Package 'HoloSimR'

October 11, 2024

Type Package

Title Simulating a selection process under the hologenome scenario

Version 0.1.0

Author Cristina Casto-Rebollo [aut, cre], Ivan Porcnic [ctb], Gregor Gorjanc [ctb], Noelia Ibáñez-Escriche [ctb]

Maintainer Cristina Casto-Rebollo <3ccasto@gmail.com>

Description This simulator develops a selection process that simulates the co-evolution of the host genome and the microbiota. Allowed species are rabbit, cattle and a GENERIC genome. The user can include their own demographic population history using the AlphaSimR package. Microbiota are modeled considering the host genetic effect (microbial heritability), the symbiosis (microbial species interaction) and the environment.

License What license is it under?

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Depends AlphaSimR,
MASS,
dplyr,
utils,
mobsim,
R6,
stats,
ggplot2

Contents

`acquiredSpecies` *Acquisition of Microbial Species*

Description

This function determines the initial count of microbial species colonizing an individual, accounting for both the environmental and maternal communities of microorganisms. When saved as *acquiredSp*, it can be accessed by other functions.

Usage

```
acquiredSpecies(pop,
                parentPop = NULL,
                founderM = NULL,
                sym = NULL,
                rndSeed = NULL,
                globalSP = NULL)
```

Arguments

`pop`	An object of class `Pop-class` generated with **AlphaSimR** representing the current population.
`parentPop`	An object of class `Pop-class` generated with **AlphaSimR** representing the parental population. If `NULL`, the function assumes it needs to compute the microbiota for the base population.
`founderM`	A list containing the details of the microbiota in the founder population, generated by `setFounderM`.
`sym`	A value of 1 to consider the symbiosis effect on microbiota. This value is required when the population is not the base population.
`rndSeed`	A seed value to initialize random sampling.
`globalSP`	An object of class `GlobalSP` containing the global parameters of the simulation. If assigned as `gSP`, the function will access it directly.

Value

A matrix representing the initial microbial species in the current population.

Examples

```
# Set global simulation parameters
gSP <- GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

# Define the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0,1))

# Define the trait
gSP$setTrait(meanP = 10, varP = 3, h2 = 0.10, m2 = 0.10)

# Create founder genetics
founderG <- setFounderG(globalSP = gSP)

# Set simulation genetic parameters
SP <- essentialSP(founder = founderG,
                  minSnpFreq = 0.05, nSnpChr = 10000)

# Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

# Create the founder population
founderPop <- newPop(founderG)

# Create base population
Pop <- randCross(founderPop,
                 nCrosses = gSP$nDam,
                 nProgeny = gSP$nSon)

# Generate acquisition matrix for base population
acquiredSp <- acquiredSpecies(pop = Pop)
```

`essentialSP` *Essential Simulation Parameter Generator*

Description

This function generates an object of type `SimParam` with the essential parameters needed to perform the simulation. After running this function, other methods from `SimParam` can be used to define additional global parameters not included here. Saving this object as *SP* will allow it to be accessed by other functions with default parameters.

Usage

```
essentialSP(founder = NULL,
            minQTLchr = 1,
            nSnpChr = NULL,
            minSnpChr = NULL,
            minSnpFreq = NULL,
            rndSeed = NULL)
```

Arguments

founder	An object of class MapPop-class generated by the function setFounderG. If NULL, the function retrieves the founder haplotypes from the system.
minQTLchr	A value or vector specifying the minimum number of QTL per chromosome. *Default* is 1.
nSnpChr	A value or vector specifying the number of SNPs per chromosome. If NULL, the *Default* is 100,000 for "RABBIT" (based on the 200K Affymetrix Axiom OrcunSNP array *ThermoFisher Scientific*) and 1724 for "CATTLE" (based on the Bovine SNP50 v2 BeadChip from *Illumina*). For "GENERIC", a value must be specified.
minSnpChr	A value or vector specifying the number of segregating sites for SNPs per chromosome. If NULL, it is assumed to be half of *nSnpChr*.
minSnpFreq	The minimum allowable frequency for SNP loci. No minimum SNP frequency is applied if NULL.
rndSeed	A seed to initialize random sampling. If NULL, a seed is randomly set.

Value

An object of class SimParam.

Examples

```
## Not run:
# Set genetic parameters
SP <- essentialSP(founder = founderG, nSnpChr = 10000, minSnpFreq = 0.05)

## End(Not run)
```

---

extractM *Extracting Microbiota Composition*

Description

This function extracts the microbiota composition of a population and generates a *txt* file with the data.

Usage

```
extractM(pop,
                sym,
                globalSP = NULL,
                name = NULL,
                path = "./")
```

Arguments

pop	An object of class MapPop-class.
sym	A value of 1 to consider the symbiosis effect on the microbiota.
globalSP	An object of class GlobalSP. If NULL, it will be retrieved from the global environment.
name	The name of the *txt* file to be generated.
path	The path where the *txt* file will be saved. *Default* is the current directory ("./").

Value

A matrix representing the microbiota composition.

Examples

```
## Not run:
microbiota <- extractM(Pop, sym = 0, name = "M_gen3_sym")

## End(Not run)
```

---

fillSp *Fill Species Information*

Description

This function incorporates microbial species abundances and the microbiota value into an object of class Pop-class.

Usage

```
fillSp(pop, w = NULL, mbiome = NULL, sym = 0)
```

Arguments

pop	An object of class Pop-class generated using the **AlphaSimR** package, representing the current population.
w	A vector with the ω effect scaled according to the base population.
mbiome	A microbiota abundance matrix generated using the function makeM. If NULL, mbiome will be extracted from the global environment.
sym	A value of 1 to consider the symbiosis effect on the microbiota. *Default* is 0.

Value

An object of class Pop-class with updated microbiota information.

GlobalSP *Global Simulation Parameters*

Description

The 'GlobalSP' class represents global simulation parameters for a specific simulation scenario. It includes general parameters, trait parameters, and microbiota parameters.

Methods

Public methods

- `GlobalSP$new()` Create a new GlobalSP object.
- `GlobalSP$setTrait()` Set trait parameters for the GlobalSP object.
- `GlobalSP$setSpecies()` Set species parameters for the GlobalSP object.
- `GlobalSP$getValues()` Retrieve values from the GlobalSP object.

Public fields

`nPop` (numeric) Number of individuals in the founder population.

`nChr` (numeric) Total number of chromosomes to simulate using `runcMacs` from `AlphaSimR`.

`nQTLchr` (numeric) Number of QTL per chromosome affecting the simulated trait.

`segSITESchr` (numeric) Total number of segregating sites to simulate using `runcMacs` from `AlphaSimR`. If it is `NULL`, all are retained.

`nt` (numeric) If OpenMP is available, this allows for simulating chromosomes in parallel. *Default* is 1.

`nSim` (numeric) Number of repetitions of the simulation. *Default* is 1.

`animal` (character) Type of animal. Currently, three species histories are available: "GENERIC", "CATTLE", and "RABBIT". All except the rabbit demographic histories are implemented in **AlphaSimR**. The rabbit demographic is implemented in this package following *Carneiro et al. (2014)*.

`selType` (character) Selection type (e.g., "Divergent", "Low", "High").

`model` (character) Simulation model. There are five models implemented:

- "G": Effect of the genome only
- "M": Effect of microbiota only
- "NMH": Hologenome without microbial heritability (Null Microbial heritability)
- "LMH": Hologenome considering Low Microbial Heritability
- "MHM": Hologenome considering Medium Microbial Heritability
- "HMH": Hologenome considering High Microbial Heritability

All models can run simultaneously starting from the same base population if "All" is specified.

`nyear` (numeric) Number of years.

`nSire` (numeric) Number of sires.

`nDam` (numeric) Number of dams.

`nCross` (numeric) Number of crosses.

`nSon` (numeric) Number of progeny per cross.

`meanP` (numeric) Mean value for the simulated trait.

`varP` (numeric) Variance of the simulated trait.

`h2` (numeric) Heritability of the simulated trait.

`m2` (numeric) Microbiability for the simulated trait.

`nSpecies` (numeric) Total number of microbial species simulated in a community.

`nSp0` (numeric) Total number of simulated microbial species in the base population.

`nSpEff` (numeric) Number of microbial species affecting the simulated trait.

`PM` (numeric) Proportion of the initial parental microbiota (dam microbiota) exposed to the offspring.

`EM` (numeric) Proportion of the initial environmental microbiota exposed to the offspring.

`propMH` (numeric) Proportion of microbial species affected by the host genome.

`propQTL` (numeric) Proportion of QTLs affecting the abundance of the microbial species.

`propMH.wSp` (numeric) Proportion of microbial species with effects on the simulated trait that are affected by the host genome.

`MH.G` (numeric) Microbial heritability in the G scenario.

`MH.M` (numeric) Microbial heritability in the M scenario.

`MH.low` (numeric) Microbial heritability in the LMH scenario.

`MH.medium` (numeric) Microbial heritability in the MHM scenario.

`MH.high` (numeric) Microbial heritability in the HMH scenario.

`symbiosis` (numeric) Value or vector to specify whether to simulate (1) or not (0) the microbial species interaction or symbiosis effect on microbial abundance. There are five types of interactions simulated according to *Coyte KZ and Rakoff-Nahoum S (2019)*: Commensalism, Mutualism, Ammensalism, Competition, and Exploitation/Pathogen.

`s2` (numeric) Proportion of microbial species abundance due to the symbiosis or interaction between microbial species.

Methods

Public methods:

- `GlobalSP$new()`
- `GlobalSP$setTrait()`
- `GlobalSP$setSpecies()`
- `GlobalSP$getValues()`
- `GlobalSP$getPrivate()`
- `GlobalSP$clone()`

Method `new()`**:** Function to initialize the parameters needed for the simulation. Saving this object as *gSP* will facilitate the functionality of the other functions.

Usage:

```
GlobalSP$new(
  nPop,
  nChr = 1,
  nQTLchr,
  segSITESchr = NULL,
  animal = "GENERIC",
  selType = "Divergent",
```

```
  nyear,
  model = "NMH",
  nt = 1,
  nSim = 1
)
```

Arguments:

`nPop` The total number of individuals to simulate in the founder population.

`nChr` The total number of chromosomes to simulate using `runcMacs` from `AlphaSimR`. *Default* is 1.

`nQTLchr` A value or vector indicating the number of QTL per chromosome affecting the simulated trait.

`segSITEschr` The total number of segregating sites per chromosome to simulate using `runcMacs` from `AlphaSimR`. If it is *NULL*, all are retained.

`animal` The species to simulate. Can be "GENERIC", "CATTLE", or "RABBIT". All except the rabbit demographic histories are implemented in **AlphaSimR**.

`selType` The selection type (e.g., "Divergent", "Low", "High").

`nyear` The total number of years to simulate.

`model` The simulation model. There are five models implemented: "G", "M", "NMH", "LMH", "MHM", and "HMH".

`nt` If OpenMP is available, this allows for simulating chromosomes in parallel. *Default* is 1.

`nSim` The number of repetitions of the simulation. *Default* is 1.

Method `setTrait()`**:** Function to set trait parameters for the GlobalSP object.

Usage:

```
GlobalSP$setTrait(meanP, varP, h2, m2)
```

Arguments:

`meanP` The mean value for the trait.

`varP` The variance of the trait.

`h2` Heritability of the trait.

`m2` Microbiability of the trait.

Method `setSpecies()`**:** Function to set species parameters for the GlobalSP object.

Usage:

```
GlobalSP$setSpecies(
  nSpecies,
  nSp0,
  nSpEff,
  PM = 0.5,
  EM = 0.5,
  propMH = 0.1,
  propQTL = 0.1,
  propMH.wSp = 0.5,
  symbiosis = c(0, 1),
  s2 = 0.2,
  MH.G = 0.2,
  MH.M = 0.2,
  MH.low = 0.1,
  MH.medium = 0.3,
  MH.high = 0.6
)
```

Arguments:

`nSpecies` Total number of microbial species in the community.

`nSp0` Number of initial microbial species.

`nSpEff` Number of effective microbial species affecting the trait.

`PM` Proportion of parental microbiota exposed to offspring.

`EM` Proportion of environmental microbiota exposed to offspring.

`propMH` Proportion of species influenced by the host genome.

`propQTL` Proportion of QTLs affecting the microbial species.

`propMH.wSp` Proportion of species affecting the trait influenced by the genome.

`symbiosis` Whether to simulate symbiotic effects (1) or not (0).

`s2` Proportion of microbial abundance explained by symbiosis.

`MH.G` Microbial heritability in the G scenario.

`MH.M` Microbial heritability in the M scenario.

`MH.low` Microbial heritability in the LMH scenario.

`MH.medium` Microbial heritability in the MHM scenario.

`MH.high` Microbial heritability in the HMH scenario.

Method `getValues()`**:** Function to retrieve the values of the GlobalSP object.

Usage:

```
GlobalSP$getValues()
```

Method `getPrivate()`**:** Function to retrieve private parameters.

Usage:

```
GlobalSP$getPrivate()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GlobalSP$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

References

Coyte KZ, Rakoff-Nahoum S. Understanding competition and cooperation within the mammalian gut microbiome. Curr Biol. 2019;29:R538-R544

Examples

```
# Set simulation parameters
gSP <- GlobalSP$new(nPop = 1000, nQTLchr = 100, nyear = 10)

# Defining the trait
gSP$setTrait(meanP = 8, varP = 2, h2 = 0.15, m2 = 0.15)

# Defining the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 35)
```

makeM *Compute Microbiota Abundances*

Description

The microbiota abundance is computed according to the variance set by the user in the object GlobalSP.

Usage

```
makeM(pop,
      sym = 0,
      founderM = NULL,
      acquiredSp = NULL,
      rndSeed = NULL,
      globalSP = NULL)
```

Arguments

pop	An object of MapPop-class or a list from simBasePop.
sym	A value of 1 to consider the symbiosis effect on microbiota. *Default* is 0.
founderM	A list from setFounderM. If NULL, it will be obtained from the global environment.
acquiredSp	A matrix of the starting microbial species in the current population from acquiredSpecies. If NULL, it will be obtained from the global environment.
rndSeed	A seed for random sampling. If it is NULL, a random seed will be set.
globalSP	An object of class GlobalSP. If NULL, it will be obtained from the global environment.

Details

The microbiota abundance is computed considering that the variability of each microbial species is due to: $\text{var}_{m_k} = \text{var}_{hg_k} + \text{var}_{s_k} + \text{var}_{e_k}$, where var_{m_k} is the total abundance variance of species k, var_{hg_k} is the variance due to the host genetic effect on k, var_{s_k} is the variance of the symbiosis effect, and var_{e_k} is the residual variance of species k.

The microbiota abundance is considered to be mediated by two components: a stable abundance generated by the host genetics and the environment (x_{ik_0}), and the abundance due to the symbiosis in this stable abundance: $x_{ik} = (x_{ik})_0 + (x_{ik})_s$,

$x_{ik_0} = \mu_k + \sum_{j=1}^{n} z_{ij}\beta_{jk} + e_{ik}$, where x_{ik_0} is the stable abundance of individual i and species k, μ_k is the mean abundance for species k, z_{ij} is the allele dosage of individual i for SNP j, β_{jk} is the host genetic effect of SNP j on the species abundance k, and e_{ik} is the residual. This is based on *Pérez-Enciso et al. (2021)*.

$(x_{ik})_s = (x_i)_0'\gamma_k$, where $(x_{ik})_s$ is the abundance due to the symbiosis effect for individual i and species k, and $(x_i)_0'$ is the vector of the stable microbial abundance of individual i.

Value

A MapPop-class object that includes microbiota abundance information (M) and the microbiota value (mv) for each individual, according to fillSp.

References

Pérez-Enciso M, Zingaretti LM, Ramayo-Caldas Y, et al. Opportunities and limits of combining microbiome and genome data for complex trait prediction. Genet Sel Evol. 2021;53:65. https://doi.org/10.1186/s12711-021-00658-7

Examples

```
# Set simulation global parameters
gSP <- GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

# Define the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0, 1))

# Define the trait
gSP$setTrait(meanP = 10, varP = 3, h2 = 0.10, m2 = 0.10)

# Create founder genetic
founderG <- setFounderG(globalSP = gSP)

# Set simulation genetic parameters
SP <- essentialSP(founder = founderG, minSnpFreq = 0.05, nSnpChr = 10000)

# Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

# Create the founder population
founderPop <- newPop(founderG)

# Create the base population
basePop <- simBasePop(model = "NMH")

# Create a new population from the base population
Pop <- randCross(basePop$Pop, nCrosses = gSP$nDam, nProgeny = gSP$nSon)

# Make acquisition matrix of the new population
acquiredSp <- acquiredSpecies(pop = Pop)

# Compute the microbiota
Pop <- makeM(pop = Pop, globalSP = gSP)
```

---

makeP *Compute Phenotypes*

Description

This function computes the phenotype of a population according to a specified scenario and generation.

Usage

```
makeP(pop,
      model,
```

```
      sym = 0,
      limTrait = NULL,
      sex = "both",
      globalSP = NULL,
      rndSeed = NULL)
```

Arguments

pop	An object of MapPop-class for computing the phenotype.
model	A string indicating the model to simulate. Only the values "G", "M", "NMH", "LMH", "MMH", or "HMH" are available.
sym	A value of 1 to consider the symbiosis effect on microbiota. *Default* is 0.
limTrait	A vector with the minimum and maximum values for the trait. If NULL, it will be obtained from the global environment.
sex	Required if the phenotype is attributed to only one sex (e.g., litter size). Use "M" for male and "F" for female. Default is "both" for both sexes.
globalSP	An object of class GlobalSP. If NULL, it will be obtained from the global environment.
rndSeed	A seed for random sampling. If it is NULL, a random seed will be set.

Value

An object of MapPop-class that includes the computed phenotypes.

Examples

```
# Set simulation global parameters
gSP <- GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

# Define the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0, 1))

# Define the trait
gSP$setTrait(meanP = 10, varP = 3, h2 = 0.10, m2 = 0.10)

# Create founder genetic
founderG <- setFounderG(globalSP = gSP)

# Set simulation genetic parameters
SP <- essentialSP(founder = founderG, minSnpFreq = 0.05, nSnpChr = 10000)

# Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

# Create the founder population
founderPop <- newPop(founderG)

# Create the base population
basePop <- simBasePop(model = "NMH")

# Compute phenotype of the base population
Pop <- makeP(pop = basePop$Pop, model = "NMH", sym = 1, limTrait = c(0, 18), sex = "F")
```

makeSelection *Perform Selection*

Description

This function automates the selection process across different generations and returns a report of class retValues.

Usage

```
makeSelection(g = NULL,
              model = NULL,
              selType = NULL,
              sym = NULL,
              limTrait = NULL,
              sex = "both",
              LS = FALSE,
              globalSP = NULL,
              save = c(),
              iter = 1,
              progressBar = FALSE)
```

Arguments

g	Number of generations for selection. If NULL, it will be obtained from the global environment.
model	A value for the model to simulate. Only values according to "G", "M", "NMH", "LMH", "MMH", "HMH", "All" are available. If NULL, it will be obtained from the global environment.
selType	Selection type (e.g., "Divergent", "Low", "High"). If NULL, it will be obtained from the global environment.
sym	A value of 1 to consider the symbiosis effect on microbiota. If NULL, it will be obtained from the global environment.
limTrait	Vector with the minimum and maximum values for the trait. If NULL, it will be obtained from the global environment.
sex	Indicates if the phenotype belongs exclusively to females ("F"). Default is "both".
LS	TRUE if the phenotype being evaluated is litter size (LS). Default is FALSE.
globalSP	An object of class GlobalSP. If NULL, it will be obtained from the global environment.
save	If not null, the base and last population of the simulation will be saved.
iter	Number of iterations to perform. If NULL, just one iteration will be done.
progressBar	If TRUE, a progress bar and text will be displayed. *Default* is FALSE.

Value

A list of class `retValues` with the following information:

1. **Generation**: Generation computed
2. **Model**: Model computed
3. **iter**: Iteration performed. In case the selection will be replicated
4. **P**: Average of the population phenotype
5. **varP**: Variance of the phenotype
6. **gv**: Average of the additive genetic value
7. **varG**: Genetic variance
8. **S_gv**: Selection differential for additive genetic values
9. **mv**: Average of the microbiome values
10. **varM**: Variance of the microbiome values
11. **S_mv**: Selection differential for the microbiome values

Examples

```
# Set simulation global parameters
gSP = GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

# Define the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0,1))

# Define trait
gSP$setTrait(meanP = 10, varP = 3, h2 = 0.10, m2 = 0.10)

# Create founder genetic
founderG = setFounderG(globalSP = gSP)

# Set simulation genetic parameters
SP <- essentialSP(founder = founderG,
                  minSnpFreq = 0.05, nSnpChr = 10000)

# Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

# Create the founder population
founderPop <- newPop(founderG)

# Initialize the selection process
selection <- makeSelection(sex = "F", LS = TRUE, limTrait = c(0,18))
```

`matingPLAN` *Create Mating Plan*

Description

Creates crosses based on the selected breeding animals using `selectBreeding`.

Selection of breeding animals is according to the Phenotype ("phe") or the direct genetic value ("gv").

Usage

```
matingPLAN(parent,
           nCross = NULL,
           selType = NULL,
           globalSP = NULL)
```

Arguments

parent	List of selected breeding animals obtained using the function selectBreeding.
nCross	Number of females mating with each male. If NULL, it will be obtained from the global environment.
selType	Selection type (e.g., "Divergent", "Low", "High"). If NULL, it will be obtained from the global environment.
globalSP	An object of class GlobalSP. If NULL, it will be obtained from the global environment.

Value

A list including two dataframes with the mating plans for increasing and decreasing the phenotype:

1. **PopLow**: Mating plan for decreasing the phenotype. ID of breeding females (Dam) and males (Sire), and value for the genetic or phenotypic value (Sel_value).
2. **PopHigh**: Mating plan for increasing the phenotype. ID of breeding females (Dam) and males (Sire), and value for the genetic or phenotypic value (Sel_value).

Examples

```
# Set simulation global parameters
gSP = GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

# Define the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0,1))

# Define trait
gSP$setTrait(meanP = 10, varP = 3, h2 = 0.10, m2 = 0.10)

# Create founder genetic
founderG = setFounderG(globalSP = gSP)

# Set simulation genetic parameters
SP <- essentialSP(founder = founderG,
                  minSnpFreq = 0.05, nSnpChr = 10000)

# Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

# Create the founder population
founderPop <- newPop(founderG)

# Create the base population
basePop <- simBasePop(model = "NMH")

# Compute phenotype in a base population
Pop <- makeP(pop = basePop$Pop, model = "NMH", sym = 1, limTrait = c(0,18), sex = "F")
```

```
# Select breeding animals from basePop
parent <- selectBreeding(pop = Pop, nDam = 125, nSire = 25, sex = "F", g0 = TRUE)

# Create mating plan for next generation
crossPlan <- matingPLAN(parent = parent)
```

---

nextPop *Generate the Next Population*

Description

This function simulates animals for the next generation. A mating plan of class matingPLAN is required.

Usage

```
nextPop(pop,
        crossPlan,
        selType = NULL,
        nSon = NULL,
        LS = FALSE,
        globalSP = NULL)
```

Arguments

pop	An object of MapPop-class. If NULL, it will be obtained from the global environment (Pop).
crossPlan	A list or data.frame of type matingPLAN with the crosses to perform. If NULL, it will be obtained from the global environment.
selType	Selection type (e.g., "Divergent", "Low", "High"). If NULL, it will be obtained from the global environment.
nSon	Number of offspring per cross. If NULL, it will be obtained from the global environment.
LS	TRUE if the phenotype being evaluated is litter size (LS). Default is FALSE.
globalSP	An object of class GlobalSP. If NULL, it will be obtained from the global environment.

Value

A list including two objects of MapPop-class for the population with high ("PopHigh") and low ("PopLow") phenotypes.

Examples

```
# Set simulation global parameters
gSP = GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

# Define the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0,1))
```

```
# Define trait
gSP$setTrait(meanP = 10, varP = 3, h2 = 0.10, m2 = 0.10)

# Create founder genetic
founderG = setFounderG(globalSP = gSP)

# Set simulation genetic parameters
SP <- essentialSP(founder = founderG,
                  minSnpFreq = 0.05, nSnpChr = 10000)

# Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

# Create the founder population
founderPop <- newPop(founderG)

# Create the base population
basePop <- simBasePop(model = "NMH")

# Compute phenotype in a base population
Pop <- makeP(pop = basePop$Pop, model = "NMH", sym = 1, limTrait = c(0,18), sex = "F")

# Select breeding animals from basePop
parent <- selectBreeding(pop = Pop, nDam = 125, nSire = 25, sex = "F", g0 = TRUE)

# Create mating plan for next generation
crossPlan <- matingPLAN(parent = parent)

# Generate new generation
Pop <- nextPop(pop = Pop, crossPlan = crossPlan, nSon = 8, LS = TRUE)
```

---

plotREND *Plot Trend*

Description

plotREND generates a linear plot of genetic, phenotypic, and microbiome trends across generations for one scenario or for all, focusing on a single effect.

Usage

```
plotREND(data,
                plot = "line",
                draw = "effect",
                model = NULL,
                effect = "P",
                type_se = "classic",
                data2 = NULL,
                selType = "High",
                name = NULL,
                path = "./",
                show = FALSE,
                dpi = 600,
```

```
                width = 17,
                height = 12,
                units = "cm")
```

Arguments

data	A data.frame from the retValues object.
plot	The type of plot to generate ("boxplot" or "line"). The default is "line", which creates trends for all effects within a single model when draw is set to "Model" (the model parameter is required). To plot trends of all models for a specific effect, use the "Effect" option in draw. The default is "P", which plots the phenotype for all evaluated models. To change it, use the effect argument with "gv" or "mv". For boxplot, just data is required.
model	The model for which to generate the plot line for draw option equal to "Model".
effect	The effect to plot when plot is set to "Effect". The default is "P" for Phenotype. Other options are "gv" for genetic values and "mv" for microbiota values.
type_se	Graph style for plotting the standard error ("ribbon" or "classic"). Default is "classic".
data2	A data.frame from the retValues object. Required if selType is set to "Divergent".
selType	The population to plot. The default is "High".
name	The name of the graph file.
path	The path where the graph file will be saved.
show	If TRUE, the plot will be displayed in the current R session.
dpi	The resolution of the image. The default is 600 dpi.
width	The width of the image. The default is 17 cm.
height	The height of the image. The default is 12 cm.
units	The units for the width and height.

Value

A graph file and a ggplot2 object.

Examples

```
## Not run:
plotREND(data=selection[["Report"]][["PopHigh"]]$Symbiosis,
         plot = "boxplot",
         selType = "High",
         show = TRUE,
         name = "lastG_boxplot_symbiosis_high.tiff")

## End(Not run)
```

`retValues` *Retrieve Simulation Values*

Description

Retrieve information from the simulation process, including:

1. **Generation**: Computed generation
2. **Model**: Computed model
3. **iter**: Iteration performed; relevant if the selection is replicated
4. **P**: Average phenotype of the population
5. **varP**: Phenotypic variance
6. **gv**: Average additive genetic value
7. **varG**: Genetic variance
8. **S_gv**: Mean genetic value of the breeding animals
9. **mv**: Average microbiota value
10. **varM**: Variance of microbiota value
11. **S_mv**: Mean microbiota value of the breeding animals

Usage

```
retValues(model,
          g,
          sym = 0,
          selType,
          iter = 1,
          pop = NULL,
          breedInd = NULL)
```

Arguments

`model`	The model value to simulate. Only values corresponding to "G", "M", "NMH", "LMH", "MMH", or "HMH" are available.
`g`	Current generation of selection. A value is required.
`sym`	A value of 1 to consider the symbiosis effect on microbiota. *Default* is 0.
`selType`	Selection type ("Low", "High"). A value is required.
`iter`	Current iteration. *Default* is 1.
`pop`	An object of class `MapPop-class`. If `NULL`, it will be obtained from the environment (`Pop`).
`breedInd`	A list of selected breeding animals obtained using the function `selectBreeding`. If `NULL`, it will be obtained from the environment (`parent`). A list of type `selectBreeding` can be used.

Value

A list containing two sub-lists with data frames that include the above information for the populations with high ("PopHigh") and low ("PopLow") phenotypes. Each population has a data frame for values with ("Symbiosis") and without ("No_symbiosis") the symbiosis effect.

Examples

```
#Set simulation global parameters
gSP = GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

#Defining the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0,1))

#Defining trait
gSP$setTrait(meanP = 10,varP=3,h2=0.10,m2=0.10)

#Create founder genetic
founderG = setFounderG(globalSP = gSP)

#Set simulation genetic parameters
SP <- essentialSP(founder = founderG,
                  minSnpFreq = 0.05,rndSeed=NULL, nSnpChr = 10000)

#Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

#Create the founder population
founderPop <- newPop(founderG)

#Create the base population
basePop <- simBasePop(model = "NMH")

#Computing phenotype in a base population
Pop <- makeP(pop = basePop$Pop, model = "NMH", sym = 0, limTrait = c(0,18), sex = "F")

#Selection breeding animals from basePop
parent <- selectBreeding(pop = Pop, nDam = 125, nSire = 25, sex = "F", g0 = TRUE)

#Retrieving values
report <- retValues(model = "NMH", g = 1, sym = 0, selType = "Low", pop=Pop, breedInd= parent)
```

`savePop` *Save Population*

Description

This function allows populations to be stored in a common list.

Usage

```
savePop(pop, model, g, save = NULL)
```

Arguments

`pop`	An object of class `MapPop-class`.
`model`	The model used for the simulation. Valid values are "G", "M", "NMH", "LMH", "MMH", or "HMH".
`g`	The current generation of selection. A value is required.

save — A list of previously saved populations. If a variable called *sortedPop* exists, it will be retrieved from the environment and used to continue storing the populations.

Value

A list of MapPop-class objects, ordered by generation.

Examples

```
## Not run:
# Saving the population
storedPop <- savePop(Pop, model = "NMH")

## End(Not run)
```

---

selectBreeding *Selection of Breeding Animals*

Description

Selection of breeding animals according to the Phenotype ("phe") or the direct genetic value ("gv").

Usage

```
selectBreeding(pop,
               nDam = NULL,
               nSire = NULL,
               selType = NULL,
               selby = "phe",
               sex = "both",
               maxFS = 2,
               g0 = FALSE,
               LS = FALSE,
               globalSP = NULL)
```

Arguments

pop	an object of MapPop-class
nDam	number of females to select. If NULL, it will be set by nDam from GlobalSP. Default 125.
nSire	number of males to select. If NULL, it will be set by nSire from GlobalSP. Default 25.
selType	Selection type (e.g., "Divergent", "Low", "High"). If NULL, it will be obtained from the global environment
selby	type of selection; Phenotypic selection ("phe"; Default) or Genomic selection ("gv")
sex	indicate if the phenotype belongs exclusively to females ("F"). Default "both"
maxFS	maximum number of full siblings allowed when selecting females.

g0	TRUE for indicating that the animals will be selected from the base population. Default FALSE
LS	TRUE if the phenotype being evaluated is litter size (LS). Default FALSE
globalSP	an object of class GlobalSP. If NULL, it will be obtained from the global environment.

Value

a list including four dataframes:

1. **Dam_low**: ID of breeding females for decreasing the phenotype, mother ID, father ID and value for the genetic value or phenotype, depending on the selby
2. **Dam_high**: ID of breeding females for increasing the phenotype, mother ID, father ID and value for the genetic value or phenotype, depending on the selby
3. **Male_low**: ID of breeding males for decreasing the phenotype, mother ID, and father ID
4. **Male_high**: ID of breeding males for increasing the phenotype, mother ID, and father ID

Examples

```
# Set simulation global parameters
gSP = GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

# Defining the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0,1))

# Defining trait
gSP$setTrait(meanP = 10, varP = 3, h2 = 0.10, m2 = 0.10)

# Create founder genetic
founderG = setFounderG(globalSP = gSP)

# Set simulation genetic parameters
SP <- essentialSP(founder = founderG,
                  minSnpFreq = 0.05, nSnpChr = 10000)

# Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

# Create the founder population
founderPop <- newPop(founderG)

# Create the base population
basePop <- simBasePop(model = "NMH")

# Computing phenotype in a base population
Pop <- makeP(pop = basePop$Pop, model = "NMH", sym = 1, limTrait = c(0,18), sex = "F")

# Selection of breeding animals from basePop
parent <- selectBreeding(pop = Pop, nDam = 125, nSire = 25, sex = "F", g0 = TRUE)
```

setFounderG *Generate Founder Population Demography*

Description

This function calls runMacs or runMacs2 from the **AlphaSimR** package to generate the founder haplotypes. For more details, refer to the documentation for those functions. This function allows the implementation of the rabbit's demographic history using runMacs2, based on *Carneiro et al. (2014)*. Saving this object as *founderGenomes* will enable access by default in other functions.

Usage

```
setFounderG(globalSP = NULL, rndSeed = NULL)
```

Arguments

globalSP	An object of class GlobalSP containing the global parameters of the simulation. If assigned as gSP, the function will access it directly.

Value

An object of class MapPop-class.

References

Carneiro M, Albert FW, Afonso S, Pereira RJ, Burbano H, Campos R, et al. The genomic architecture of population divergence between subspecies of the European rabbit. PLoS Genet. 2014;10:e1003519

Examples

```
# Set global simulation parameters
gSP <- GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

# Define the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0,1))

# Define the trait
gSP$setTrait(meanP = 10, varP = 3.8, h2 = 0.15, m2 = 0.15)

# Create founder genetics
founderG <- setFounderG(globalSP = gSP)
```

`setFounderM` *Generate Microbiota Architecture*

Description

Generation of the architecture for the parental and environmental microbiota at founder population. The **mobsim** package is used to help simulate the species abundance distribution. Saving it as `founderM` will facilitate access by other functions.

Usage

```
setFounderM(globalSP = NULL)
```

Arguments

`globalSP`	an object of `GlobalSP` with the global parameters of the simulation. If it is `NULL`, the globalSP will be extracted from the Global Environment

Value

a list including three dataframes:

1. **architecture**: Dataframe with the architecture of the microbiota summarized in the following columns:
   - `RA_EM`: Mean of the relative abundance of the environmental microbiota
   - `Species`: Artificial name for microbial species
   - `EM`: Mean of the real abundance of the environmental microbiota
   - `PM0`: Vector indicating which microbial species will form the microbiota of the base population
   - `PM`: Mean of the real abundance of the parental microbiota
   - `GR_PM`: Mean of the growth rate of the microbial species when belonging to the parental microbiota
   - `SD`: Standard deviation of the growth rate
   - `RA_PM0`: Mean of the relative abundance of the microbial species in the parental microbiota
   - `w`: Vector of the effect of species abundance on the trait
   - `beta`: Vector of 0 or 1 indicating the species abundance influenced by the host genome
2. **beta**: Dataframe with the values of the genetic effect on the abundance of each species. Dimensions $nQTL$ x $nSpecies$
3. **symbiosis**: Dataframe with the symbiosis effect among species. Dimensions $nSpecies$ x $nSpecies$

References

Pérez-Enciso M, Zingaretti LM, Ramayo-Caldas Y et al. Opportunities and limits of combining microbiome and genome data for complex trait prediction. Genet Sel Evol. 2021;53:65. doi:10.1186/s12711021006587

Caballero A, Tenesa A, Keightley PD. The nature of genetic variation for complex traits revealed by GWAS and regional heritability mapping analyses. Genetics. 2015;201:1601–13. doi:10.1534/genetics.115.177220

Duvallet C, Gibbons SM, Gurry T et al. Meta-analysis of gut microbiome studies identifies disease-specific and shared responses. Nat Commun 2017;8:1784. doi:10.1038/s41467017019738

Examples

```
# Set simulation parameters
gSP <- GlobalSP$new(nPop = 1000, nQTLchr = 100, nyear = 10)

# Defining the microbiome
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 35)

# Making microbiome architecture
founderM <- setFounderM(globalSP = gSP)
```

`simBasePop` *Simulation base population*

Description

Simulation of the base population according to the parameters specified in the `GlobalSP` and the genome and microbiota created for the founder population with `setFounderG` and `setFounderM`. This function sets the base population, scales the host genetic (β), symbiosis (γ) and microbiota (ω) effects according to the variance specified by the user, and simulates the microbiota in the base population.

Usage

```
simBasePop(model,
           founderPop = NULL,
           founderM = NULL,
           globalSP = NULL,
           rndSeed=NULL,
           progressBar = FALSE)
```

Arguments

`model`	a value for the model to simulate. Only one value is available according to "G", "M", "NMH", "LMH", "MMH", "HMH".
`founderM`	a list of `setFounderM`. If `NULL`, it will be retrieved from the global environment.
`globalSP`	an object of class `GlobalSP`. If `NULL`, it will be retrieved from the global environment.
`rndSeed`	rndSeed seed to initialise the random sampling. If it is `NULL`, it will be set randomly.
`progressBar`	progressBar If TRUE, progress bar and text will be displayed. *Default* `FALSE`
`foundePop`	Founder population to use for founding the base population. If it is `NULL`, it will be taken from the environment.

Details

The β (host genetic) and γ (symbiosis) effect on the microbiota were scaled according to the variance set by the user in the object GlobalSP. The scale values were obtained by applying a scale factor follows $\sqrt{\frac{\text{var}_{\text{set}}}{\text{var}_{\text{sim}}}}$ where var_{set} is the variance set according the parameters on GlobalSP, and var_{set} is the variance on the simulated matrix (setFounderM), to be adjusted.

Value

a list including three dataframes:

1. **Pop**: an object of MapPop-class the base population
2. **Beta_scale**: a matrix with the scaled β effect
3. **Symbiosis_scale**: a matrix with the scaled $gamma$ effect. It is NULL if there is no symbiosis.
4. **varG**: a vector with the genetic variance of each microbial species.
5. **varS**: a vector with the symbiosis variance of each microbial species. It is NULL if there is no symbiosis.
6. **varE**: a vector with the environmental variance of each microbial species.
7. **w_scale0**: a vector with the scaled ω effect without taking the symbiosis effect into account.
8. **w_scale1**: a vector with the scaled ω effect including the symbiosis effect. It is NULL if there is no symbiosis.

Examples

```
#Set simulation global parameters
gSP = GlobalSP$new(nPop = 1000, nyear = 10, nQTLchr = 100)

#Defining the microbiota
gSP$setSpecies(nSpecies = 1000, nSp0 = 600, nSpEff = 100, symbiosis = c(0,1))

#Defining trait
gSP$setTrait(meanP = 10,varP=3,h2=0.10,m2=0.10)

#Create founder genetic
founderG = setFounderG(globalSP = gSP)

#Set simulation genetic parameters
SP <- essentialSP(founder = founderG,
                  minSnpFreq = 0.05,rndSeed=NULL, nSnpChr = 10000)

#Create founder microbiota
founderM <- setFounderM(globalSP = gSP)

#Create the founder population
founderPop <- newPop(founderG)

#Create the base population
basePop <- simBasePop(model = "NMH")
```

`summarySel` *Summary of a generation of selection*

Description

Extract the mean, standard deviation and range of genetic, microbiota and phenotypic valuesof the scenarios, taking into account all iterations performed.

Usage

```
summarySel(data, g)
```

Arguments

`data`	data.frame from `retValues` object.
`g`	generation for calculating the statistics

Value

a data.frame with the following columns:

1. **Generation**: Generation computed
2. **Model**: Scenario computed
3. **Gv_mean**: Model computed
4. **Gv_sd**: Iteration performed. In case the selection will be replicated
5. **Gv_range**: Average of the population phenotype
6. **Gvba_mean**: Mean of the microbiota value for the breeding animals
7. **Gvba_sd**: Standard deviation of the microbiota value for the breeding animals
8. **Gvba_range**: Range of the microbiota value for the breeding animals
9. **Mv_mean**: Variance of the phenotype
10. **Mv_sd**: Average of the additive genetic value
11. **Mv_range**: Genetic variance
12. **Mvba_mean**: Mean of the microbiota value for the breeding animals
13. **Mvba_sd**: Standard deviation of the microbiota value for the breeding animals
14. **Mvba_range**: Range of the microbiota value for the breeding animals
15. **P_mean**: phenotypic mean
16. **P_sd**: standard deviation of the phenotype
17. **P_range**: range of the phenotype
18. **Pba_mean**: Mean of the phenotypic value for the breeding animals
19. **Pba_sd**: Standard deviation of the phenotypic value for the breeding animals
20. **Pba_range**: Range of the phenotypic value for the breeding animals

Examples

```
## Not run:
summary_lastg <- summarySel(selection$Report$PopHigh$Symbiosis, globalSP$nyear)

## End(Not run)
```

Index