5)
TBP <- trueBreedComp(Haplotype = P1)
```

TwoPopCross *Function for crossing two populations*

Description

Function for making a cross between two populations.

Usage

```
TwoPopCross(
  map = map,
  populationSim_A,
  populationSim_B,
  nSireA = 54,
  nDamA = 1000,
  nSireB = 26,
  nDamB = 1000,
  Sire_From,
  mutationRate = 2.5 * 10^-5,
  SelType,
  h2 = 0.3,
  trait_mean,
  VarE_PopA,
  VarE_PopB,
  Effect_PopA,
  Effect_PopB,
  IndStartVal = 1,
  prefixID = "Cr",
  nProgeny = 1000,
  recL,
  nChr,
  recProb,
  minLength
)
```

Arguments

map	Map file for the population comprising of marker ID, chromosome number and marker position.
populationSim_A	Coded haplotype information of the population A where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
populationSim_B	Coded haplotype information of the population B where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.

nSireA	Number of sires to be selected as parents for the simulation from population A.
nDamA	Number of dams to be selected as parents for the simulation from population A.
nSireB	Number of sires to be selected as parents for the simulation from population B.
nDamB	Number of dams to be selected as parents for the simulation from population B.
Sire_From	The population from which sires should being selected for cross population. If sire should be selected from population A, then use "A" and if sire should be selected from population B, then use "B".
mutationRate	Mutation rate to added to the simulated population.
SelType	Selection Type. It can be based on phenotype, random or true breeding value. The default selection type is "random". For phenotypic selection, use "Pheno" and for true breeding value, use "TBV".
h2	Heritability of the trait to be simulated.
trait_mean	Mean of the trait to be simulated.
VarE_PopA	Error Variance obtained after rescaling the QTL effects for PopA.
VarE_PopB	Error Variance obtained after rescaling the QTL effects for PopB.
Effect_PopA	SNP effect for the calculation of TBV for population A.
Effect_PopB	SNP effect for the calculation of TBV for population B.
IndStartVal	Starting value for the animal ID.
prefixID	Prefix to be added to the ID of the animal.
nProgeny	Number of progeny to be simulated in the population.
recL	Average number of recombinations to be introduced per animal.
nChr	Number of haploid chromosomes (n) defined in the population.
recProb	A vector of recombination probability between the markers. Its length should be total number of markers minus 1. User can define the recombination probability between markers helping in simulating a desired population.
minLength	Minimum length between recombination points beyond which recombination is allowed. The default value is 0. If a user define minimum length, it will ensure that difference between two recombination points less than minimum length won't allow recombination.

Value

List of Haplotype information , ID of the simulated animals, population with TBV and phenotype, recombination point list of sire and dam.

See Also

generateMAP(), PopSelect(), RecombinationPoint()

Examples

```
crossmap <- generateMAP(nChr = 5, n_markers = 2, len_Chr = 20)
QTL_cross <- c(-0.001, 0.0003, 0, 0, 0, 0.0032, 0.0023, -0.0012,0, 0 )
P1cross <- matrix(data = sample(c(-1, 1), 40, replace = TRUE), nrow = 4, ncol = 10)
P2cross <- matrix(data = sample(c(-2, 2), 40, replace = TRUE), nrow = 4, ncol = 10)
P1cross <- PopulationID(Population = P1cross, PopCode = "P1")
P2cross <- PopulationID(Population = P2cross, PopCode = "P2")
F2cross <- TwoPopCross(map = crossmap, populationSim_A = P1cross, populationSim_B = P2cross,
nSireA = 1, nDamA = 1, nSireB = 1, nDamB = 1, Sire_From = "A", mutationRate = 2.5 * 10^-5,
SelType = "TBV", h2 = 0.3, Effect_PopA = QTL_cross, Effect_PopB = QTL_cross, IndStartVal = 1,
prefixID ="Cr", nProgeny = 10, recL = 1, nChr = 2, minLength = 0, trait_mean = 5,
VarE_PopA = 0.6, VarE_PopB = 0.6)
```

Var_PhenoRecords *Generate variance components and phenotypic records*

Description

Function to generate variance components and phenotype records for the simulated population.

Usage

```
Var_PhenoRecords(Haplotype, h2, trait_mean, VarE)
```

Arguments

Haplotype	Haplotype information (0,1) of the population where rows are number of animals and columns are number of markers in matrix form. In case of haplotype, 2 rows provide information on genomic material of an animal.
h2	Heritability of the trait to be simulated.
trait_mean	Mean of the trait to be simulated.
VarE	Error Variance obtained after rescaling the QTL effects

Value

Additive genetic variance, phenotypic variance, error variance and haplotype marker information with phenotypic records.

See Also

TBV_Haplo()

Examples

```
popF <- matrix(data= sample(c(0,1), 16, replace= TRUE), nrow = 4, ncol = 4)
QTLEff <- c(0.2, 0.1, 0.3, 0.1)
TBV_haplotype <- TBV_Haplo(Haplotype = popF, Effect = QTLEff)
Var_Ph_R <- Var_PhenoRecords(Haplotype = TBV_haplotype, h2 = 0.3,
trait_mean = 20, VarE = 0.6)
```

Index

Admixplot, 2
Admixplot(), *36*
AlleleFreq, 3

BreedEff_Haplo, 4
BreedEff_Haplo(), *6*, *8*, *13*
BreedPopCross, 4
BreedPopSelect, 7
BreedPopSelect(), *6*, *13*

CalQTLeffect, 9
CalQTLeffect(), *29*
Coded_to_Haplo, 10

DamSample, 10

Ext_Pheno, 11
Ext_Pheno(), *21*

GenBreedPop, 12
generateHP, 14
generateMAP, 15
generateMAP(), *15*, *18, 19*, *23, 24*, *27*, *33*, *38*
GenOnePopulation, 16

MakeGeno, 18
mapQTLfile, 19
mapQTLfile(), *15, 16*, *27*
MatingPop, 19
MutationRate, 20
MutationRate(), *15*

Ped_Pheno, 21
plinkped, 22
plotHaplo, 22
PopSelect, 23
PopSelect(), *18*, *38*
PopulationID, 25

QMSim_PlinkMAP, 26
QTL_densityplot, 27
QTLeffects, 26
QTLeffects(), *15, 16*, *19*, *28*

Random_gametes, 28
read_qmsim_geno, 29
ReadQTL, 29
ReadQTL(), *9*
Recode_Haplotype, 31
RecombinationPoint, 31
RecombinationPoint(), *33*, *38*
recProbplot, 32
Reformat_BLUPF90, 33
Rescale, 34

SireSample, 35

TBV_Haplo, 35
TBV_Haplo(), *39*
trueBreedComp, 36
trueBreedComp(), *3*
TwoPopCross, 37

Var_PhenoRecords, 39
Var_PhenoRecords(), *36*