Package 'selfingTree'

May 9, 2026

Type Package

Title Genotype Probabilities in Intermediate Generations of Inbreeding Through Selfing

Version 0.2

License BSD_3_clause + file LICENSE

Date 2014-12-18

Author Frank Technow [aut, cre] (Pioneer Hi-Bred International, Inc., Johnston, Iowa)

LazyData TRUE

Maintainer Frank Technow <Frank.Technow@pioneer.com>

Depends R (>= 2.15.1),foreach

Description A probability tree allows to compute probabilities of complex events, such as genotype probabilities in intermediate generations of inbreeding through recurrent self-fertilization (selfing). This package implements functionality to compute probability trees for two- and three-marker genotypes in the F2 to F7 selfing generations. The conditional probabilities are derived automatically and in symbolic form. The package also provides functionality to extract and evaluate the relevant probabilities.

NeedsCompilation no

Repository CRAN

Date/Publication 2014-12-18 17:31:18

Contents

`selfingTree-package` *Genotype Probabilities in Intermediate Generations of Inbreeding Through Selfing*

Description

A probability tree allows to compute probabilities of complex events, such as genotype probabilities in intermediate generations of inbreeding through recurrent self-fertilization (selfing). This package implements functionality to compute probability trees for two- and three-marker genotypes in the F2 to F7 selfing generations. The conditional probabilities are derived automatically and in symbolic form. The package also provides functionality to extract and evaluate the relevant probabilities.

Details

Package:	selfingTree
Type:	Package
Version:	0.2
Date:	2014-12-18
LazyData:	yes

Depends: foreach

Function buildSelfingTree generates the probability trees for two- and three-marker genotypes. This is done by recursively calling functions genSubtree.2M (two-marker genotypes) or genSubtree.3M (three-marker genotypes). The core functionality of deriving the symbolic conditional haplotype probabilities is implemented in functions haploProb.2M and haploProb.3M. The function nodeProbabilities is used to symbolically multiply the conditional probabilities along all branches and uses function extractProbs to extract the conditional probabilities from the trees. Finally function evalProb symbolically sums the marginal probabilities of relevant nodes and evaluates them with user specified values for the recombination frequencies. The function getTargets can be used to identify relevant events given a target genotype.

Author(s)

Frank Technow

at Pioneer Hi-Bred International, Inc., Breeding Technologies, Johnson/IA, USA.

Maintainer: Frank Technow < Frank.Technow@pioneer.com >

---

buildSelfingTree *Builds the probability tree*

Description

This function builds the probability tree for recurrent selfing.

Usage

```
buildSelfingTree(genF,generation,gam1,gam2)
```

Arguments

genF	A function that generates a sub-tree of all possible genotypes given a parental genotype, either genSubtree.2M or genSubtree.3M.
generation	Integer giving the selfing generation to which the tree will be build. Values can range from 2 to 7, e.g., the F2 generation build by default and the highest possible generation is currently the F7.
gam1, gam2	Three (three marker genotypes) or two (two marker genotypes) character string with the configuration of gametes one and two of the parental F1 genotype.

Value

A recursive data type in the form of a nested list. Each element is a list with three elements. Element [[1]] holds the genotype configuration as "gam1-gam2" (e.g., "ABA-BAB"), element [[2]] the symbolic formula representing the probability of observing this genotype given the parental genotype and element [[3]] is again a list containing the sub-tree rooted at this genotype.

Author(s)

Frank Technow

Examples

```
## F2 and F3 genotypes
F.2M <- buildSelfingTree(genSubtree.2M,3,"AA","BB")

## F2 and F3 genotypes
F.3M <- buildSelfingTree(genSubtree.3M,3,"AAA","BBB")
```

---

evalProb *Evaluates the genotype probability*

Description

This function symbolically sums the marginal probabilities of relevant nodes and evaluates them with user specified values for the recombination frequencies.

Usage

```
evalProb(node.prob, x = 0, y = 0, z = 0, chunk.size = min(length(node.prob),75))
```

Arguments

node.prob	Character vector with symbolic marginal node probabilities, i.e., a (subset of) an element of the list returned by function nodeProbabilities.
x, y, z	Recombination frequencies. For three-marker genotypes, x is the recombination frequency between markers 1 and 2 and y that between markers 2 and 3. For two-marker genotypes, z is recombination frequency between markers 1 and 2.
chunk.size	node.prob is split into several parts of size equal to chunk.size and summation done within each chunk first and then across chunks.

Value

The genotype probability (numeric).

Author(s)

Frank Technow

Examples

```
evalProb(extractProbs(genSubtree.3M("BAA","AAB")),x = 0.123,y = 0.344)
```

extractProbs *Extracts conditional genotype probabilities*

Description

This function extracts the symbolic formulas for the conditional genotype probabilities from the uppermost level of the (sub)tree.

Usage

```
extractProbs(F)
```

Arguments

F	A sub-tree in the format generated by function genSubtree.2M or genSubtree.3M.

Value

A character vector with the symbolic formulas. For three-marker genotypes, symbol x is the recombination frequency between markers 1 and 2 and y that between markers 2 and 3. For two-marker genotypes, symbol z is the recombination frequency between markers 1 and 2. The names of the elements indicate the allelic configuration of the two gametes comprising the genotype as gamete1-gamete2 (e.g., "AAB-AAA"). The elements sum to 1.

Author(s)

Frank Technow

Examples

```
probs.2M <- extractProbs(genSubtree.2M("BA","AA"))
probs.3M <- extractProbs(genSubtree.3M("BAA","AAB"))

## must sum to 1
stopifnot(all.equal(evalProb(probs.2M, z = 0.044),1))
stopifnot(all.equal(evalProb(probs.3M, x = 0.123, y = 0.344),1))
```

F4 *Genotypes of one million simulated F4 lines.*

Description

These are three-marker genotypes of one million F4 lines from a cross between parent A and B, simulated using R package **hypred**.

Usage

```
F4
```

Format

A character matrix with one million rows and three columns. Homozygousity for parents A or B is coded as "A" and "B", respectively. Heterozygousity as "H".

References

Frank Technow(2013). hypred: Simulation of Genomic Data in Applied Genetics. R package version 0.4.

---

genSubtree *Generates a sub-tree*

Description

These functions generate sub-trees consisting of all genotypes (and their conditional probabilities) that can result after selfing the parental genotype.

Usage

```
genSubtree.2M(gam1,gam2) ## two-marker genotypes
genSubtree.3M(gam1,gam2) ## three-marker genotypes
```

Arguments

gam1, gam2	Three (three-marker genotypes) or two (two-marker genotypes) character string with the configuration of gamete one and two of the parental genotype.

Value

A `list` with one element per possible genotype. Each element is itself a `list` with two elements. Element `[[1]]` holds the genotype configuration as "gam1-gam2" (e.g., `"ABA-BAB"`), element `[[1]]` the symbolic formula representing the probability of observing this genotype given the parental genotype.

Author(s)

Frank Technow

Examples

```
genSubtree.2M("AB","AA")
genSubtree.3M("ABA","AAA")
```

getTargets — *Obtain all possible genotypes that match a certain target configuration*

Description

This convenience function finds all genotypes that match a certain target configuration. It is used only if the target configuration contains heterozygous states, but order (e.g., A/B or B/A) does not matter.

Usage

```
getTargets(target.geno)
```

Arguments

target.geno	Three (three-marker genotypes) or two (two-marker genotypes) character string specifying the target configuration. Homozygousity for parent A allele is indicated as A, homozygousity for parent B allele as B, heterozygousity (A/B or B/A) as H.

Value

A character vector with all genotypes matching the target configuration. The format complies with the output format of branchProbabilities (gamete1-gamete2, e.g., "AAB-AAA")

Author(s)

Frank Technow

Examples

```
getTargets("AHB")
```

haploProb — *Probability of observing the target haplotype given the parental genotype.*

Description

These functions derive the symbolic formula for the probability of observing the target haplotype given the parental genotype.

Usage

```
haploProb.2M(gam1,gam2,target) ## two-marker genotypes
haploProb.3M(gam1,gam2,target) ## three-marker genotypes
```

Arguments

gam1, gam2	Three (three-marker genotypes) or two (two-marker genotypes) character string with the configuration of gamete one and two of the parental genotype.
target	Three (three-marker genotypes) or two (two-marker genotypes) character string with the configuration of the target haplotype.

Details

The idea behind the algorithm is to conceptually "recode" the alleles of the parental genotype into "target" and "non-target", where "target" is relative to the target haplotype. Then the rules are determined that would rearrange the gametes of the parental genotype into a "target-target-target" haplotype. These rearrangement rules are then translated into the symbolic formula.

Value

A character string with the symbolic formula. For three-marker genotypes, x is the recombination frequency between markers 1 and 2 and y that between markers 2 and 3. For two-marker genotypes, z is the recombination frequency between markers 1 and 2.

Author(s)

Frank Technow

Examples

```
haploProb.2M("AA","BB","AB")
haploProb.3M("AAA","BBB","ABA")
```

---

map *Genetic map of the three markers in the F4 data set.*

Description

Genetic map of the three markers in the F4 data set. The unit is Morgan. This map can be used to compute the recombination frequencies between the markers using the inverse of the Haldane mapping function.

Usage

```
map
```

Format

A numeric vector with three elements (c(0.00, 0.05, 0.20)).

References

Frank Technow(2013). hypred: Simulation of Genomic Data in Applied Genetics. R package version 0.4.

nodeProbabilities *Multiplies conditional probabilities along all branches of the tree*

Description

This function generates the symbolic formulas representing the marginal node probabilities.

Usage

```
nodeProbabilities(F,generation)
```

Arguments

F	A recurrent selfing tree, as generated by function buildSelfingTree.
generation	Integer giving the highest selfing generation contained in F. Values can range from 2 to 7, e.g., the F2 generation build by default and the highest possible generation is currently the F7.

Details

Each formula represents the marginal probability of a particular node. Summing over all nodes for a particular genotype gives the probability of observing this genotype in this generation. The sum over all marginal node probabilities within a generation is 1.

Value

A list with as many elements as there were generations in F. The list elements are named "F2", "F3", etc. Each element is a vector with the symbolic formulas for the marginal probabilities of all possible nodes. The vector elements are named and the names indicate the allelic configuration of the two gametes comprising the genotype as gamete1-gamete2 (e.g., "AAB-AAA").

Author(s)

Frank Technow

Examples

```
## F2 and F3 genotypes
node.probs <- nodeProbabilities(buildSelfingTree(genSubtree.2M,3,"AA","BB"),3)

## must sum to 1
stopifnot(all.equal(evalProb(node.probs[["F2"]],z = 0.045),1))
stopifnot(all.equal(evalProb(node.probs[["F3"]],z = 0.045),1))
```

Index