Package 'resevol'

January 6, 2022

Type Package

Title Simulate Agricultural Production and Evolution of Pesticide Resistance

Version 0.2.0.9

Imports stats(>= 4.0.0), utils (>= 4.0.0)

Maintainer A. Bradley Duthie <brad.duthie@gmail.com>

Description Simulates individual-based models of agricultural pest management and the evolution of pesticide resistance. Management occurs on a spatially explicit landscape that is divided into an arbitrary number of farms that can grow one of up to 10 crops and apply one of up to 10 pesticides. Pest genomes are modelled in a way that allows for any number of pest traits with an arbitrary covariance structure that is constructed using an evolutionary algorithm in the mine_gmatrix() function. Simulations are then run using the run_farm_sim() function. This package thereby allows for highly mechanistic social-ecological models of the evolution of pesticide resistance under different types of crop rotation and pesticide application regimes.

URL https://bradduthie.github.io/resevol/

BugReports https://github.com/bradduthie/resevol/issues

Depends R (>= 4.0.0)

License GPL (>= 2)

LazyData TRUE

Encoding UTF-8

VignetteBuilder knitr

Suggests knitr, rmarkdown, testthat, markdown

RoxygenNote 7.1.1

NeedsCompilation yes

Author A. Bradley Duthie [aut, cre] (<https://orcid.org/0000-0001-8343-4995>),
Rose McKeon [aut, ctr],
Rosie Mangan [ctr],
Luc Bussiére [ctr],
Matthew Tinsley [ctr]

Repository CRAN

Date/Publication 2022-01-06 18:10:02 UTC

R topics documented:

initialise_inds 2
make_landscape 5
mg_n1 6
mg_n2 6
mg_n3 7
mg_n4 7
mg_n5 7
mg_v1 8
mg_v2 8
mg_v3 8
mg_v4 9
mg_v5 9
mine_gmatrix 10
run_farm_sim 11

Index **17**

`initialise_inds` *Initialise individuals*

Description

Initialise new individuals into the IBM. This function is generally not needed because it is run inside the run_farm_sim function to generate new individuals for simulations. To initialise individuals with this function, it is necessary to set the mine_output argument to output from the mine_gmatrix function. This output includes all of the information necessary to build individuals with genomes that produce traits that covary in a pre-specified way. The arguments of this function include addition information for building the individual array, which is a two-dimensional array in which each individual occupies a row, and each column specifies a character of the individual (including all genome loci). See vignettes for a more detailed explanation.

Usage

```
initialise_inds(
  mine_output,
  N = 1000,
  xdim = 100,
  ydim = 100,
  repro = "sexual",
  neutral_loci = 10,
  max_age = 9,
  min_age_move = 0,
```

```
  max_age_move = 9,
  min_age_reproduce = 0,
  max_age_reproduce = 9,
  min_age_feed = 0,
  max_age_feed = 9,
  food_consume = 0.25,
  pesticide_consume = 0.1,
  rand_age = FALSE,
  move_distance = 1,
  food_needed_surv = 0.25,
  pesticide_tolerated_surv = 0.1,
  food_needed_repr = 0,
  pesticide_tolerated_repr = 0,
  reproduction_type = "lambda",
  mating_distance = 1,
  lambda_value = 1,
  movement_bouts = 1,
  selfing = TRUE,
  feed_while_moving = FALSE,
  pesticide_while_moving = FALSE,
  mortality_type = 0,
  age_food_threshold = NA,
  age_pesticide_threshold = NA,
  metabolism = 0,
  baseline_metabolism = 0,
  min_age_metabolism = 1,
  max_age_metabolism = 9
)
```

Arguments

mine_output	The output from mine_gmatrix
N	Number of individuals to be initialised
xdim	Horizontal dimensions of the landscape
ydim	Vertical dimensions of the landscape
repro	Type of reproduction allowed: "asexual", "sexual", and "biparental". Note that if repro != "asexual", this causes a diploid genome.
neutral_loci	The number of neutral loci individuals have (must be > 0)
max_age	The maximum age of an individual
min_age_move	The minimum age at which an individual can move
max_age_move	The maximum age at which an individual can move
min_age_reproduce	The minimum age which an individual can reproduce
max_age_reproduce	The maximum age which an individual can reproduce
min_age_feed	The minimum age at which an individual feeds

`max_age_feed` The maximum age at which an individual feeds

`food_consume` The amount of food consumed during feeding

`pesticide_consume`
Amount of pesticide consumed while on a cell

`rand_age` Initialise individuals with a random age (TRUE/FALSE)

`move_distance` Maximum cells moved in one bout of movement

`food_needed_surv`
Food needed to survive (if over min_age_feed)

`pesticide_tolerated_surv`
Pesticide tolerated by individual

`food_needed_repr`
Food needed to reproduce 1 offspring

`pesticide_tolerated_repr`
Pesticide tolerated to allow reproduction

`reproduction_type`
Poisson reproduction ("lambda") vs "food_based"

`mating_distance`
Distance in cells within which mate is available

`lambda_value` individual value for poisson reproduction

`movement_bouts` Number of bouts of movement per time step

`selfing` If sexual reproduction, is selfing allowed? (TRUE/FALSE)

`feed_while_moving`
Do individuals feed after each movement bout?

`pesticide_while_moving`
Individuals consume pesticide after move bout?

`mortality_type` Type of mortality (currently only one option)

`age_food_threshold`
Age at which food threshold is enacted

`age_pesticide_threshold`
Age at which pesticide threshold is enacted

`metabolism` The amount of consumed food lost each time step

`baseline_metabolism`
A fixed baseline rate added to 'metabolism'+

`min_age_metabolism`
The minimum age affected by metabolism

`max_age_metabolism`
The maximum age affected by metabolism

Value

A two-dimensional array of individuals for simulation

Examples

```
gmt       <- matrix(data = 0, nrow = 2, ncol = 2);
diag(gmt) <- 1;
mg        <- mine_gmatrix(gmatrix = gmt, loci = 4, layers = 2, indivs = 100,
                          npsize = 100, max_gen = 4, prnt_out = FALSE);
inds      <- initialise_inds(mine_output = mg, N = 40, repro = "asexual");
```

`make_landscape` *Landscape initialisation*

Description

Initialise the landscape for a simulation. This should not normally need to be done explicitly with this function because it is run inside of the run_farm_sim function, but this gives the option to generate a landscape without actually running a simulation. All landscapes are produced as three dimensional arrays with varying numbers of rows and columns that determine landscape size, and a depth of 21 layers. The top layer defines which cells belong to which farm, while the remaining layers define how much of a given crop is on the landscape cell (2-11) or how much pesticide has been applied to it (12-21). An arbitrary number of farms are placed in a blocked design on the landscape using a shortest split-line algorithm, which attempts to make farm size as even as possible. Specifying public land is possible, and adds sections of land that are not farms, but this is not recommended.

Usage

```
make_landscape(
  rows,
  cols,
  depth = 21,
  farms = 4,
  public_land = 0,
  farm_var = 0
)
```

Arguments

`rows`	The dimension of the other side of the landscape (e.g., Longitude)
`cols`	The dimension of one side of the landscape (e.g., Latitude)
`depth`	The number of layers in the 3D landscape
`farms`	The number of farms on the landscape
`public_land`	The proportion of landscape cells that are not farmland
`farm_var`	Does the land distribution vary among farms (>=0, <1)

Value

the_land A cols by rows landscape with randomly distributed cell types

Examples

```
land <- make_landscape(rows = 10, cols = 10, depth = 2, farms = 4)
```

`mg_n1` *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure of the identity matrix (replicate 1)

Usage

```
mg_n1
```

Format

A list of 8 elements used in individual-based model initialisation

---

`mg_n2` *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure of the identity matrix (replicate 2)

Usage

```
mg_n2
```

Format

A list of 8 elements used in individual-based model initialisation

mg_n3 *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure of the identity matrix (replicate 3)

Usage

```
mg_n3
```

Format

A list of 8 elements used in individual-based model initialisation

mg_n4 *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure of the identity matrix (replicate 4)

Usage

```
mg_n4
```

Format

A list of 8 elements used in individual-based model initialisation

mg_n5 *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure of the identity matrix (replicate 5)

Usage

```
mg_n5
```

Format

A list of 8 elements used in individual-based model initialisation

mg_v1 *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure including negative and positive values (replicate 1)

Usage

```
mg_v1
```

Format

A list of 8 elements used in individual-based model initialisation

mg_v2 *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure including negative and positive values (replicate 2)

Usage

```
mg_v2
```

Format

A list of 8 elements used in individual-based model initialisation

mg_v3 *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure including negative and positive values (replicate 3)

Usage

```
mg_v3
```

Format

A list of 8 elements used in individual-based model initialisation

mg_v4 *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure including negative and positive values (replicate 4)

Usage

```
mg_v4
```

Format

A list of 8 elements used in individual-based model initialisation

mg_v5 *Sample mine_gmatrix outputs*

Description

Output from mine_gmatrix that maps random normals from 12 loci to four traits with a covariance structure including negative and positive values (replicate 5)

Usage

```
mg_v5
```

Format

A list of 8 elements used in individual-based model initialisation

`mine_gmatrix` *Mine G-matrices*

Description

Mine networks for establishing the link between genome and g-matrix. The output from this function is required to run individual-based simulations in the rest of the package. The key input to this function, 'gmatrix', is a (square) covariance matrix, with each row and column representing a trait for the individual-based model. This function will run an evolutionary algorithm to try to find a network that produces traits with the covariance structure of gmatrix from a set of random standard normal values. The network from loci values to trait values goes through a number of linked nodes to achieve this, and each generation tests the stress of the resulting network in terms of expected squared deviation of trait covariances from the input gmatrix. Simulations can take minutes to hours or longer, depending on parameters chosen and the number of traits. See vignettes for a more comprehensive explanation for what this function is doing.

Usage

```
mine_gmatrix(
  loci = 18,
  layers = 6,
  indivs = 1000,
  npsize = 2000,
  mu_pr = 0.05,
  mu_sd = 0.01,
  max_gen = 1000,
  pr_cross = 0.05,
  sampleK = 40,
  chooseK = 4,
  term_cri = -5.3,
  sd_ini = 0.1,
  use_cor = FALSE,
  prnt_out = TRUE,
  gmatrix
)
```

Arguments

`loci`	The number of loci that individuals in the model will have
`layers`	The number of layers in the network from loci to traits
`indivs`	The number of individuals to test the covariance matrix
`npsize`	The size of the network population in the evolutionary algorithm
`mu_pr`	The probability of a network value to mutate
`mu_sd`	The standard deviation of mutation effect size
`max_gen`	The maximum number of generations of the evolutionary algorithm

pr_cross	The probability of a crossover occurring for a network
sampleK	Number of networks sampled to take part in tournament selection
chooseK	Number of winners in tournament selection
term_cri	Stress criteria (ln) for evolutionary algorithm terminating
sd_ini	StDev of initialised networked values
use_cor	Compare correlation matrix rather than the covariance matrix
prnt_out	Print out progress showing stress for each generation
gmatrix	G-matrix that the evolutionary algorithm will match

Value

A list of eight elements that includes the following: (1) A vector of input parameters, (2) the pre-specified covariance matrix, (3) matrix defining the effects of loci values on the first layer of the network, (4) a three dimensional array link the first network layer to trait values, (5) a matrix of the marginal effect of each locus on each trait, (6) the mined covariance structure, (7) all network values to be inserted into individual genomes, and (8) the log stress of the mined matrix against the pre-specified matrix.

Examples

```
gmt       <- matrix(data = 0, nrow = 4, ncol = 4);
diag(gmt) <- 1;
mg        <- mine_gmatrix(gmatrix = gmt, loci = 4, layers = 3, indivs = 100,
                          npsize = 100, max_gen = 2, prnt_out = FALSE);
```

---

run_farm_sim *Initialise individuals and simulate farming*

Description

Initialises a new set of individuals and then simulates farming over time. This is the main function that runs individual-based simulations of crop and pesticide use and the evolution of pesticide resistance over time. To run this function, output from the mine_gmatrix function is required to specify the covariance structure of individual traits and individual genomes. The arguments to this function are used to initialise a landscape with the make_landscape function and initialise individuals with the initialise_inds function. After initialisation, the simulation continues for up to a set number of time steps (unless extinction occurs), and individuals on the landscape feed, encounter pesticide, move, reproduce, and die depending upon the arguments specified in this function. After a specified number of time steps, the crop or pesticide applied to a landscape cell can also change. The end result is an evolving population of individuals that express traits that can potentially affect fitness (e.g., food consumption, pesticide consumption, movement). Population level statistics are calculated by default and printed to a CSV, but individual level data (which includes all individual characteristics in a large table) need to be turned on because files can become extremely large (use print_inds with extreme caution and print_last with care).

Usage

```
run_farm_sim(
  mine_output,
  N = 1000,
  xdim = 100,
  ydim = 100,
  repro = "sexual",
  neutral_loci = 1000,
  max_age = 9,
  min_age_move = 0,
  max_age_move = 9,
  min_age_reproduce = 0,
  max_age_reproduce = 9,
  min_age_feed = 0,
  max_age_feed = 9,
  food_consume = 0.25,
  pesticide_consume = 0.1,
  rand_age = FALSE,
  move_distance = 1,
  food_needed_surv = 0.25,
  pesticide_tolerated_surv = 0.1,
  food_needed_repr = 0,
  pesticide_tolerated_repr = 0,
  reproduction_type = "lambda",
  mating_distance = 1,
  lambda_value = 1,
  movement_bouts = 1,
  selfing = TRUE,
  feed_while_moving = FALSE,
  pesticide_while_moving = FALSE,
  mortality_type = 0,
  age_food_threshold = 0,
  age_pesticide_threshold = 0,
  farms = 4,
  time_steps = 100,
  mutation_pr = 0,
  crossover_pr = 0,
  mutation_type = 0,
  net_mu_layers = 0,
  net_mu_dir = 0,
  mutation_direction = 0,
  crop_rotation_type = 2,
  crop_rotation_time = 1,
  pesticide_rotation_type = 2,
  pesticide_rotation_time = 1,
  crop_per_cell = 1,
  pesticide_per_cell = 1,
  crop_sd = 0,
```

```
  pesticide_sd = 0,
  crop_min = 0,
  crop_max = 1000,
  pesticide_min = 0,
  pesticide_max = 1000,
  crop_number = 2,
  pesticide_number = 1,
  print_inds = FALSE,
  print_gens = TRUE,
  print_last = FALSE,
  K_on_birth = 1e+06,
  pesticide_start = 0,
  immigration_rate = 0,
  get_f_coef = FALSE,
  get_stats = TRUE,
  metabolism = 0,
  baseline_metabolism = 0,
  min_age_metabolism = 1,
  max_age_metabolism = 9
)
```

Arguments

mine_output	The output from mine_gmatrix
N	Number of individuals to be initialised
xdim	Horizontal dimensions of the landscape
ydim	Vertical dimensions of the landscape
repro	Type of reproduction allowed: "asexual", "sexual", and "biparental". Note that if repro > 0, this causes a diploid genome.
neutral_loci	The number of neutral loci individuals have
max_age	The maximum age of an individual
min_age_move	The minimum age at which an individual can move
max_age_move	The maximum age at which an individual can move
min_age_reproduce	The minimum age which an individual can reproduce
max_age_reproduce	The maximum age which an individual can reproduce
min_age_feed	The minimum age at which an individual feeds
max_age_feed	The maximum age at which an individual feeds
food_consume	The amount of food consumed during feeding
pesticide_consume	Amount of pesticide consumed while on a cell
rand_age	Initialise individuals with a random age
move_distance	Maximum cells moved in one bout of movement

`food_needed_surv`
: Food needed to survive (if over min_age_feed)

`pesticide_tolerated_surv`
: Pesticide tolerated by individual

`food_needed_repr`
: Food needed to reproduce 1 offspring

`pesticide_tolerated_repr`
: Pesticide tolerated to allow reproduction

`reproduction_type`
: Poisson reproduction ("lambda") vs "food_based"

`mating_distance`
: Distance in cells within which mate is available

`lambda_value` individual value for poisson reproduction

`movement_bouts` Number of bouts of movement per time step

`selfing` If sexual reproduction, is selfing allowed?

`feed_while_moving`
: Do individuals feed after each movement bout?

`pesticide_while_moving`
: Individuals consume pesticide after move bout?

`mortality_type` Type of mortality (currently only one option)

`age_food_threshold`
: Age at which food threshold is enacted

`age_pesticide_threshold`
: Age at which pesticide threshold is enacted

`farms` How many farms should there be on the landscape?

`time_steps` Time steps in the simulation

`mutation_pr` Probability of a loci mutating

`crossover_pr` Probability of crossover at homologous loci

`mutation_type` Type of mutation used

`net_mu_layers` Layers of the network allowed to mutate

`net_mu_dir` Layers mutate from loci to (1) or traits back (0)

`mutation_direction`
: Is mutation directional (unlikely to need)

`crop_rotation_type`
: None (1) or random (2) rotation of crop type

`crop_rotation_time`
: How frequently are the crops rotated?

`pesticide_rotation_type`
: None (1) or random (2) rotation of pesticide

`pesticide_rotation_time`
: How frequently are the pesticides rotated?

`crop_per_cell` How much crop is put on a single cell?

`pesticide_per_cell`
: How much pesticide is put on a single cell?

`crop_sd`
: What is the standard deviation of crop on a cell?

`pesticide_sd`
: What is the standard deviation of pesticide on a cell?

`crop_min`
: What is the minimum crop amount allowed per cell?

`crop_max`
: What is the maximum crop amount allowed per cell?

`pesticide_min`
: What is the minimum pesticide amount allowed per cell?

`pesticide_max`
: What is the maximum pesticide amount allowed per cell?

`crop_number`
: How many crops exist on the landscape?

`pesticide_number`
: How many pesticides are applied on the landscape?

`print_inds`
: Should the full list of individuals be printed? (CAREFUL)

`print_gens`
: Should a summary of each time step be printed?

`print_last`
: Should the last time step of individuals be printed?

`K_on_birth`
: Is there a carrying capacity applied on newborns?

`pesticide_start`
: What time step should pesticide start being applied?

`immigration_rate`
: Mean number of immigrants per time step

`get_f_coef`
: Get the inbreeding coefficient (not for asexual)

`get_stats`
: Get population level statistics in a CSV printout

`metabolism`
: The rate at which food consumed is burned in a time step

`baseline_metabolism`
: A fixed baseline rate added to 'metabolism'

`min_age_metabolism`
: The minimum age affected by metabolism

`max_age_metabolism`
: The maximum age affected by metabolism

Value

The output in the R console is a list with two elements; the first element is a vector of parameter values used by the model, and the second element is the landscape in the simulation. The most relevant output will be produced as CSV files within the working directory. When get_stats = TRUE, a file named 'population_data.csv' is produced in the working directory. When print_last = TRUE, a complete array of all individuals and their characteristics is printed for the last time step in the working directory in a file named 'last_time_step.csv' (for large simulations, this file can be > 1GB in size). When print_inds = TRUE, a complete array of all individuals in all time steps is produced in the working directory in a file named 'individuals.csv' (use this option with extreme caution for all but the smallest simulations).

Examples

```
gmt       <- matrix(data = 0, nrow = 4, ncol = 4);
diag(gmt) <- 1;
mg        <- mine_gmatrix(gmatrix = gmt, loci = 4, layers = 3, indivs = 100,
                          npsize = 100, max_gen = 2, prnt_out = FALSE);
sim       <- run_farm_sim(mine_output = mg, N = 100, xdim = 40, ydim = 40,
                          repro = "asexual", time_steps = 1,
                          print_inds = FALSE, print_gens = FALSE,
                          print_last = FALSE, get_stats = FALSE);
```

Index

∗ **datasets**
mg_n1, 6
mg_n2, 6
mg_n3, 7
mg_n4, 7
mg_n5, 7
mg_v1, 8
mg_v2, 8
mg_v3, 8
mg_v4, 9
mg_v5, 9

initialise_inds, 2

make_landscape, 5
mg_n1, 6
mg_n2, 6
mg_n3, 7
mg_n4, 7
mg_n5, 7
mg_v1, 8
mg_v2, 8
mg_v3, 8
mg_v4, 9
mg_v5, 9
mine_gmatrix, 10

run_farm_sim, 11