Package 'pedSimulate'

September 24, 2023

Title Pedigree, Genetic Merit, Phenotype, and Genotype Simulation

Version 1.4.1

Description Simulate pedigree, genetic merits and phenotypes with random/non-random matings followed by random/non-random selection with different intensities and patterns in males and females. Genotypes can be simulated for a given pedigree, or an appended pedigree to an existing pedigree with genotypes.
Mrode, R. A. (2005) <ISBN:9780851989969, 0851989969>;
Nilforooshan, M.A. (2022) <doi:10.37496/rbz5120210131>.

License GPL-3

LazyData true

URL https://github.com/nilforooshan/pedSimulate

BugReports https://github.com/nilforooshan/pedSimulate/issues

RoxygenNote 7.2.3

Encoding UTF-8

Repository CRAN

R topics documented:

appendGen *Simulate genotypes for an appended pedigree*

Description

Simulate genotypes for an appended pedigree to an existing pedigree with genotypes.

Usage

```
appendGen(ped, M, AF = c(), mut.rate = 0, seed = NA)
```

Arguments

ped	: Pedigree data.frame with columns for animal, sire, and dam identification.
M	: Genotype data.frame with rows corresponding to the initial rows of the pedigree and columns corresponding to markers.
AF	: Vector of allele frequencies at different loci for the genotypes to be simulated. If no value is provided, it will be estimated from M.
mut.rate	: Vector of mutation rates at different loci for the genotypes to be simulated, default = 0 for no mutation.
seed	: A numeric variable input to the random number generator for reproducible simulations, default = 'NA' for non-reproducible simulations.

Details

Only diploid and bi-allelic situations are covered. No linkage disequilibrium is simulated.

Value

M2 : New simulated genotypes appended to M.

Examples

```
nSNP = 100
AF = runif(nSNP, 0.01, 0.99)
mut.rate = runif(nSNP, 0, 10^-5)
ped = data.frame(ID=1:5, SIRE=c(0,0,1,0,3), DAM=c(0,0,2,2,4))
gen = simulateGen(ped, AF, mut.rate)
ped = rbind(ped, data.frame(ID=6:8, SIRE=c(3,6,6), DAM=c(0,4,5)))
gen = appendGen(ped, M=gen, AF, seed=34)
```

appendPed *Simulate new generations from an existing pedigree*

Description

Simulate pedigree, genetic merits and phenotypes with random/assortative/disassortative matings followed by random/non-random selection of males and females with similar/different selection patterns in males and females, starting from an existing pedigree.

Usage

```
appendPed(
  ped,
  Va0,
  Ve,
  littersize = 1,
  ngen,
  mort.rate = 0,
  overlap.s = 0,
  overlap.d = 0,
  f.rate = 1,
  m.rate = 1,
  fsel = "R",
  msel = "R",
  f.order = "fsel",
  m.order = "msel",
  seed = NA
)
```

Arguments

`ped`	: The input pedigree `data.frame` with 9 columns: ID, SIRE, DAM, SEX, GEN (generation), PA (parent average), MS (Mendelian Sampling), E (environment and residuals), and P (phenotype). Note that PA + MS + E = P - μ, where μ is the population mean, and PA + MS = BV (genetic merit or breeding value). If MS and E are unknown, those can be set to 0. PA should be equal to the average of sire BV (SBV) and dam BV (DBV). If this condition is not met, PA - (SBV + DBV)/2 is added to MS and (SBV + DBV)/2 replaces PA.
`Va0`	: Additive genetic variance in the base generation (i.e., F0).
`Ve`	: Residual variance, constant across generations.
`littersize`	: Litter size, default = 1.
`ngen`	: Number of generations to simulate after the founder generation.
`mort.rate`	: Mortality rate per generation, after the availability of phenotype (e.g., birth weight, weaning weight) and before the age of maturity (i.e., before mating), default = 0. Maximum `mort.rate` = 0.5.
`overlap.s`	: Number of overlapping generations for sires, default = 0 for no generation overlap.
`overlap.d`	: Number of overlapping generations for dams, default = 0 for no generation overlap.

f.rate	: Proportion of females selected as dams, default = 1.
m.rate	: Proportion of males (<= f.rate) selected as sires, default = 1.
fsel	: If "R" (default), random selection on females; if "P", selection on phenotypes or true breeding values if Ve = 0; if "PA", selection on true parent averages. "-P" and "-PA" work in opposite direction of "P" and "PA", respectively.
msel	: If "R" (default), random selection on males; if "P", selection on phenotypes or true breeding values if Ve = 0; if "PA", selection on true parent averages. "-P" and "-PA" work in opposite direction of "P" and "PA", respectively.
f.order	: Ordering selected females for mating; if "fsel" (default), same as the selection order; if "R" random ordering; if "P", ordering based on phenotypes or true breeding values if Ve = 0; if "PA", ordering based on true parent averages. "-P" and "-PA" work in opposite direction of "P" and "PA", respectively.
m.order	: Ordering selected males for mating; if "msel" (default), same as the selection order; if "R" random ordering; if "P", ordering based on phenotypes or true breeding values if Ve = 0; if "PA", ordering based on true parent averages. "-P" and "-PA" work in opposite direction of "P" and "PA", respectively.
seed	: A numeric variable input to the random number generator for reproducible simulations, default = 'NA' for non-reproducible simulations.

Value

ped2 : New generations appended to the input pedigree data.frame.

Examples

```
ped = simulatePed(
    F0size = 100,
    Va0 = 9,
    Ve = 36,
    littersize = 2,
    ngen = 4,
    mort.rate = 0.05,
    overlap.s = 1,
    overlap.d = 0,
    f.rate = 0.8,
    m.rate = 0.5,
    fsel = "P",
    msel = "PA"
)
ped2 = appendPed(
    ped = ped,
    Va0 = 9,
    Ve = 36,
    littersize = 2,
    ngen = 2,
    mort.rate = 0.05,
    overlap.s = 1,
    overlap.d = 0,
    f.rate = 0.8,
    m.rate = 0.5,
    fsel = "R",
    msel = "R",
    f.order = "P",
    m.order = "PA",
```

```
    seed = 76
)
```

`fs_mate_finder` *Find fullsib mates*

Description

Find fullsib matings in the pedigree

Usage

```
fs_mate_finder(ped)
```

Arguments

ped	: A pedigree `data.frame`. The first three columns (ID, SIRE, DAM) are used.

Value

fs_mates : A `data.frame` with two columns (SIRE, DAM) representing fullsib mates.

Examples

```
ped = data.frame(ID=1:7, SIRE=c(0,0,1,0,3,3,5), DAM=c(0,0,0,2,4,4,6))
fs_mate_finder(ped)
```

`hs_mate_finder` *Find halfsib mates*

Description

Find halfsib matings in the pedigree

Usage

```
hs_mate_finder(ped)
```

Arguments

ped	: A pedigree `data.frame`. The first three columns (ID, SIRE, DAM) are used.

Value

hs_mates : A `data.frame` with two columns (SIRE, DAM) representing halfsib mates.

Examples

```
ped = data.frame(ID=1:7, SIRE=c(0,0,1,1,0,3,5), DAM=c(0,0,2,2,2,4,4))
hs_mate_finder(ped)
```

pp_mate_finder *Find parent-progeny mates*

Description

Find parent-progeny matings in the pedigree

Usage

```
pp_mate_finder(ped)
```

Arguments

ped	: A pedigree data.frame. The first three columns (ID, SIRE, DAM) are used.

Value

pp_mates : A data.frame with two columns (SIRE, DAM) representing parent-progeny mates.

Examples

```
ped = data.frame(ID=1:4, SIRE=c(0,0,1,1), DAM=c(0,0,2,3))
pp_mate_finder(ped)
```

simulateGen *Simulate genotypes*

Description

Simulate genotypes for a given pedigree, allele frequency and mutation rate at each marker locus.

Usage

```
simulateGen(ped, AF, mut.rate = 0, seed = NA)
```

Arguments

ped	: Pedigree data.frame with columns for animal, sire, and dam identification.
AF	: Vector of allele frequencies at different loci for the genotypes to be simulated.
mut.rate	: Vector of mutation rates at different loci for the genotypes to be simulated, default = 0 for no mutation.
seed	: A numeric variable input to the random number generator for reproducible simulations, default = 'NA' for non-reproducible simulations.

Details

Only diploid and bi-allelic situations are covered. No linkage disequilibrium is simulated.

Value

M : The simulated genotype `data.frame` with rows corresponding to animals (in the same order as in the pedigree) and columns corresponding to markers.

Examples

```
nSNP = 100
AF = runif(nSNP, 0.01, 0.99)
mut.rate = runif(nSNP, 0, 10^-5)
ped = data.frame(ID=1:5, SIRE=c(0,0,1,0,3), DAM=c(0,0,2,2,4))
gen = simulateGen(ped, AF, mut.rate, seed=684)
```

---

simulatePed *Simulate pedigree, genetic merits and phenotypes*

Description

Simulate pedigree, genetic merits and phenotypes with random/assortative/disassortative matings followed by random/non-random selection of males and females with similar/different selection patterns in males and females.

Usage

```
simulatePed(
  F0size,
  Va0,
  Ve,
  littersize = 1,
  ngen,
  mort.rate = 0,
  overlap.s = 0,
  overlap.d = 0,
  f.rate = 1,
  m.rate = 1,
  fsel = "R",
  msel = "R",
  f.order = "fsel",
  m.order = "msel",
  seed = NA
)
```

Arguments

`F0size`	: Even number of founder animals. No mortality, selection and non-random mating in this generation.
`Va0`	: Additive genetic variance in the base generation (i.e., F0).
`Ve`	: Residual variance, constant across generations.
`littersize`	: Litter size, default = 1.
`ngen`	: Number of generations to simulate after the founder generation.

`mort.rate` : Mortality rate per generation, after the availability of phenotype (e.g., birth weight, weaning weight) and before the age of maturity (i.e., before mating), default = 0. Maximum `mort.rate` = 0.5.

`overlap.s` : Number of overlapping generations for sires, default = 0 for no generation overlap.

`overlap.d` : Number of overlapping generations for dams, default = 0 for no generation overlap.

`f.rate` : Proportion of females selected as dams, default = 1.

`m.rate` : Proportion of males (<= `f.rate`) selected as sires, default = 1.

`fsel` : If `"R"` (default), random selection on females; if `"P"`, selection on phenotypes or true breeding values if `Ve` = 0; if `"PA"`, selection on true parent averages. `"-P"` and `"-PA"` work in opposite direction of `"P"` and `"PA"`, respectively.

`msel` : If `"R"` (default), random selection on males; if `"P"`, selection on phenotypes or true breeding values if `Ve` = 0; if `"PA"`, selection on true parent averages. `"-P"` and `"-PA"` work in opposite direction of `"P"` and `"PA"`, respectively.

`f.order` : Ordering selected females for mating; if `"fsel"` (default), same as the selection order; if `"R"` random ordering; if `"P"`, ordering based on phenotypes or true breeding values if `Ve` = 0; if `"PA"`, ordering based on true parent averages. `"-P"` and `"-PA"` work in opposite direction of `"P"` and `"PA"`, respectively.

`m.order` : Ordering selected males for mating; if `"msel"` (default), same as the selection order; if `"R"` random ordering; if `"P"`, ordering based on phenotypes or true breeding values if `Ve` = 0; if `"PA"`, ordering based on true parent averages. `"-P"` and `"-PA"` work in opposite direction of `"P"` and `"PA"`, respectively.

`seed` : A numeric variable input to the random number generator for reproducible simulations, default = 'NA' for non-reproducible simulations.

Details

The output pedigree `data.frame` (`ped`) has 9 columns: ID, SIRE, DAM, SEX, GEN (generation number starting with 0 for the base generation), PA (parent average), MS (Mendelian Sampling), E (environment and residuals), and P (phenotype).

Random, assortative, and disassortative matings can be simulated with different combinations of `fsel`, `msel`, `f.order`, and `m.order`.

Value

ped : The output pedigree `data.frame`. Further information provided in **Details**.

Examples

```
ped = simulatePed(
    F0size = 100,
    Va0 = 9,
    Ve = 36,
    littersize = 2,
    ngen = 4,
    mort.rate = 0.05,
    overlap.s = 1,
    overlap.d = 0,
    f.rate = 0.8,
    m.rate = 0.5,
```

```
        fsel = "P",
        msel = "PA",
        f.order = "fsel",
        m.order = "msel",
        seed = 68
    )
```

Index