

Package ‘mozzie’

May 4, 2019

Type Package

Title Agent-based model of Wolbachia release in a nice tidy package

Version 0.1.0

Author person(``Sarah'', ``Belet'', email = ``sarah.belet@monash.edu'',
role = c(``aut'', ``cre''))

Maintainer The package maintainer <yourself@somewhere.net>

Description More about what it does (maybe more than one line)

Use four spaces when indenting paragraphs within the Description.

Depends R (>= 3.1.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports data.table, dplyr, ggvis, magrittr, geosphere, MCMCpack,
rbenchmark, progress, sp, truncnorm, flexsurv, ggplot2

RoxygenNote 6.1.1

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

R topics documented:

find_mate	2
hello	3
initialise_adults	3
initialise_eggs	4
initialise_juveniles	5
initialise_release	6
initialize_enzyme	7
init_juv_stage	7
init_position	8
juv_emergence	8
juv_stage_update	9
juv_to_adult	9
produce_eggs	10
random_dispersal	11
temperature_chart	11
update_enzyme	12
update_juv_stage	12

Index**14**

find_mate	<i>Finds a mate for a female mosquitoes of breeding age.</i>
-----------	--

Description

We draw a box of 'radius' k around the agent. k is an ABC parameter and usually between 11m to 100m. Note that here we mainly deal with ID number, an actual data.table variable, and not the agent's index in the main data.table.

Usage

```
find_mate(femID, k, max_daily_mates)
```

Arguments

femID	ID of female looking for mate.
k	Distance within which female mosquito 'searches' for mate. ABC parameter.
max_daily_mates	Number of times a male can mate in a day.

Value

mateID of agent, or -1 if agent does not find a mate.

Algorithm

Algorithm for finding a mate:

- If there are no males within box, return -1 for no mate.
- If there is 1 male, that is the new mate.
- If there is >1 male, randomly choose one.

Note that there is a max. number of agents that a male can mate with in a day, given by max_daily_mates. It is important in testing to ensure that the gonoCycle variable for each male mate is properly updated on a global scope. We assume that females only mate once.

Who can mate?

Recall that in order to mate, a female mosquito must have:

- enzyme between 1.2 and 1.8 inclusive
- gonoCycle 1, 2 or 3

hello	<i>Hello, World!</i>
-------	----------------------

Description

Prints 'Hello, world!'.

Usage

```
hello()
```

Examples

```
hello()
```

<code>initialise_adults</code>	<i>Creates an initial data.table of adult agents. Since this is an initial spread, some data is made up/dummy: for example, we don't know the ID of the parents. Latitude/Longitude of agents is randomly determined through another function <code>init_position</code> and added to the data.table after construction.</i>
--------------------------------	--

Description

Creates an initial data.table of adult agents. Since this is an initial spread, some data is made up/dummy: for example, we don't know the ID of the parents. Latitude/Longitude of agents is randomly determined through another function `init_position` and added to the data.table after construction.

Usage

```
initialise_adults(N, pmale)
```

Arguments

N	The number of initial adult agents.
pmale	Probability of being male.

Value

A data.table of N adult agents.

Data.table variables and initialisation

- ID: Unique ID number of agent.

gender: Male is 0, female is 1. Sampled by 1 Binomial trial as opposed to a Bernoulli trial as Bernoulli requires another package. mateID: Unique ID of their mate. Since no initial mosquitoes will have a mate yet, it is initialised as -1. Males will always have mateID as -1 since they can have multiple mates. enzyme: Enzyme Kinetic Score. See `init` for explanation. Initialised uniform randomly. age: Age in days. Initialised uniform randomly around what we would expect young adults to be. gonoCycle: Gonotrophic cycle. Means something different for males and females. males: number of times they've mated in a day, to be reset daily females: how many times they've laid a clutch of eggs we roughly estimate the gonoCycle of females based on age. timeDeath: Timestep they died: initialised as -1 as they are alive. typeDeath: Random mortality/trapped death/death due to old age: which type? whereTrapped: in the event of trapped death, where did they die? -1 otherwise. motherID: Unique ID of mother. -1 for initial adult data.table fatherID: Unique ID of father. -1 for initial adult data.table. infStatus: 1 if they carry Wolbachia, 0 if no Wolbachia, -1 for CI for initial wild type we assume they all start with 0. lat: Initial north/south or 'y' coordinate of agent. Should start with -16. long: Initial east/west of 'x' coordinate of agent. Should start with 145.

<code>initialise_eggs</code>	<i>Creates a data.table of newly laid eggs to add to the juvenile data.table.</i>
------------------------------	---

Description

Input is the indices of mothers attempting to lay eggs. Much of this code is regarding the handling of CI:

- Neither mother nor father carry Wolbachia: no Wolbachia in offspring
- Mother carries Wolbachia: offspring have nonzero probability of carrying
- Mother does not carry, father does: offspring suffer CI and won't hatch

Latitude/Longitude of agents is determined as the same position that the mother is currently at. Clutch sizes, as per literature, differ depending on Wolbachia status of mother. This is handled by the ABC parameter `eta_1` and `eta_2`.

Usage

```
initialise_eggs(toLay)
```

Arguments

`toLay` List of the indices of mothers attempting to lay a clutch.

Value

A data.table of juvenile agents in stage 1 corresponding to each mother in `toLay`.

data.table variables and initialisation

- motherID: ID of mother.
- fatherID: ID of father.
- age: Age in days. Initialised at 0 since they're new!
- stage: Development stage of clutch. Should only be 1 since these are eggs.
- infProb: Probability of carrying Wolbachia. 0: no Wolbachia, -1: Cytoplasmic Incompatability, else infProb is nonzero.
- lat: Initial north/south or 'y' coordinate of agent. Should start with -16. Should be same as mother.
- long: Initial east/west of 'x' coordinate of agent. Should start with 145. Should be same as mother.
- clutchSize: Number of juveniles in the clutch.
- pDeath: Probability of death.

initialise_juveniles	<i>Creates an initial data.table of juvenile agents. Since this is an initial spread, some data is made up/dummy: for example, we don't know the ID of the parents. Latitude/Longitude of agents is randomly determined and added to the data.table after construction.</i>
----------------------	---

Description

Creates an initial data.table of juvenile agents. Since this is an initial spread, some data is made up/dummy: for example, we don't know the ID of the parents. Latitude/Longitude of agents is randomly determined and added to the data.table after construction.

Usage

```
initialise_juveniles(Njuv)
```

Arguments

Njuv	The number of initial juvenile agents.
------	--

Value

A data.table of Njuv juvenile agents.

Data.table variables and initialisation

motherID: ID of mother. fatherID: ID of father. age: Age in days. Initialised uniform randomly around what we would expect young adults to be. stage: Development stage of clutch. 1: egg, 2: larvae, 3: pupae. infProb: Probability of carrying Wolbachia. 0: no Wolbachia, -1: Cytoplasmic Incompatability, else infProb is nonzero. lat: Initial north/south or 'y' coordinate of agent. Should start with -16. long: Initial east/west of 'x' coordinate of agent. Should start with 145. clutchSize: Number of juveniles in the clutch. enzyme: Enzyme Kinetic Score of agents. Initialised uniform randomly. FIX pDeath: Probability of death.

initialise_release	<i>Simulates a release of Wolbachia-carrying mosquitoes. Releases in this experiment are always adults. Much of this is functionally similar to initialise_adults. Since this is a release, some data is made up/dummy: for example, we don't know the ID of the parents. Latitude/Longitude of agents is given by data supplied by the WMP.</i>
--------------------	--

Description

Simulates a release of Wolbachia-carrying mosquitoes. Releases in this experiment are always adults. Much of this is functionally similar to initialise_adults. Since this is a release, some data is made up/dummy: for example, we don't know the ID of the parents. Latitude/Longitude of agents is given by data supplied by the WMP.

Usage

```
initialise_release(noReleased, pmale, lat, long)
```

Arguments

noReleased	The number of initial adult agents.
pmale	Probability of being male.
lat	Latitude of release site.
long	Longitude of release site.

Value

A data.table of N adult agents.

Data.table variables and initialisation

ID: Unique ID number of agent. gender: Male is 0, female is 1. We know proportions of female/males released from data. mateID: Unique ID of their mate. Since no initial mosquitoes will have a mate yet, it is initialised as -1. Males will always have mateID as -1 since they can have multiple mates. enzyme: Enzyme Kinetic Score. See `init` for explanation. Initialised uniform randomly. age: Age in days. Initialised uniform randomly around what we would expect young adults to be. gonoCycle: Gonotrophic cycle. Means something different for males and females. males: number of times they've mated in a day, to be reset daily females: how many times they've laid a clutch of eggs we roughly estimate the gonoCycle of females based on age. timeDeath: Timestep they died: initialised as -1 as they are alive. typeDeath: Random mortality/trapped death/death due to old age: which type? whereTrapped: in the event of trapped death, where did they die? -1 otherwise. motherID: Unique ID of mother. -1 since wild release. fatherID: Unique ID of father. -1 since wild release. infStatus: 1 if they carry Wolbachia, 0 if no Wolbachia, -1 for CI. They should mostly be 1 with a few 0 due to incomplete transmission. lat: Initial north/south or 'y' coordinate of agent. Should start with -16. long: Initial east/west of 'x' coordinate of agent. Should start with 145.

initialize_enzyme	<i>Initialises the Enzyme Kinetic model lookup table for the model run. The EKM is how we dictate the development rate of agents. The EKM model is the same one used in Focks 1993. Input data is in CELSIUS, EKM works with KELVIN.</i>
-------------------	--

Description

The model uses different sets of experimentally-derived parameters for each four development stage: Egg, Larvae, Pupae, Gonotrophic. Hence there are four equations.

Usage

```
initialize_enzyme(dailyTemps)
```

Arguments

dailyTemps Dataframe of daily average temperatures as output from function temperature_chart

Value

Chart of EKM updates for each development stage for each day.

Notation and constants

KELV_CONV: Conversion constant between Kelvin <-> Celcius. R: Universal gas constant. All other constants are experimentally-determined constants which can be found in Focks 1993. Suffixes E, L, P, G for the constants refer to each of the development stages.

"Why do you multiply by 24?"

We multiply by 24 since timesteps are in days, and EKM model is in timestep unit of hours.

init_juv_stage	<i>Roughly determines the stage of a juvenile clutch given numerical 'age'. Needs checking with literature, or replace with better method.</i>
----------------	--

Description

Roughly determines the stage of a juvenile clutch given numerical 'age'. Needs checking with literature, or replace with better method.

Usage

```
init_juv_stage(age)
```

Arguments

age 'Age' of clutch in days.

Value

Juvenile stage: 0 for egg, 1 for larval, 3 for pupal.

<code>init_position</code>	<i>Generates random 'starting' positions for adult agents. Uses boundary data for simulation region. Draws a polygon from the boundary points given in boundaryDat and picks N randomly spaced points within it. Includes a boundary check: will keep drawing points until there are N in the region/polygon. Requires package 'sp' for Polygon object.</i>
----------------------------	---

Description

Generates random 'starting' positions for adult agents. Uses boundary data for simulation region. Draws a polygon from the boundary points given in boundaryDat and picks N randomly spaced points within it. Includes a boundary check: will keep drawing points until there are N in the region/polygon. Requires package 'sp' for Polygon object.

Usage

```
init_position(boundaryDat, N)
```

Arguments

<code>boundaryDat</code>	Geographic boundary of simulation region.
<code>N</code>	Number of agents.

Value

A data frame of lat and long positions for each adult agent.

<code>juv_emergence</code>	<i>Runs the 'juvenile emergence' step. Determines which juvenile agents are to age into adult agents, and calls juv_to_adult on them. IMPORTANT: this should update mozzie.dt and juv.dt on a global scale.</i>
----------------------------	---

Description

Runs the 'juvenile emergence' step. Determines which juvenile agents are to age into adult agents, and calls juv_to_adult on them. IMPORTANT: this should update mozzie.dt and juv.dt on a global scale.

Usage

```
juv_emergence(juv.dt)
```

Arguments

<code>juv.dt</code>	The juvenile data.table.
---------------------	--------------------------

juv_stage_update	<i>Determines the juvenile agents that develop into the next stage, and updates the development stage of these juveniles. Once enzyme exceeds 0.95, a juvenile agent develops to the next stage. Recall that Stage 1: egg, 2: larvae, 3: pupae. Any pupae that are to develop into the next stage, i.e. adulthood, should have been handled by juv_emergence. IMPORTANT: this should update juv.dt on a global scale.</i>
------------------	---

Description

Determines the juvenile agents that develop into the next stage, and updates the development stage of these juveniles. Once enzyme exceeds 0.95, a juvenile agent develops to the next stage. Recall that Stage 1: egg, 2: larvae, 3: pupae. Any pupae that are to develop into the next stage, i.e. adulthood, should have been handled by juv_emergence. IMPORTANT: this should update juv.dt on a global scale.

Usage

```
juv_stage_update(juv.dt)
```

Arguments

juv.dt	The juvenile data.table.
--------	--------------------------

juv_to_adult	<i>Takes a juvenile agent and splits it into adult ages This is to simulate 'emergence' of juveniles form aquatic stage. Many variables stay the same, like motherID, but new ones are added or changed Agents also undergo one motility event when they emerge. This becomes their new lat/long.</i>
--------------	---

Description

Takes a juvenile agent and splits it into adult ages This is to simulate 'emergence' of juveniles form aquatic stage. Many variables stay the same, like motherID, but new ones are added or changed Agents also undergo one motility event when they emerge. This becomes their new lat/long.

Usage

```
juv_to_adult(juvID, idStart, pmale, lambda)
```

Arguments

juvID	The index of the agent in the juvenile data.table to be converted to adult agents.
idStart	Where to start 'counting' the ID numbers of new agents from.
pmale	Probability of being male.
lambda	Shape parameter for distance calculation. ABC parameter.

Value

A data.table of N adult agents.

Data.table variables and initialisation

ID: Unique ID number of agent. gender: Male is 0, female is 1. Sampled by 1 Binomial trial as opposed to a Bernoulli trial as Bernoulli requires another package. mateID: Unique ID of their mate. Since no new adults will have a mate yet, it is initialised as -1. Males will always have mateID as -1 since they can have multiple mates. enzyme: Enzyme Kinetic Score. See `init` for explanation. Initialised at 0 since they have just moved up from previous stage. age: Age in days. Given from their juvenile agent entry. gonoCycle: Gonotrophic cycle. Means something different for males and females. males: number of times they've mated in a day, to be reset daily females: how many times they've laid a clutch of eggs Starts at 0 for everyone since they've just emerged. timeDeath: Timestep they died: initialised as -1 as they are alive. typeDeath: Random mortality/trapped death/death due to old age: which type? whereTrapped: in the event of trapped death, where did they die? -1 otherwise. motherID: Unique ID of mother. Taken from their juvenile agent entry. fatherID: Unique ID of father. Taken from their juvenile agent entry. infStatus: 1 if they carry Wolbachia, 0 if no Wolbachia, -1 for CI There should not be any with -1 because they should not have hatched. lat: Initial north/south or 'y' coordinate of agent. Should start with 145. Determined by motility event. long: Initial east/west of 'x' coordinate of agent. Should start with -16. Determined by motility event.

produce_eggs	<p><i>Determines which mothers are ready to lay eggs and calls initialise_eggs. Conditions to be satisfied for a mother to lay a clutch of eggs:</i></p> <ol style="list-style-type: none"> 1. <i>Has a mate</i> 2. <i>Enzyme Kinetic Score > 1 and gonoCycle = 0 OR</i> 3. <i>EKS > 1.58 and gonoCycle = 1 OR</i> 4. <i>EKS > 2.16 and gonoCycle = 2.</i>
--------------	---

Description

IMPORTANT: This should update `juv.dt` on a global scale.

Usage

```
produce_eggs(mozzie.dt)
```

Arguments

`mozzie.dt` data.table of adult agents.

random_dispersal	<i>Random dispersal of a single agent according to distance formulae.</i>
------------------	---

Description

Method we use: distance and bearing, Haversine method Note we work in units of radians and kilometres.

1. Calculate 'd', the distance travelled
2. Calculate 'theta', the direction or bearing travelled
3. Convert to lat and long.

Usage

```
random_dispersal(lat, long, lambda)
```

Arguments

lat	Latitude or 'y' coordinate of agent.
long	Longitude or 'x' coordinate of agent.
lambda	Shape parameter for distance calculation. ABC parameter.

Value

Updated lat and long of agent, as a list.

temperature_chart	<i>Calculates the average daily temperature over the simulation period. Temperature calculations are in Celcius.</i>
-------------------	--

Description

Function splits the data into separate months, lays it end to end, and then calculates the mean temperature for each day of the simulation period.

Usage

```
temperature_chart(mindf, maxdf, noTimeSteps)
```

Arguments

mindf	Raw data of all minimum daily temperatures, read into a dataframe.
maxdf	Raw data of all maximum daily temperatures, read into a dataframe.
noTimeSteps	Number of timesteps in the simulation.

Details

Note that due to months not having the same number of days, we remove any -1 entries which correspond to days like Feb 30.

This function is deterministic and should yield the same result for the same input.

Value

The mean temperature for each day of the simulation period, expressed as a dataframe of length noTimeSteps.

update_enzyme	<i>Gets the amount by which the Enzyme Kinetic Score for an agent will increase. This value is added to their current EKS. This function works as a 'lookup' for the EKS data, which is all calculated at the beginning of the simulation- this function just grabs it</i>
---------------	--

Description

Gets the amount by which the Enzyme Kinetic Score for an agent will increase. This value is added to their current EKS. This function works as a 'lookup' for the EKS data, which is all calculated at the beginning of the simulation- this function just grabs it

Usage

```
update_enzyme(stage, timestep)
```

Arguments

stage	Stage of agent. 1: egg, 2: larvae, 3: pupae, 4: adult.
timestep	Current timestep.

Value

Amount by which an agent's EKS will increase for that timestep.

update_juv_stage	<i>Updates the development stage of juvenile agents.</i>
------------------	--

Description

If a juvenile agent's Enzyme Kinetic Score is greater than 0.95, they age up to the next development stage:

- Egg to larvae OR
- Larvae to pupae.

Pupae development into adult agents should have already been handled. This function runs on the entire list of juvenile agents at once.

Usage

```
update_juv_stage(stage, enzyme)
```

Arguments

stage	List of all juvenile stages from juv.dt.
enzyme	List of all juvenile agent enzyme score from juv.dt.

Value

Updated stage and enzyme of agents, as a list.

Index

find_mate, [2](#)

hello, [3](#)

init_juv_stage, [7](#)

init_position, [8](#)

initialise_adults, [3](#)

initialise_eggs, [4](#)

initialise_juveniles, [5](#)

initialise_release, [6](#)

initialize_enzyme, [7](#)

juv_emergence, [8](#)

juv_stage_update, [9](#)

juv_to_adult, [9](#)

produce_eggs, [10](#)

random_dispersal, [11](#)

temperature_chart, [11](#)

update_enzyme, [12](#)

update_juv_stage, [12](#)