Package ‘MGDrivE2’

January 13, 2021

Type Package
Title Mosquito Gene Drive Explorer 2
Version 1.1.0
Maintainer Sean L. Wu <slwood89@gmail.com>
URL https://marshalllab.github.io/MGDrivE/,
https://www.marshallab.com/
BugReports https://github.com/MarshallLab/MGDrivE/issues
Description A simulation modeling framework which significantly extends capabilities from the 'MGDrivE' simulation package via a new mathematical and computational framework based on stochastic Petri nets. For more information about 'MGDrivE', see our publication: <https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/2041-210X.13318>. Some of the notable capabilities of 'MGDrivE2' include: incorporation of human populations, epidemiological dynamics, time-varying parameters, and a continuous-time simulation framework with various sampling algorithms for both deterministic and stochastic interpretations. 'MGDrivE2' relies on the genetic inheritance structures provided in package 'MGDrivE', so we suggest installing that package initially.
License GPL-3
Encoding UTF-8
ByteCompile true
LazyData true
Depends R (>= 3.1.0)
Imports Matrix, deSolve
Suggests MGDrivE, knitr, rmarkdown, ggplot2
VignetteBuilder knitr
RoxygenNote 7.1.1
NeedsCompilation no
Author Sean L. Wu [aut, cre], Jared B. Bennett [aut], Héctor Manuel Sánchez Castellanos [ctb],
Tomás M. León [ctb],
Andrew J. Dolgert [ctb],
John M. Marshall [aut]

Repository CRAN

Date/Publication 2021-01-13 06:00:03 UTC

R topics documented:

base_aquatic_gen
base_aquatic_stage
base_erlang
base_erlang_F
base_gen
base_gen_FE
base_MQ
base_MUH
base_summarize_humans
base_sum_F
batch_migration
batch_migration_stage
calc_move_rate
equilibrium_lifecycle
equilibrium_SEI_SEIR
equilibrium_SEI_SIS
get_shape
make_Q_SEI
movement_prob2rate
mu_ts
sim_trajectory_base_CSV
sim_trajectory_base_R
sim_trajectory_CSV
sim_trajectory_R
solve_muAqua
split_aggregate_CSV
spn_hazards
spn_Post
spn_P_epiSEIR_network
spn_P_epiSEIR_node
spn_P_epiSIS_network
spn_P_epiSIS_node
spn_P_lifecycle_network
spn_P_lifecycle_node
spn_S
spn_T_epiSEIR_network
spn_T_epiSEIR_node
spn_T_epiSIS_network
base_aquatic_genotype

### Description

This function takes a given aquatic (egg, larval, pupal) stage and sums over the Erlang-distributed stages, returning summary trajectories by genotype.

### Usage

```r
base_aquatic_genotype(out, spn_P, elp)
```

### Arguments

- **out**: the output of `sim_trajectory_R`
- **spn_P**: the places of the SPN, see details
- **elp**: stage to summarize, one of: "egg", "larvae", "pupae"

### Details

This function is the base function for `summarize_eggs_genotype, summarize_larvae_genotype, and summarize_pupae_genotype`. The places (spn_P) object is generated from one of the following: `spn_P_lifecycle_node, spn_P_lifecycle_network, spn_P_epiSIS_node, spn_P_epiSIS_network, spn_P_epiSEIR_node, or spn_P_epiSEIR_network`. The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, genotype, and value.
base_aquatic_stage

Value

a 3 to 5 column dataframe for plotting with ggplot2

Description

This function takes a given aquatic (egg, larval, pupal) stage and sums over the genotypes, returning summary trajectories by Erlang-distributed stage.

Usage

base_aquatic_stage(out, spn_P, elp)

Arguments

out the output of sim_trajectory_R
spn_P the places of the SPN, see details
elp stage to summarize, one of: "egg", "larvae", "pupae"

Details

This function is the base function for summarize_eggs_stage, summarize_larvae_stage, and summarize_pupae_stage.

The places (spn_P) object is generated from one of the following: spn_P_lifecycle_node, spn_P_lifecycle_network, spn_P_epiSIS_node, spn_P_epiSIS_network, spn_P_epiSEIR_node, or spn_P_epiSEIR_network.

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, Erlang-stage, and value.

Value

a 3 to 5 column dataframe for plotting with ggplot2
base_erlang

*Base Summary of Erlang Stages for Aquatic Life Stages*

**Description**

This function takes the given aquatic stage and summarizes them by Erlang-distributed dwell times, writing output to provided folders.

**Usage**

```plaintext
base_erlang(fileVec, outList, genos, nGenos, nErlang, times, nTimes, nNodes)
```

**Arguments**

- `fileVec` Vector of files for analysis
- `outList` List of files, organized by repetition, to write output
- `genos` Genotypes to summarize by
- `nGenos` Number of genotypes
- `nErlang` Number of Erlang stages
- `times` Vector of sampling times
- `nTimes` Number of sampled times
- `nNodes` Number of nodes in the network

**Details**

This function is a base function used in `split_aggregate_CSV`.

**Value**

None

---

base_erlang_F

*Base Summary of Erlang Stages for Adult Females*

**Description**

This function takes ALL of the adult female stages and summarized them by Erlang-distributed latent infection, writing output to provided folders.

**Usage**

```plaintext
base_erlang_F(fileList, outList, nGenos, nErlang, times, nTimes, nNodes)
```
Arguments

- **fileList**: Length 3 list holding 'FS', 'FE', and 'FI' files for analysis
- **outList**: List of files, organized by repetition, to write output
- **nGenos**: Number of genotypes
- **nErlang**: Number of Erlang stages
- **times**: Vector of sampling times
- **nTimes**: Number of sampled times
- **nNodes**: Number of nodes in the network

Details

This function is a base function used in `split_aggregate_CSV`.

Value

None

---

base_gen

*Base Summary for Eggs, Larvae, Pupae, Susceptible Females, and Infectious Females*

Description

This function takes a given stage and summarizes them by genotype, writing output to provided folders.

Usage

```r
base_gen(fileVec, outList, genos, nGenos, nIDX1, times, nTimes, nNodes)
```

Arguments

- **fileVec**: Vector of files for analysis
- **outList**: List of files, organized by repetition, to write output
- **genos**: Genotypes to summarize by
- **nGenos**: Number of genotypes
- **nIDX1**: First index to expand over, nE/nL/nP for aquatic stages, 1 for the rest
- **times**: Vector of sampling times
- **nTimes**: Number of sampled times
- **nNodes**: Number of nodes in the network

Details

This function is a base function used in `split_aggregate_CSV`. 
**base_gen_FE**

**Value**

None

---

**Description**

This function takes 'E' stage females and summarizes them by genotype, writing output to provided folders.

**Usage**

```r
base_gen_FE(fileVec, outList, genos, nGenos, nIDX1, times, nTimes, nNodes)
```

**Arguments**

- `fileVec`: Vector of files for analysis
- `outList`: List of files, organized by repetition, to write output
- `genos`: Genotypes to summarize by
- `nGenos`: Number of genotypes
- `nIDX1`: First index to expand over, nE/nL/nP for aquatic stages, 1 for the rest
- `times`: Vector of sampling times
- `nTimes`: Number of sampled times
- `nNodes`: Number of nodes in the network

**Details**

This function is a base function used in `split_aggregate_CSV`.

**Value**

None
base_MQ  

Base Summary Function

Description

This function does the actual calculations for `summarize_stats_CSV`. It calculates mean and quantiles, writing output to the appropriate folder.

Usage

```r
base_MQ(
  fList, 
  oDir, 
  sName, 
  nodeNames, 
  nNodes, 
  genos, 
  nGenos, 
  times, 
  nTimes, 
  num_repss, 
  mean, 
  quantiles, 
  oDepth
)
```

Arguments

- `fList`: File list, all files for this stage, organized by repetition
- `oDir`: Output directory
- `sName`: Stage signifier
- `nodeNames`: Properly formatted vector of node names for printing
- `nNodes`: Number of nodes in the simulation
- `genos`: Vector of genotypes for the header
- `nGenos`: Number of genotypes
- `times`: Vector of sampling times
- `nTimes`: Number of sampled times
- `num_repss`: Number of repetitions from the simulation
- `mean`: Boolean, calculate mean or not
- `quantiles`: Vector of quantiles to calculate, or NULL
- `oDepth`: Max(1, number of quantiles)

Value

None
**base_MUH**

*Base Summary for Males, Unmated Females, and Humans*

**Description**

This function takes a given stage (males, unmated females, or humans) and summarizes them by genotype (infection status for humans), writing output to provided folders.

**Usage**

```r
base_MUH(fileVec, outList, genos, nGenos, nTimes, nNodes)
```

**Arguments**

- **fileVec**: Vector of files for analysis
- **outList**: List of files, organized by repetition, to write output
- **genos**: Genotypes to summarize by
- **nGenos**: Number of genotypes
- **nTimes**: Number of sampled times
- **nNodes**: Number of nodes in the network

**Details**

This function is a base function used in `split_aggregate_CSV`.

**Value**

None

---

**base_summarize_humans**

*Base Function for Human Summary*

**Description**

This function takes a given infection (‘S’, ‘E’, ‘I’, ‘R’) status and returns a summary trajectory

**Usage**

```r
base_summarize_humans(out, infState)
```

**Arguments**

- **out**: the output of `sim_trajectory_R`
Details
This function is the base function for `summarize_humans_epiSIS, summarize_humans_epiSEIR`.
The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, inf, genotype, and value.

Value

a 4 to 6 column dataframe for plotting with ggplot2

base_sum_F

| base_sum_F | Base Summary of Infection (SEI) Stages for Adult Females |

Description
This function takes ALL of the adult female stages and summarized them by Erlang-distributed latent infection, writing output to provided folders.

Usage

`base_sum_F(fileList, outList, genos, nGenos, nErlang, times, nTimes, nNodes)`

Arguments

- `fileList`  Length 3 list holding ‘FS’, ‘FE’, and ‘FI’ files for analysis
- `outList`  List of files, organized by repetition, to write output
- `genos`  Genotypes to summarize by
- `nGenos`  Number of genotypes
- `nErlang`  Number of Erlang stages
- `times`  Vector of sampling times
- `nTimes`  Number of sampled times
- `nNodes`  Number of nodes in the network

Details
This function is a base function used in `split_aggregate_CSV`.

Value

None
Sample Batch Migration Events

Description

Sample batch migration events for simulation given rates of occurrence and probability of destination for each patch. Batch migration can be simulated for the aquatic life stages (eggs, larvae, pupae), adult females, and/or adult males. To simulate batch migration, each life stage needs all 3 of its arguments specified. If any arguments are left unspecified (NULL), batch migration for that life stage will not be sampled. The output of this function should be passed to sim_trajectory_R or sim_trajectory_CSV as the argument batch. Calls the internal function batch_migration_stage.

Usage

batch_migration(
  SPN_P,
  tmax,
  ELPrates = NULL,
  ELPmove = NULL,
  ELPprob = NULL,
  Frates = NULL,
  Fmove = NULL,
  Fprob = NULL,
  Mrates = NULL,
  Mmove = NULL,
  Mprob = NULL
)

Arguments

SPN_P | places of the SPN
--- | ---
tmax | maximum time of the simulation
ELPrates | rate at which aquatic stage batch migration occurs for each node (nodes without mosquitoes should be set to NaN or NA)
ELPmove | movement matrix for destinations of aquatic stage batch migration events (diagonal will be set to zero and off-diagonal elements normalized)
ELPprob | probability for each individual to be chosen for aquatic stage batch migration events (must be same length as ELPrates)
Frates | rate at which adult female batch migration occurs for each node (nodes without mosquitoes should be set to NaN or NA)
Fmove | movement matrix for destinations of adult female batch migration events (diagonal will be set to zero and off-diagonal elements normalized)
Fprob | probability for each individual to be chosen for adult female batch migration events (must be same length as Frates)
**Mrates**  
rate at which adult male batch migration occurs for each node (nodes without mosquitoes should be set to NaN or NA)

**Mmove**  
movement matrix for destinations of adult male batch migration events (diagonal will be set to zero and off-diagonal elements normalized)

**Mprob**  
probability for each individual to be chosen for adult male batch migration events (must be same length as Mrates)

---

**batch_migration_stage**  
*Internal function to sample and set up data structure for batch migration*

**Description**  
Internal function to sample and set up data structure for batch migration

**Usage**  
`batch_migration_stage(SPN_P, rates, move, prob, stage, tmax)`

**Arguments**

- **SPN_P**  
a set of Petri net places
- **rates**  
a vector of rates for each node
- **move**  
a movement matrix (where do the batches go?)
- **prob**  
the probability vector for each individual moving in each batch
- **stage**  
the life stage (one of 'ELP', 'F', 'M')
- **tmax**  
maximum simulation time

---

**calc_move_rate**  
*Calculate Outbound Movement Rate*

**Description**  
Given $P$, the cumulative probability of moving before dying, and $\mu$, the daily mortality rate, calculate the movement rate $\gamma$ to get $P$. The equation comes from integrating the competing risks and solving for $\gamma$.

**Usage**  
`calc_move_rate(mu, P)`

**Arguments**

- **mu**  
daily mortality rate
- **P**  
cumulative probability to move before dying
Value

numeric probability of movement

Examples

# parameters, see vignette MGDrivE2: One Node Lifecycle Dynamics
theta <- list(qE = 1/4, nE = 2, qL = 1/3, nL = 3, qP = 1/6, nP = 2,
muE = 0.05, muL = 0.15, muP = 0.05, muF = 0.09, muM = 0.09,
beta = 16, nu = 1/(4/24) )

# lets say a 70% chance to move over the entire lifespan
rMoveRate <- calc_move_rate(mu = theta$muF, P = 0.70)

equilibrium_lifecycle  Calculate Equilibrium for Lifecycle Model (Logistic or Lotka-Volterra)

Description

This function calculates deterministic equilibria for the mosquito lifecycle model.

Usage

equilibrium_lifecycle(  
  params,  
  NF,  
  phi = 0.5,  
  log_dd = TRUE,  
  spn_P,  
  pop_ratio_Aq = NULL,  
  pop_ratio_F = NULL,  
  pop_ratio_M = NULL,  
  cube  
)

Arguments

params a named list of parameters (see details)
NF vector of female mosquitoes at equilibrium, for every population in the environment
phi sex ratio of mosquitoes at emergence
log_dd Boolean: TRUE implies logistic density dependence, FALSE implies Lotka-Volterra model
spn_P the set of places (P) (see details)
pop_ratio_Aq May be empty; if not, a named vector or matrix. (see details)
Details

Equilibrium can be calculated using one of two models: classic logistic dynamics or following the Lotka-Volterra competition model. This is determined by the parameter `log_dd`, and it changes elements of the return list: `K` is returned for logistic dynamics, or `gamma` is returned for Lotka-Volterra dynamics.

The places (spn_P) object is generated from one of the following: `spn_P_lifecycle_node, spn_P_lifecycle_network, spn_P_epiSIS_node, spn_P_epiSIS_network, spn_P_epiSEIR_node, or spn_P_epiSEIR_network`.

The initial population genotype ratios are set by supplying the `pop_ratio_Aq`, `pop_ratio_F`, and `pop_ratio_M` values. The default value is NULL, and the function will use the wild-type alleles provided in the `cube` object. However, one can supply several different objects to set the initial genotype ratios. All genotypes provided must exist in the `cube` (this is checked by the function). If a single, named vector is provided, then all patches will be initialized with the same ratios. If a matrix is provided, with the number of columns (and column names) giving the initial genotypes, and a row for each patch, each patch can be set to a different initial ratio. The three parameters do not need to match each other.

The `params` argument supplies all of the ecological parameters necessary to calculate equilibrium values. This is used to set the initial population distribution and during the simulation to maintain equilibrium. `params` must include the following named parameters:

- `qE`: inverse of mean duration of egg stage
- `nE`: shape parameter of Erlang-distributed egg stage
- `qL`: inverse of mean duration of larval stage
- `nL`: shape parameter of Erlang-distributed larval stage
- `qP`: inverse of mean duration of pupal stage
- `nP`: shape parameter of Erlang-distributed pupal stage
- `muE`: egg mortality
- `muL`: density-independent larvae mortality
- `muP`: pupae mortality
- `muF`: adult female mortality
- `muM`: adult male mortality
- `beta`: egg-laying rate, daily
- `nu`: mating rate of unmated females

The return list contains all of the `params` parameters, along with the density-dependent parameter, either `K` or `gamma`. These are the parameters necessary later in the simulations. This was done for compatibility with `equilibrium_SEI_SIS`, which requires several extra parameters not required further in the simulations.

For equilibrium with epidemiological parameters, see `equilibrium_SEI_SIS`. For equilibrium with latent humans (SEIR dynamics), see `equilibrium_SEI_SEIR`.
equilibrium_SEI_SEIR  

**Value**

a list with 3 elements: init a matrix of equilibrium values for every life-cycle stage, params a list of parameters for the simulation, M0 a vector of initial conditions

---

**Description**

Given prevalence of disease in humans (modeled as an SEIR: Susceptible-Latent-Infected-Recovered process with birth and death) and entomological parameters of transmission, this function calculates the quasi-stationary distribution of adult female mosquitoes across SEI (Susceptible-Exposed-Infectious) stages, allowing for Erlang distributed E stage.

**Usage**

```r
equilibrium_SEI_SEIR(
  params,
  node_list = "b",
  NF = NULL,
  phi = 0.5,
  NH = NULL,
  log_dd = TRUE,
  spn_P,
  pop_ratio_Aq = NULL,
  pop_ratio_F = NULL,
  pop_ratio_M = NULL,
  pop_ratio_H = c(1, 0, 0, 0),
  cube
)
```

**Arguments**

- **params**: a named list of parameters (see details)
- **node_list**: a character vector specifying what type of nodes to create; (m = a node with only mosquitoes, h = a node with only humans, b = a node with both humans and mosquitoes)
- **NF**: vector of female mosquitoes at equilibrium, for mosquito-only nodes
- **phi**: sex ratio of mosquitoes at emergence
- **NH**: vector of humans at equilibrium, for human-only nodes
- **log_dd**: Boolean: TRUE implies logistic density dependence, FALSE implies Lotka-Volterra model
- **spn_P**: the set of places (P) (see details)
- **pop_ratio_Aq**: May be empty; if not, a named vector or matrix. (see details)
pop_ratio_F  May be empty; if not, a named vector or matrix. (see details)
pop_ratio_M  May be empty; if not, a named vector or matrix. (see details)
pop_ratio_H  Prevalence in human-only nodes, default is all susceptible

cube  an inheritance cube from the MGDrivE package (e.g. `cubeMendelian`)

**Details**

This function handles 3 types of nodes: Human only, mosquito only, and nodes with both. These nodes are set using the `node_list` parameter. Mosquito-only node equilibrium calls `equilibrium_lifecyle`, which follows one of two models: classic logistic dynamics or the Lotka-Volterra competition model. This is determined by the parameter `log_dd`, and it changes elements of the return list: `K` is returned for logistic dynamics, or `gamma` is returned for Lotka-Volterra dynamics. This is parameterized with the `NF` parameter to define the adult female numbers. This parameter only needs to be supplied if there are mosquito-only nodes.

Human-only nodes don’t require any equilibrium calculations. These nodes use the `NH` and `pop_ratio_H` to set adult human populations and infection rates in nodes. These two parameters only need to be supplied if there are human-only nodes. `pop_ratio_H` needs to be a matrix with the number of rows equal to the number of human-only patches, and 4 columns. The columns are assumed to be fractions of the population in "S", "E", "I", or "R" states, and every row must sum to 1.

For human and mosquito nodes, this function calls `make_Q_SEI` to construct the infinitesimal generator matrix which is used to solve for the quasi-stationary (stochastic) or equilibrium (deterministic) distribution of mosquitoes over stages. Parameters are provided by `params`.

For information on the method used to solve this distribution, see section "3.1.3 Nonsingularity of the Subintensity Matrix" of:


The places (`spn_P`) object is generated from one of the following: `spn_P_lifecycle_node`, `spn_P_lifecycle_network`, `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`.

The initial population genotype ratios are set by supplying the `pop_ratio_Aq`, `pop_ratio_F`, and `pop_ratio_M` values. The default value is NULL, and the function will use the wild-type alleles provided in the `cube` object. However, one can supply several different objects to set the initial genotype ratios. All genotypes provided must exist in the cube (this is checked by the function). If a single, named vector is provided, then all patches will be initialized with the same ratios. If a matrix is provided, with the number of columns (and column names) giving the initial genotypes, and a row for each patch, each patch can be set to a different initial ratio. The three parameters do not need to match each other.

The `params` argument supplies all of the ecological and epidemiological parameters necessary to calculate equilibrium values. This is used to set the initial population distribution and during the simulation to maintain equilibrium. This `params` must include the following named parameters, noted as being the same as lifecycle parameters, or new for the epidemiological equilibrium

- **(Lifecycle parameters)**
  - `qE`: inverse of mean duration of egg stage
  - `nE`: shape parameter of Erlang-distributed egg stage
  - `qL`: inverse of mean duration of larval stage
equilibrium_SEI_SIS

- nL: shape parameter of Erlang-distributed larval stage
- qP: inverse of mean duration of pupal stage
- nP: shape parameter of Erlang-distributed pupal stage
- μE: egg mortality
- μL: density-independent larvae mortality
- μP: pupae mortality
- μF: adult female mortality
- μM: adult male mortality
- beta: egg-laying rate, daily
- nu: mating rate of unmated females

- (Epidemiological parameters)
  - NH: number of humans, can be a vector
  - X: SEIR prevalence in humans, can be a vector of length 4 for 1 node, or a matrix for many nodes
  - NFX: number of female mosquitoes, only required if any prevalence (X) is zero
  - b: mosquito to human transmission efficiency, can be a vector
  - c: human to mosquito transmission efficiency, can be a vector
  - r: rate of recovery in humans (1/duration of infectiousness)
  - μH: death rate of humans (1/avg lifespan)
  - f: rate of blood feeding
  - Q: human blood index
  - qEIP: related to scale parameter of Gamma distributed EIP (1/qEIP is mean length of EIP)
  - nEIP: shape parameter of Gamma distributed EIP
  - delta: inverse duration of the latent stage (E)

The return list contains all of the parameters necessary later in the simulations.
For equilibrium without epidemiological parameters, see equilibrium_lifeycle. For equilibrium without latent humans (SIS dynamics), see equilibrium_SEI_SIS.

Value

- a vector of the equilibrium number of females in each SEI stage

---

equilibrium_SEI_SIS Calculate Equilibrium for Mosquito SEI - Human SIS Model

Description

Given prevalence of disease in humans (modeled as an SIS: Susceptible-Infected-Susceptible process with birth and death) and entomological parameters of transmission, this function calculates the quasi-stationary distribution of adult female mosquitoes across SEI (Susceptible-Exposed-Infectious) stages, allowing for Erlang distributed E stage.
Usage

```r
equilibrium_SEI_SIS(
  params,
  node_list = "b",
  NF = NULL,
  phi = 0.5,
  NH = NULL,
  log_dd = TRUE,
  spn_P,
  pop_ratio_Aq = NULL,
  pop_ratio_F = NULL,
  pop_ratio_M = NULL,
  pop_ratio_H = 1,
  cube
)
```

Arguments

- `params`: a named list of parameters (see details)
- `node_list`: a character vector specifying what type of nodes to create; (m = a node with only mosquitoes, h = a node with only humans, b = a node with both humans and mosquitoes)
- `NF`: vector of female mosquitoes at equilibrium, for mosquito-only nodes
- `phi`: sex ratio of mosquitoes at emergence
- `NH`: vector of humans at equilibrium, for human-only nodes
- `log_dd`: Boolean: TRUE implies logistic density dependence, FALSE implies Lotka-Volterra model
- `spn_P`: the set of places (P) (see details)
- `pop_ratio_Aq`: May be empty; if not, a named vector or matrix. (see details)
- `pop_ratio_F`: May be empty; if not, a named vector or matrix. (see details)
- `pop_ratio_M`: May be empty; if not, a named vector or matrix. (see details)
- `pop_ratio_H`: Prevalence in human-only nodes
- `cube`: an inheritance cube from the MGDrivE package (e.g. `cubeMendelian`)

Details

This function handles 3 types of nodes: Human only, mosquito only, and nodes with both. These nodes are set using the `node_list` parameter. Mosquito-only node equilibrium calls `equilibrium_lifecyle`, which follows one of two models: classic logistic dynamics or the Lotka-Volterra competition model. This is determined by the parameter `log_dd`, and it changes elements of the return list: `K` is returned for logistic dynamics, or `gamma` is returned for Lotka-Volterra dynamics. This is parameterized with the `NF` parameter to define the adult female numbers. This parameter only needs to be supplied if there are mosquito-only nodes.

Human-only nodes don’t require any equilibrium calculations. These nodes use the `NH` and `pop_ratio_H` to set adult human populations and infection rates in nodes. These two parameters only need to be supplied if there are human-only nodes.
For human and mosquito nodes, this function calls `make_Q_SEI` to construct the infinitesimal generator matrix which is used to solve for the quasi-stationary (stochastic) or equilibrium (deterministic) distribution of mosquitoes over stages. Parameters are provided by `params`.

For information on the method used to solve this distribution, see section "3.1.3 Nonsingularity of the Subintensity Matrix" of:


The places (`spn_P`) object is generated from one of the following: `spn_P_lifecycle_node`, `spn_P_lifecycle_network`, `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`.

The initial population genotype ratios are set by supplying the `pop_ratio_Aq`, `pop_ratio_F`, and `pop_ratio_M` values. The default value is NULL, and the function will use the wild-type alleles provided in the `cube` object. However, one can supply several different objects to set the initial genotype ratios. All genotypes provided must exist in the `cube` object. If a single, named vector is provided, then all patches will be initialized with the same ratios. If a matrix is provided, with the number of columns (and column names) giving the initial genotypes, and a row for each patch, each patch can be set to a different initial ratio. The three parameters do not need to match each other.

The `params` argument supplies all of the ecological and epidemiological parameters necessary to calculate equilibrium values. This is used to set the initial population distribution and during the simulation to maintain equilibrium. This `params` must include the following named parameters, noted as being the same as lifecycle parameters, or new for the epidemiological equilibrium

- **(Lifecycle parameters)**
  - $q_E$: inverse of mean duration of egg stage
  - $n_E$: shape parameter of Erlang-distributed egg stage
  - $q_L$: inverse of mean duration of larval stage
  - $n_L$: shape parameter of Erlang-distributed larval stage
  - $q_P$: inverse of mean duration of pupal stage
  - $n_P$: shape parameter of Erlang-distributed pupal stage
  - $\mu_E$: egg mortality
  - $\mu_L$: density-independent larvae mortality
  - $\mu_P$: pupae mortality
  - $\mu_F$: adult female mortality
  - $\mu_M$: adult male mortality
  - $\beta$: egg-laying rate, daily
  - $\nu$: mating rate of unmated females

- **(Epidemiological parameters)**
  - $N_H$: number of humans, can be a vector
  - $X$: prevalence in humans, can be a vector
  - $N_{FX}$: number of female mosquitoes, only required if any prevalence ($X$) is zero
  - $b$: mosquito to human transmission efficiency, can be a vector
  - $c$: human to mosquito transmission efficiency, can be a vector
  - $r$: rate of recovery in humans (1/duration of infectiousness)
get_shape

- \( \mu_H \): death rate of humans (1/avg lifespan)
- \( f \): rate of blood feeding
- \( Q \): human blood index
- \( q_{EIP} \): related to scale parameter of Gamma distributed EIP (1/qEIP is mean length of EIP)
- \( n_{EIP} \): shape parameter of Gamma distributed EIP

The return list contains all of the parameters necessary later in the simulations.

For equilibrium without epidemiological parameters, see `equilibrium_lifecycle`. For equilibrium with latent humans (SEIR dynamics), see `equilibrium_SEI_SEIR`.

For examples of using this function, see: vignette("lifecycle-node",package = "MGDrivE2")

**Value**

A vector of the equilibrium number of females in each SEI stage

---

**get_shape**

*Calculate Erlang shape parameter*

**Description**

Calculate Erlang shape parameter

**Usage**

`get_shape(cv, q)`

**Arguments**

- `cv` coefficient of variation (CV) between mean and standard deviation of dwell times, smaller values of CV correspond to distributions less dispersed around their mean and larger value to more dispersed distributions.
- `q` inverse of mean dwell time
**make_Q_SEI**

**Rate Matrix (Q) for Adult Mosquito SEI Dynamics**

**Description**

Construct the infinitesimal generator matrix for (individual) adult female infection dynamics. Adult females follow SEI (Susceptible-Exposed-Infectious) style dynamics with a Gamma distributed EIP, with a mean duration 1/q and variance 1/nq^2 (following shape-scale parameterization, EIP \( \sim \Gamma(n,1/nq) \)). This function only constructs the rate matrix for either a single mosquito or cohort that all emerged at the same time (the rate matrix for a population with emergence is infinite in dimension).

**Usage**

\[
\text{make\_Q\_SEI}(q, n, \mu, c, a, x)
\]

**Arguments**

- **q**: related to scale parameter of Gamma distributed EIP (1/q is mean length of EIP)
- **n**: shape parameter of Gamma distributed EIP
- **\mu**: mosquito mortality rate
- **c**: human to mosquito transmission efficiency
- **a**: human biting rate
- **x**: prevalence of disease in humans

**Value**

rate matrix for a single (emergence) cohort of SEI mosquito

---

**movement_prob2rate**

**Convert Stochastic Matrix to Rate Matrix**

**Description**

Given a stochastic matrix, return the rate matrix (infinitesimal generator) that would generate it when exponentiated over the interval of unit time.

**Usage**

\[
\text{movement\_prob2rate}(\tau)
\]

**Arguments**

- **\tau**: a row normalized stochastic matrix
Details

Warning: if the matrix provided has diagonal-only rows (i.e., the location is independent), the rate matrix will return 0 in that row, as there is no movement rate that can generate that scenario.

Value

a list with two elements: gamma negative diagonal of the rate matrix, mat matrix of row normalized off-diagonal elements

Examples

# generate random matrix for example
# This represents a 3-node landscape, with random movement between nodes
moveMat <- matrix(data = runif(n = 9), nrow = 3, ncol = 3)
moveMat <- moveMat/rowSums(moveMat)
moveRate <- movement_prob2rate(tau = moveMat)

---

mu_ts Mosquito Death Rates, Comoros Islands

Description

This is a matrix containing estimated mosquito death rates from the Comoros Islands, between Mozambique and Madagascar. It provides hourly death rates over the course of one year.

Usage

data(mu_ts)

Format

matrix with 3 named columns and 8760 rows:

Grande_Comore Hourly death rates for main island
Moheli Hourly death rates for second island
Anjouan Hourly death rates for smallest island
sim_trajectory_base_CSV

Simulate Trajectory From one SPN Model

Description

This is an internal function to `sim_trajectory_CSV`. It does the actual sampling once all of the functions have been checked and setup.

Usage

```r
sim_trajectory_base_CSV(
  x0,  # the initial marking of the SPN (initial state)
  times,  # sequence of sampling times
  stepFun,  # a sampling function
  folders,  # vector of folders to write output
  stage,  # vector of life-stages to print
  events0 = NULL,  # a data.frame of events (uses the same format as required in package deSolve for consistency, see events for more information)
  batch = NULL,  # a list of batch migration events, created from batch_migration, may be set to NULL if not used
  Sout = NULL,  # an optional matrix to track event firings
  verbose = TRUE  # print a progress bar?
)
```

Arguments

- `x0`: the initial marking of the SPN (initial state)
- `times`: sequence of sampling times
- `stepFun`: a sampling function
- `folders`: vector of folders to write output
- `stage`: vector of life-stages to print
- `events0`: a data.frame of events (uses the same format as required in package deSolve for consistency, see `events` for more information)
- `batch`: a list of batch migration events, created from `batch_migration`, may be set to `NULL` if not used
- `Sout`: an optional matrix to track event firings
- `verbose`: print a progress bar?

Value

- no return, prints `.csv` files into provided folders
Simulate Trajectory From one SPN Model

Description

This is an internal function to sim_trajectory_R. It does the actual sampling once all of the functions have been checked and setup.

Usage

sim_trajectory_base_R(
  x0, times, num_reps, stepFun, events = NULL, batch = NULL, Sout = NULL, verbose = TRUE
)

Arguments

x0          the initial marking of the SPN (initial state)
times       sequence of sampling times
num_reps    number of repetitions to run
stepFun     a sampling function
events      a data.frame of events (uses the same format as required in package deSolve for consistency, see events for more information)
batch       a list of batch migration events, created from batch_migration, may be set to NULL if not used
Sout        an optional matrix to track event firings
verbose     print a progress bar?

Value

matrix of sampled values
Simulate Trajectory From a SPN Model

**Description**

This function provides a unified interface to the various simulation algorithms for SPN, returning output sampled at a lattice of time points to the user, and handling various exogenous events that may occur during the simulation (such as release of adult mosquitoes).

**Usage**

```r
sim_trajectory_CSV(
  x0,  
  tmax,  
  dt = 1,  
  dt_stoch = 0.1,  
  folders = "/./*",  
  stage = c("M", "F"),  
  S,  
  hazards,  
  Sout = NULL,  
  sampler = "tau",  
  method = "lsoda",  
  events = NULL,  
  batch = NULL,  
  verbose = TRUE,  
  ...
)
```

**Arguments**

- `x0`: the initial marking of the SPN (initial state, M0)
- `tmax`: the final time to end simulation
- `dt`: the time-step at which to return output (not the time-step of the sampling algorithm)
- `dt_stoch`: time-step used for approximation of hazards
- `folders`: vector of folders to write output
- `S`: a stoichiometry Matrix-class object
- `hazards`: list of hazard functions
- `Sout`: an optional matrix to track event firings
- `sampler`: determines sampling algorithm, one of; "ode", "tau", "cle", or "dm"
- `method`: if sampler is "ode", the solver to use, from deSolve
- `events`: a data.frame of events
**Details**

dt`_stoch` is used by the Poisson Time-Step (`step_PTS`) and Chemical Langevin (`step_CLE`) methods to approximate the hazards. A smaller `dt_stoch` provides a better approximation, but will take longer to run.

The stoichiometry matrix (S) is generated in `spn_S`.

The list of hazards (hazards) come from `spn_hazards`.

Several samplers are provided. The default is a Poisson Time-Step (`step_PTS`) method. Other options are Gillespie’s Direct Method (`step_DM`) and a Chemical Langevin sampler (`step_CLE`). Additionally, for convenience, an ODE "sampler" (`step_ODE`) is provided for compatibility with other samplers. This function uses methods from deSolve.

If using the ode sampler, several methods are provided in the deSolve package, see `ode`. For inhomogeneous systems, consider using the "rk4" method to avoid excessive integration times.

Additionally, events objects must follow the format required by deSolve. This was done for consistency, see `events` for more information.

This function writes all output to .csv files. Each simulation is written to a folder element - the number of repetitions is the number of folders provided. What life-stages get recorded is specified by the `stage` parameter. All life-stages can be stored, or any subset thereof. Females are split by infection status, i.e. by "S", "E", or "I".

This function tracks state variables specified by argument `stage` by default; an optional argument Sout can be provided to track number of event firings each time step (for discrete stochastic simulations), or cumulative intensity (for continuous stochastic simulations), or the rate function of particular events for ODE simulation. The matrix must have number of columns equal to number of events in the system (the number of hazard functions), and a row for each tracking variable. If Sout is provided, it output an additional csv, "events.csv". The function `track_hinf` is provided, which builds a matrix to track human infection events.

To return simulations to R for further processing, see `sim_trajectory_R`.

**Value**

NULL - prints output to .csv files
Usage

```r
sim_trajectory_R(
  x0,  
  tmax,  
  dt = 1,  
  dt_stoch = 0.1,  
  num_reps = 1,  
  S,  
  hazards,  
  Sout = NULL,  
  sampler = "tau",  
  method = "lsoda",  
  events = NULL,  
  batch = NULL,  
  verbose = TRUE,  
  ...  
)
```

Arguments

- **x0**: the initial marking of the SPN (initial state, M0)
- **tmax**: the final time to end simulation (all simulations start at 0)
- **dt**: the time-step at which to return output (**not** the time-step of the sampling algorithm)
- **dt_stoch**: time-step used for approximation of hazards
- **num_reps**: number of repetitions to run, default is 1.
- **S**: a stoichiometry `Matrix-class` object
- **hazards**: list of hazard functions
- **Sout**: an optional matrix to track event firings
- **sampler**: determines sampling algorithm, one of: "ode", "tau", "cle", or "dm"
- **method**: if sampler is "ode", the solver to use, from deSolve
- **events**: a data.frame of events, may be set to NULL if not used
- **batch**: a list of batch migration events, created from `batch_migration`, may be set to NULL if not used
- **verbose**: print a progress bar?
- **...**: further named arguments passed to the step function

Details

dt_stoch is used by the Poisson Time-Step (`step_PTS`) and Chemical Langevin (`step_CLE`) methods to approximate the hazards. A smaller `dt_stoch` provides a better approximation, but will take longer to run.

The stoichiometry matrix (S) is generated in `spn_S`.

The list of hazards (hazards) come from `spn_hazards`. 
Several samplers are provided. The default is a Poisson Time-Step (stepPTS) method. Other options are Gillespie's Direct Method (stepDM) and a Chemical Langevin sampler (stepCLE). Additionally, for convenience, an ODE "sampler" (stepODE) is provided for compatibility with other samplers. This function uses methods from deSolve.

If using the ode sampler, several methods are provided in the deSolve package, see ode. For inhomogeneous systems, consider using the "rk4" method to avoid excessive integration times.

Additionally, events objects must follow the format required by deSolve. This was done for consistency, see events for more information.

This function tracks state variables by default; an optional argument Sout can be provided to track number of event firings each time step (for discrete stochastic simulations), or cumulative intensity (for continuous stochastic simulations), or the rate function of particular events for ODE simulation. The matrix must have number of columns equal to number of events in the system (the number of hazard functions), and a row for each tracking variable. The function track_hinf is provided, which builds a matrix to track human infection events.

To save output as .csv files, see sim_trajectory_CSV.

Value

a list with 2 elements: "state" is the array of returned state values, and "events" will return events tracked with Sout if provided, otherwise is NULL

---

solve_muAqua  Solve for Constant Aquatic Mortality

Description

In MGDrivE, the model was typically solved at equilibrium assuming the density-independent mortality was constant over aquatic stages (eggs, larvae, pupae), given a daily growth rate, \( r_M \). Given that growth rate, it solved for that mortality \( \mu_{Aqua} \) by relating it with \( R_M \), the per-generation growth rate of the population, calculable from \( r_M \) and the mean duration of life stages. This function uses uniroot to solve for \( \mu_{Aqua} \).

Usage

solve_muAqua(params, rm)

Arguments

params a named list of parameters
rm the daily growth rate
split_aggregate_CSV

Details

This function needs the following parameters in `params`:

- `muF`: adult female mortality
- `beta`: rate of egg laying
- `phi`: sex ratio at emergence
- `qE`: inverse of mean duration of egg stage
- `nE`: shape parameter of Erlang-distributed egg stage
- `qL`: inverse of mean duration of larval stage
- `nL`: shape parameter of Erlang-distributed larval stage
- `qP`: inverse of mean duration of pupal stage
- `nP`: shape parameter of Erlang-distributed pupal stage

Value

location of the root, as provided from uniroot

Examples

```r
theta <- list(qE = 1/4, nE = 2, qL = 1/5, nL = 3, qP = 1/6, nP = 2, muF = 1/12, beta = 32, phi = 0.5);
muAqatic <- solve_muAqua(params = theta, rm = 1.096)
```

Description

This function reads in the output files from `sim_trajectory_CSV` and splits them into smaller files. The files are output by patch, with the appropriate patch numbers for mosquitoes or humans, and specific stages are aggregated by a given metric.

Usage

```r
split_aggregate_CSV(  
  read_dir,  
  write_dir = read_dir,  
  spn_P,  
  tmax,  
  dt,  
  erlang = FALSE,  
  sum_fem = FALSE,  
  rem_file = FALSE,  
  verbose = TRUE)
```
**Arguments**

- **read_dir**: Directory where output was written to
- **write_dir**: Directory to write output to. Default is read_dir
- **stage**: Life stage to print, see details
- **spn_P**: Places object, see details
- **tmax**: The final time to end simulation
- **dt**: The time-step at which to return output (not the time-step of the sampling algorithm)
- **erlang**: Boolean, default is FALSE, to return summaries by genotype
- **sum_fem**: if TRUE, in addition to FS, FE, FI output by node and repetition, output an additional file F which sums over infection states (S,E,I). Does nothing if the simulation did not include epi dynamics.
- **rem_file**: Remove original output? Default is FALSE
- **verbose**: Chatty? Default is TRUE

**Details**

Given the **read_dir**, this function assumes the follow file structure:

- **read_dir**
  - repetition 1
    * M.csv
    * FS.csv
    * ...
  - repetition 2
    * M.csv
    * FS.csv
    * ...
  - repetition 3
  - ...

This function expects the **write_dir** to be empty, and it sets up the same file structure as the **read_dir**. For a 2-node simulation, the output will be organized similar to:

- **write_dir**
  - repetition 1
    * M_0001.csv
    * M_0002.csv
    * FS_0001.csv
stage defines which life-stages the function will analyze. These stages must be any combination of: "E", "L", "P", "M", "U", "FS", "FE", "FI", "H". These must come from the set of stages provided to `sim_trajectory_CSV` via the stage argument. It can be less than what was printed by the simulation, but any extra stages provided, but not printed, will throw a warning and then be ignored.

erlang defines how aquatic (eggs, larvae, and pupae) stages and adult females (only mated females) are aggregated. By default, erlang is FALSE, and all of these stages are summarized by genotype only, combining any Erlang-distributed dwell stages (for eggs, larvae, and pupae) or latent infection (for adult females) stages. If erlang is TRUE, summaries are returned by dwell stage or infection status, combining any genotype information.

Female summaries always combine over mate-genotype, so only female genotypes are returned.

The places (`spn_P`) object is generated from one of the following: `spn_P_lifecycle_node`, `spn_P_lifecycle_network`, `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`.

tmax, dt define the last sampling time, and each sampling time in-between.

For more details about using this function to process CSV output see: vignette("data-analysis", package = "MGDrivE2")

**Value**

Writes output to files in write_dir
Usage

```r
spn_hazards(
  spn_P,
  spn_T,
  cube,
  params,
  type = "life",
  log_dd = TRUE,
  exact = TRUE,
  tol = 1e-12,
  verbose = TRUE
)
```

Arguments

- `spn_P` the set of places (P) (see details)
- `spn_T` the set of transitions (T) (see details)
- `cube` an inheritance cube from the MGDrivE package (e.g. `cubeMendelian`)
- `params` a named list of parameters (see details)
- `type` string indicating type of hazards, one of: "life", "SIS", or "SEIR"
- `log_dd` if TRUE, use logistic (carrying capacity) density dependent hazards, if FALSE use Lotka-Volterra density dependent hazards for larval mortality
- `exact` boolean, make exact (integer input) hazards? Default is TRUE
- `tol` if exact=FALSE, the value of hazard below which it is clipped to 0
- `verbose` display a progress bar when making hazards?

Details

If these hazards will be used in a continuous approximation algorithm, such as an ODE method (`step_ODE`) or Gillespie's Direct Method (`step_DM`), it is recommended to use exact=FALSE. If the hazards will be used in an integer state space method, such as tau-leaping (`step_PTS`) or Chemical Langevin (`step_CLE`) methods, it is recommended to use exact=TRUE.

The places (spn_P) object is generated from one of the following: `spn_P_lifecycle_node`, `spn_P_lifecycle_network`, `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`.

The set of transitions (spn_T) is generated from one of the following: `spn_T_lifecycle_node`, `spn_T_lifecycle_network`, `spn_T_epiSIS_node`, `spn_T_epiSIS_network`, `spn_T_epiSEIR_node`, or `spn_T_epiSEIR_network`.

The `params` object is generated from either `equilibrium_lifecycle` or `equilibrium_SEI_SIS`; it is the "params" object in the return list. The equilibrium function used must match the `type` parameter.

The `type` parameter indicates what type of simulation is being run. It is one of: "life", "SIS", or "SEIR". This must match the `params` object supplied.

Use of this function is demonstrated in many vignettes, `browseVignettes(package = "MGDrivE2")`
spn_Post

Make Post Matrix For a Petri Net

Description
Generate the Post (|v| by |u|) matrix for the SPN. This gives the edges from T to P (output arcs) in the bipartite network.

Usage
spn_Post(spn_P, spn_T)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spn_P</td>
<td>set of places (P) (see details)</td>
</tr>
<tr>
<td>spn_T</td>
<td>set of transitions (T) (see details)</td>
</tr>
</tbody>
</table>

Details
The places (spn_P) object is generated from one of the following: spn_P_lifecycle_node, spn_P_lifecycle_network, spn_P_epiSIS_node, spn_P_epiSIS_network, spn_P_epiSEIR_node, or spn_P_epiSEIR_network.

The set of transitions (spn_T) is generated from one of the following: spn_T_lifecycle_node, spn_T_lifecycle_network, spn_T_epiSIS_node, spn_T_epiSIS_network, spn_T_epiSEIR_node, spn_T_epiSEIR_network.

Value
a matrix of type dgCMatrix-class

spn_Pre

Make Pre Matrix For a Petri Net

Description
Generate the Pre (|v| by |u|) matrix for the SPN. This gives the edges from P to T (input arcs) in the bipartite network.

Usage
spn_Pre(spn_P, spn_T)
Arguments

- **spn_P** set of places (P) (see details)
- **spn_T** set of transitions (T) (see details)

Details

The places (spn_P) object is generated from one of the following: `spn_P_lifecycle_node`, `spn_P_lifecycle_network`, `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`. The set of transitions (spn_T) is generated from one of the following: `spn_T_lifecycle_node`, `spn_T_lifecycle_network`, `spn_T_epiSIS_node`, `spn_T_epiSIS_network`, `spn_T_epiSEIR_node`, or `spn_T_epiSEIR_network`.

Value

A matrix of type `dgCMatrix-class`

---

**spn_P_epiSEIR_network**  
*Make Places (P) For a Network (SEI Mosquitoes - SEIR Humans)*

Description

This function makes the set of places (P) for a SPN model of a metapopulation network for simulation of coupled SEI-SEIR dynamics. It is the network version of `spn_P_epiSEIR_node`.

Usage

```r
spn_P_epiSEIR_network(node_list, params, cube)
```

Arguments

- **node_list** a character vector specifying what type of nodes to create; (m = a node with only mosquitoes, h = a node with only humans, b = a node with both humans and mosquitoes)
- **params** a named list of parameters (see details)
- **cube** an inheritance cube from the MGDrivE package (e.g. `cubeMendelian`)

Details

The params argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the `nE`, `nL`, `nP`, and `nEIP` parameters to be specified. For more details, see `equilibrium_SEI_SEIR`

For examples of using this function, see: vignette("seir-dynamics",package = "MGDrivE2")

Value

A list with two elements: `ix` contains labeled indices of the places by life stage and node, `u` is the character vector of places (P)
Make Places (P) For a Node (SEI Mosquitoes - SEIR Humans)

Description

This function makes the set of places (P) for a SPN. It is used alone if our model is a single-node metapopulation for mosquito SEI and human SEIR dynamics; otherwise it is used as part of other functions to make SPN models with larger state spaces (metapopulation models, \texttt{spn\_P\_epiSEIR\_network}).

Usage

\texttt{spn\_P\_epiSEIR\_node}(params, cube)

Arguments

- \texttt{params} 
  a named list of parameters (see details)
- \texttt{cube} 
  an inheritance cube from the MGDrivE package (e.g. \texttt{cubeMendelian})

Details

The \texttt{params} argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the \texttt{nE}, \texttt{nL}, \texttt{nP}, and \texttt{nEIP} parameters to be specified. For more details, see \texttt{equilibrium\_SEI\_SEIR}

For examples of using this function, see: \texttt{vignette("seir-dynamics",package = "MGDrivE2")}

Value

a list with two elements: \texttt{ix} contains labeled indices of the places by life stage, \texttt{u} is the character vector of places (P)

Make Places (P) For a Network (SEI Mosquitoes - SIS Humans)

Description

This function makes the set of places (P) for a SPN model of a metapopulation network for simulation of coupled SEI-SIS dynamics. It is the network version of \texttt{spn\_P\_epiSIS\_node}.

Usage

\texttt{spn\_P\_epiSIS\_network}(node\_list, params, cube)
spn_P_epiSIS_node

Arguments

node_list a character vector specifying what type of nodes to create; (m = a node_id with only mosquitoes, h = a node_id with only humans, b = a node_id with both humans and mosquitoes)

params a named list of parameters (see details)

cube an inheritance cube from the MGDrivE package (e.g. cubeMendelian)

Details

The params argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the nE, nL, nP, and nEIP parameters to be specified. For more details, see equilibrium_SEI_SIS

For examples of using this function, see: vignette("epi-network",package = "MGDrivE2")

Value

a list with two elements: ix contains labeled indices of the places by life stage and node_id, u is the character vector of places (P)

Description

This function makes the set of places (P) for a SPN. It is used alone if our model is a single-node metapopulation for mosquito SEI and human SIS dynamics; otherwise it is used as part of other functions to make SPN models with larger state spaces (metapopulation models, see spn_P_epiSIS_network).

Usage

spn_P_epiSIS_node(params, cube)

Arguments

params a named list of parameters (see details)

cube an inheritance cube from the MGDrivE package (e.g. cubeMendelian)

Details

The params argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the nE, nL, nP, and nEIP parameters to be specified. For more details, see equilibrium_SEI_SIS

For examples of using this function, see: vignette("epi-node",package = "MGDrivE2")
spn_P_lifecycle_network

Value

a list with two elements: \( ix \) contains labeled indices of the places by life stage, \( u \) is the character vector of places (P)

Description

This function makes the set of places (P) for a SPN model of a metapopulation network. It is the network version of \( \text{spn}_P\_\text{lifecycle}_\text{node} \).

Usage

\[
\text{spn}_P\_\text{lifecycle}_\text{network}(\text{num}\_\text{nodes}, \text{params}, \text{cube})
\]

Arguments

- \( \text{num}\_\text{nodes} \): number of nodes in the network
- \( \text{params} \): a named list of parameters (see details)
- \( \text{cube} \): an inheritance cube from the MGDrivE package (e.g. \( \text{cubeMendelian} \))

Details

The \( \text{params} \) argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the \( nE, nL, \) and \( nP \) parameters to be specified. For more details, see \( \text{equilibrium}\_\text{lifecycle} \)

For examples of using this function, see: \( \text{vignette(}\text{“lifecycle-network”),package = “MGDrivE2”} \)

Value

a list with two elements: \( ix \) contains labeled indices of the places by life stage and node_id, \( u \) is the character vector of places (P)
**spn_P_lifecycle_node**  
*Make Places (P) For a Node (Mosquitoes only)*

**Description**

This function makes the set of places (P) for a SPN. It is used alone if our model is a single-node metapopulation for mosquito dynamics only; otherwise it is used as part of other functions to make SPN models with larger state spaces (metapopulation models, see `spn_P_lifecycle_network`).

**Usage**

```r
spn_P_lifecycle_node(params, cube)
```

**Arguments**

- `params` a named list of parameters (see details)
- `cube` an inheritance cube from the `MGDrivE` package (e.g. `cubeMendelian`)

**Details**

The `params` argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the `nP`, `nL`, and `nE` parameters to be specified. For more details, see `equilibrium_lifecycle`.

For examples of using this function, see: `vignette("lifecycle-node",package="MGDrivE2")`

**Value**

A list with two elements: `ix` contains labeled indices of the places by life stage, `u` is the character vector of places (P).

**spn_S**  
*Make stoichiometry Matrix For a Petri Net*

**Description**

Generate the stoichiometry (|u| by |v|) matrix for the SPN. Each column gives the net effect of that transition firing upon the state space of the model. Internally, this creates a `Pre` (`spn_Pre`) and Post (`spn_Post`) matrix, and then calculates the final stoichiometry.

**Usage**

```r
spn_S(spn_P, spn_T)
```
**spn_T_epiSEIR_network**

**Arguments**

- spn_P: set of places (P) (see details)
- spn_T: set of transitions (T) (see details)

**Details**

The places (spn_P) object is generated from one of the following: `spn_P_lifecycle_node`, `spn_P_lifecycle_network`, `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`.

The set of transitions (spn_T) is generated from one of the following: `spn_T_lifecycle_node`, `spn_T_lifecycle_network`, `spn_T_epiSIS_node`, `spn_T_epiSIS_network`, `spn_T_epiSEIR_node`, or `spn_T_epiSEIR_network`.

---

**spn_T_epiSEIR_network**  
*Make Transitions (T) For a Network (SEI Mosquitoes - SEIR Humans)*

**Description**

This function makes the set of transitions (T) for a SPN model of a metapopulation network for simulation of coupled SEI-SEIR dynamics. It is the network version of `spn_T_epiSEIR_node`.

**Usage**

```r
spn_T_epiSEIR_network(node_list, spn_P, params, cube, h_move, m_move)
```

**Arguments**

- node_list: a character vector specifying what type of nodes to create; (m = a node with only mosquitoes, h = a node with only humans, b = a node with both humans and mosquitoes)
- spn_P: set of places produced by `spn_P_epiSEIR_network`
- params: a named list of parameters (see details)
- cube: an inheritance cube from the MGDrivE package (e.g. `cubeMendelian`)
- h_move: binary adjacency matrix indicating if movement of humans between nodes is possible or not
- m_move: binary adjacency matrix indicating if movement of mosquitoes between nodes is possible or not

**Details**

This function takes the places produced from `spn_P_epiSEIR_network` and builds all possible transitions between subsets of those places.

The params argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the nE, nL, nP, and nEIP parameters to be specified. For more details, see `equilibrium_SEI_SEIR`
While this function produces all structural information related to transitions, hazards are produced by a separate function, `spn_hazards`.

For larger networks, this function may take some time to return, please be patient; the Petri Net modeling formalism trades additional computation time at model initialization for faster sampling of trajectories within a simulation.

Please note, the movement matrices (h_move and m_move) are NOT stochastic matrices, just binary matrices that say if i,j can exchange population. Diagonal elements must be FALSE, and both matrices are checked for validity; the function will stop with errors if the adjacency matrix specifies illegal movement rules (e.g.; mosquito movement from a "h" node to a "b" node).

For examples of using this function, see: vignette("seir-dynamics",package = "MGDrivE2")

Value

a list with two elements: T contains transitions packets as lists, v is the character vector of transitions (T)

```
spn_T_epiSEIR_node
```

Make Transitions (T) For a Node (SEI Mosquitoes - SEIR Humans)

Description

This function makes the set of transitions (T) for a SPN. It is used alone if our model is a single-node metapopulation of mosquito and human dynamics; otherwise it is used as part of other functions to make SPN models with larger state spaces (metapopulation models, see `spn_T_epiSEIR_network`).

Usage

```
spn_T_epiSEIR_node(spn_P, params, cube)
```

Arguments

- `spn_P` set of places produced by `spn_P_epiSEIR_node`
- `params` a named list of parameters (see details)
- `cube` an inheritance cube from the MGDrivE package (e.g. `cubeMendelian`)

Details

This function takes the places produced from `spn_P_epiSEIR_node` and builds all possible transitions between subsets of those places.

The `params` argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the nE, nL, nP, and nEIP parameters to be specified. For more details, see `equilibrium_SEI_SEIR`

While this function produces all structural information related to transitions, hazards are produced by a separate function, `spn_hazards`.

For examples of using this function, see: vignette("seir-dynamics",package = "MGDrivE2")
Value

a list with two elements: \( T \) contains transitions packets as lists, \( v \) is the character vector of transitions (T)

---

**spn_T_epiSIS_network**  *Make Transitions (T) For a Network (SEI Mosquitoes - SIS Humans)*

**Description**

This function makes the set of transitions (T) for a SPN model of a metapopulation network for simulation of coupled SEI-SIS dynamics. It is the network version of `spn_T_epiSIS_node`.

**Usage**

`spn_T_epiSIS_network(node_list, spn_P, params, cube, h_move, m_move)`

**Arguments**

- **node_list**: a character vector specifying what type of nodes to create; (m = a node with only mosquitoes, h = a node with only humans, b = a node with both humans and mosquitoes)
- **spn_P**: set of places produced by `spn_P_epiSIS_network`
- **params**: a named list of parameters (see details)
- **cube**: an inheritance cube from the `MGDrivE` package (e.g. `cubeMendelian`)
- **h_move**: binary adjacency matrix indicating if movement of humans between nodes is possible or not
- **m_move**: binary adjacency matrix indicating if movement of mosquitoes between nodes is possible or not

**Details**

This function takes the places produced from `spn_P_epiSIS_network` and builds all possible transitions between subsets of those places. The **params** argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the \( nE, nL, nP, \) and \( nEIP \) parameters to be specified. For more details, see `equilibrium_SEI_SIS`.

While this function produces all structural information related to transitions, hazards are produced by a separate function, `spn_hazards`.

For larger networks, this function may take some time to return, please be patient; the Petri Net modeling formalism trades additional computation time at model initialization for faster sampling of trajectories within a simulation.

Please note, the movement matrices (h\_move and m\_move) are NOT stochastic matrices, just binary matrices that say if i,j can exchange population. Diagonal elements must be FALSE, and both matrices are checked for validity; the function will stop with errors if the adjacency matrix specifies illegal movement rules (e.g.; mosquito movement from a "h" node to a "b" node).

For examples of using this function, see: vignette("epi-network", package = "MGDrivE2")
spn_T_epiSIS_node

Make Transitions ($T$) For a Node (SEI Mosquitoes - SIS Humans)

Description

This function makes the set of transitions ($T$) for a SPN. It is used alone if our model is a single-node metapopulation of mosquito and human dynamics; otherwise it is used as part of other functions to make SPN models with larger state spaces (metapopulation models, see `spn_T_epiSIS_network`).

Usage

```r
spn_T_epiSIS_node(spn_P, params, cube)
```

Arguments

- `spn_P`: set of places produced by `spn_P_epiSIS_node`
- `params`: a named list of parameters (see details)
- `cube`: an inheritance cube from the MGDrivE package (e.g. `cubeMendelian`)

Details

This function takes the places produced from `spn_P_epiSIS_node` and builds all possible transitions between subsets of those places.

The `params` argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the $n_E$, $n_L$, $n_P$, and $n_{EIP}$ parameters to be specified. For more details, see `equilibrium_SEI_SIS`.

While this function produces all structural information related to transitions, hazards are produced by a separate function, `spn_hazards`.

For examples of using this function, see: `vignette("epi-node", package = "MGDrivE2")`

Value

a list with two elements: $T$ contains transitions packets as lists, $v$ is the character vector of transitions ($T$)
Description

This function makes the set of transitions (T) for a SPN model of a metapopulation network. It is the network version of `spn_T_lifecycle_node`.

Usage

```r
spn_T_lifecycle_network(spn_P, params, cube, n = NULL, m_move = NULL)
```

Arguments

- `spn_P`: set of places produced by `spn_P_lifecycle_network`
- `params`: a named list of parameters (see details)
- `cube`: an inheritance cube from the MGDrivE package (e.g. `cubeMendelian`)
- `n`: an integer giving the number of nodes
- `m_move`: binary adjacency matrix indicating if movement of mosquitoes between nodes is possible or not

Details

This function takes the places produced from `spn_P_lifecycle_network` and builds all possible transitions between subsets of those places.

The `params` argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the nE, nL, and nP parameters to be specified. For more details, see `equilibrium_lifecycle`.

While this function produces all structural information related to transitions, hazards are produced by a separate function, `spn_hazards`.

For larger networks, this function may take some time to return, please be patient; the Petri Net modeling formalism trades additional computation time at model initialization for faster sampling of trajectories within a simulation.

Please note, the movement matrix (m_move) is NOT a stochastic matrices, just a binary matrix that say if i,j can exchange population. Diagonal elements must be FALSE.

At least one of the arguments `n` and `m_move` must be provided. If both are provided `n` is ignored.

For examples of using this function, see: `vignette("lifecycle-network",package = "MGDrivE2")`

Value

a list with two elements: `T` contains transitions packets as lists, `v` is the character vector of transitions (T)
spn_T_lifecycle_node  

Make Transitions (T) For a Node (Mosquitoes only)

Description

This function makes the set of transitions (T) for a SPN. It is used alone if our model is a single-node metapopulation for mosquito dynamics only; otherwise it is used as part of other functions to make SPN models with larger state spaces (metapopulation models, see spn_T_lifecycle_network).

Usage

spn_T_lifecycle_node(spn_P, params, cube)

Arguments

- spn_P: set of places produced by spn_P_lifecycle_node
- params: a named list of parameters (see details)
- cube: an inheritance cube from the MGDrivE package (e.g. cubeMendelian)

Details

This function takes the places produced from spn_P_lifecycle_node and builds all possible transitions between subsets of those places.

The params argument supplies all of the ecological parameters necessary to calculate equilibrium values. This function requires the nE, nL, and nP parameters to be specified. For more details, see equilibrium_lifecycle.

While this function produces all structural information related to transitions, hazards are produced by a separate function, spn_hazards.

For examples of using this function, see: vignette("lifecycle-node", package = "MGDrivE2")

Value

- a list with two elements: T contains transitions packets as lists, v is the character vector of transitions (T)
**Description**

Make a function closure to implement a chemical Langevin (continuous-state) approximation for a SPN.

**Usage**

```
step_CLE(S, Sout, haz, dt = 0.01, maxhaz = 1e+06)
```

**Arguments**

- **S**: a stoichiometry `Matrix-class` object
- **Sout**: an optional matrix to track of event firings. In the continuous stochastic model this will the approximate cumulative intensity of each event.
- **haz**: a list of hazard functions
- **dt**: time-step for Euler-Maruyama method used to solve the SDE system
- **maxhaz**: maximum allowable hazard

**Details**

The chemical Langevin approximation is a numerical simulation of a Fokker-Planck approximation to the Master equations (Kolmogorov Forwards Equations) governing the stochastic model; the CLE approximation is a second-order approximation that will get the correct mean and variance but higher order moments will be incorrect.


Elements of the N list come from two places: The stoichiometry matrix (S) is generated in `spn_S` and the hazards (h) come from `spn_hazards`.

For other samplers, see: `step_PTS`, `step_DM`, `step_ODE`

**Value**

function closure for use in `sim_trajectory_R` or `sim_trajectory_CSV`
**step_DM**

*Make Gillespie’s Direct Method (DM) Sampler for a SPN model*

**Description**

Make a function closure to implement Gillespie’s Direct Method sampler for a SPN.

**Usage**

```r
step_DM(S, Sout, haz, maxhaz = 1e+06)
```

**Arguments**

- `S`: a stoichiometry `Matrix-class` object
- `Sout`: an optional matrix to track of event firings
- `haz`: a list of hazard functions
- `maxhaz`: maximum allowable hazard

**Details**

The direct method is an exact sampling algorithm; it simulates each event individually. Because of this it may be extremely slow for non-trivial population sizes, and thus should be used to debug and test rather than for serious Monte Carlo simulation.


Elements of the N list come from two places: The stoichiometry matrix (S) is generated in `spn_S` and the hazards (h) come from `spn_hazards`.

For other samplers, see: `step_CLE`, `step_PTS`, `step_ODE`

**Value**

function closure for use in `sim_trajectory_R` or `sim_trajectory_CSV`

**step_ODE**

*Make Mean-field Approximation (ODE) Numerical Integrator for a SPN Model*

**Description**

Make a function closure to implement a first order mean-field ODE approximation for a SPN.

**Usage**

```r
step_ODE(S, Sout, haz, method = "lsoda")
```
step_PTS

Arguments

S a stoichiometry Matrix-class object
Sout an optional matrix to track of event firings. In the deterministic case it will return
the rate of that event at the end of the time step
haz a list of hazard functions
method a character giving the type of numerical integrator used, the default is "lsoda"

Details

This method is equivalent to considering the ODEs describing the time evolution of the mean tra-
jectory (first moment) and setting all higher order moments which appear on the right hand side to
zero.

The solvers used within can be found in the deSolve package, see ode. For inhomogeneous sys-
tems, consider using the "rk4" method to avoid excessive integration times.

The stoichiometry matrix (S) is generated in spn_S.
The list of hazards (haz) come from spn_hazards.
For other samplers, see: step_CLE, step_PTS, step_DM

Value

function closure for use in sim_trajectory_R or sim_trajectory_CSV

step_PTS Make Poisson Time-Step (PTS) Sampler for a SPN Model

Description

Make a function closure to implement a Poisson time-step (tau-leaping with fixed tau) sampler for
a SPN.

Usage

step_PTS(S, Sout, haz, dt = 0.01, maxhaz = 1e+06)

Arguments

S a stoichiometry Matrix-class object
Sout an optional matrix to track of event firings
haz a list of hazard functions
dt time-step for tau-leap method
maxhaz maximum allowable hazard
**summarize_eggs_geno**

**Details**

This sampling algorithm is based on representing a SPN as a set of competing Poisson processes; it thus uses an integer valued state space but approximates the number of events over dt.


Elements of the N list come from two places: The stoichiometry matrix (S) is generated in spn_S and the hazards (h) come from spn_hazards.

For other samplers, see: step_CLE, step_DM, step_ODE

**Value**

function closure for use in sim_trajectory_R or sim_trajectory_CSV

**summarize_eggs_geno  Summarize Eggs by Genotype**

**Description**

This function summarizes egg stage by genotype. It calls base_aquatic_geno to do all of the work.

**Usage**

summarize_eggs_geno(out, spn_P)

**Arguments**

out the output of sim_trajectory_R

spn_P the places of the SPN, see details

**Details**

The places (spn_P) object is generated from one of the following: spn_P_lifecycle_node, spn_P_lifecycle_network, spn_P_epiSIS_node, spn_P_epiSIS_network, spn_P_epiSEIR_node, or spn_P_epiSEIR_network.

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, genotype, and value.

For examples of using this function, see: vignette("lifecycle-node",package = "MGDrivE2")

**Value**

a 3 to 5 column dataframe for plotting with ggplot2
summarize_eggs_stage  Summarize Eggs by Erlang-Stage

Description

This function summarizes egg stage by Erlang-stages. It calls base_aquatic_stage to do all of the work.

Usage

summarize_eggs_stage(out, spn_P)

Arguments

out  the output of sim_trajectory_R
spn_P  the places of the SPN, see details

Details

The places (spn_P) object is generated from one of the following: spn_P_lifecycle_node, spn_P_lifecycle_network, spn_P_epiSIS_node, spn_P_epiSIS_network, spn_P_epiSEIR_node, or spn_P_epiSEIR_network.

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, Erlang-stage, and value.

For examples of using this function, see: vignette("lifecycle-node",package = "MGDrivE2")

Value

a 3 to 5 column dataframe for plotting with ggplot2

summarize_females  Summarize Adult Females (One Node or Metapopulation Network, Lifecycle Model)

Description

For MGDrivE2 simulations of mosquito lifecycle dynamics in a single node or metapopulation network, this function sums over the male mate genotype to get population trajectories of adult female mosquitoes by their genotype.

Usage

summarize_females(out, spn_P)
summarize_females_epi

Arguments

out the output of `sim_trajectory_R`

spn_P the places of the SPN, see details

Details

The places (spn_P) object is generated from one of the following: `spn_P_lifecycle_node` or `spn_P_lifecycle_network`.

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, genotype, and value.

For examples of using this function, this or any vignette which visualizes output: `vignette("lifecycle-node", package = "MGDrivE2")`

Value

a 3 to 5 column dataframe for plotting with ggplot2

Description

For MGDrivE2 simulations of mosquito epidemiological dynamics in a single node or metapopulation network, this function sums over the male mate genotype as well as EIP bins to get population trajectories of adult female mosquitoes by their genotype and (S,E,I) status.

Usage

`summarize_females_epi(out, spn_P)`

Arguments

out the output of `sim_trajectory_R`

spn_P the places of the SPN, see details

Details

The places (spn_P) object is generated from one of the following: `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`.

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, inf, genotype, and value.

For examples of using this function, this or any vignette which simulates epi dynamics: `vignette("epi-node", package = "MGDrivE2")`
summarize_humans_epiSEIR

**Value**

a 4 to 6 column dataframe for plotting with ggplot2

---

**summarize_humans_epiSEIR**  
*Summarize Humans (One Node or Metapopulation Network, SEI Mosquitoes - SEIR Humans)*

---

**Description**

For MGDrivE2 simulations of mosquito epidemiological dynamics in a node or network, this function summarizes human infection status, S, E, I, and R. It uses `base_summarize_humans` to do all of the work.

**Usage**

`summarize_humans_epiSEIR(out)`

**Arguments**

- `out` the output of `sim_trajectory_R`

**Details**

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: `time`, `inf`, `genotype`, and `value`.  
For examples of using this function, see: vignette("seir-dynamics",package = "MGDrivE2")

**Value**

a 4 to 6 column dataframe for plotting with ggplot2

---

**summarize_humans_epiSIS**  
*Summarize Humans (One Node or Metapopulation Network, SEI Mosquitoes - SIS Humans)*

---

**Description**

For MGDrivE2 simulations of mosquito epidemiological dynamics in a node or network, this function summarizes human infection status, S and I. It uses `base_summarize_humans` to do all of the work.
summarize_larvae_geno  

Description

This function summarizes larval stage by genotype. It calls base_aquatic_geno to do all of the work.

Usage

summarize_larvae_geno(out, spn_P)

Arguments

out  the output of sim_trajectory_R
spn_P  the places of the SPN, see details

Details

The places (spn_P) object is generated from one of the following: spn_P_lifecycle_node, spn_P_lifecycle_network, spn_P_epiSIS_node, spn_P_epiSIS_network, spn_P_epiSEIR_node, or spn_P_epiSEIR_network.

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, genotype, and value.

For examples of using this function, see: vignette("lifecycle-node",package = "MGDrivE2")

Value

a 3 to 5 column dataframe for plotting with ggplot2
**summarize_larvae_stage**

*Summarize Larval by Erlang-Stage*

**Description**

This function summarizes larval stage by Erlang-stages. It calls `base_aquatic_stage` to do all of the work.

**Usage**

```
summarize_larvae_stage(out, spn_P)
```

**Arguments**

- `out` - the output of `sim_trajectory_R`
- `spn_P` - the places of the SPN, see details

**Details**

The places (`spn_P`) object is generated from one of the following: `spn_P_lifecycle_node`, `spn_P_lifecycle_network`, `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`.

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, Erlang-stage, and value.

For examples of using this function, see: vignette("lifecycle-node",package = "MGDrivE2")

**Value**

a 3 to 5 column dataframe for plotting with ggplot2

---

**summarize_males**

*Summarize Adult Males (One Node or Metapopulation Network)*

**Description**

For MGDrivE2 simulations of mosquito lifecycle dynamics or human infection dynamics, in a node or metapopulation network, this function summarizes population trajectories of adult male mosquitoes by their genotype.

**Usage**

```
summarize_males(out)
```
summarize_pupae_geno

Arguments

out        the output of sim_trajectory_R

Details

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, genotype, and value.

For examples of using this function, see: vignette("lifecycle-node",package = "MGDrivE2")

Value

a 3 to 5 column dataframe for plotting with ggplot2
summarize_pupae_stage  Summarize Pupal by Erlang-Stage

Description

This function summarizes pupal stage by Erlang-stages. It calls base_aquatic_stage to do all of the work.

Usage

summarize_pupae_stage(out, spn_P)

Arguments

out the output of sim_trajectory_R
spn_P the places of the SPN, see details

Details

The places (spn_P) object is generated from one of the following: spn_P_lifecycle_node, spn_P_lifecycle_network, spn_P_epiSIS_node, spn_P_epiSIS_network, spn_P_epiSEIR_node, or spn_P_epiSEIR_network.

The return object depends on the data provided. If the simulation was only 1 node, then no node designation is returned. If only one repetition was performed, no rep designation is returned. Columns always returned include: time, Erlang-stage, and value.

For examples of using this function, see: vignette("lifecycle-node",package = "MGDrivE2")

Value

a 3 to 5 column dataframe for plotting with ggplot2

summarize_stats_CSV  Summary Statistics for MGDrivE2

Description

This function reads in all repetitions for each patch and calculates either the mean, quantiles, or both. User chooses the quantiles, up to 4 decimal places, and enters them as a vector. Quantiles are calculated empirically. (order does not matter)
Usage

```r
summarize_stats_CSV(
  read_dir,
  write_dir = read_dir,
  mean = TRUE,
  quantiles = NULL,
  spn_P,
  tmax,
  dt,
  rem_file = FALSE,
  verbose = TRUE
)
```

Arguments

- `read_dir`: Directory to find repetition folders in
- `write_dir`: Directory to write output
- `mean`: Boolean, calculate mean or not. Default is `TRUE`
- `quantiles`: Vector of quantiles to calculate. Default is `NULL`
- `spn_P`: Places object, see details
- `tmax`: The final time to end simulation
- `dt`: The time-step at which to return output (not the time-step of the sampling algorithm)
- `rem_file`: Remove original output? Default is `FALSE`
- `verbose`: Chatty? Default is `TRUE`

Details

Given the `read_dir`, this function assumes the follow file structure:

- `read_dir`
  - repetition 1
    * `M_0001.csv`
    * `M_0002.csv`
    * `FS_0001.csv`
    * `FS_0001.csv`
    * ...
  - repetition 2
    * `M_0001.csv`
    * `M_0002.csv`
    * `FS_0001.csv`
    * `FS_0001.csv`
**track_hinf**

* ...
  - repetition 3
  - ...

The places (spn_P) object is generated from one of the following: `spn_P_lifecycle_node`, `spn_P_lifecycle_network`, `spn_P_epiSIS_node`, `spn_P_epiSIS_network`, `spn_P_epiSEIR_node`, or `spn_P_epiSEIR_network`.

t₀, tₜ, dt define the first sampling time, the last sampling time, and each sampling time in-between.

Output files are *.csv and contain the mean or quantile in the file name, e.g. `stageMean(patchNum).csv` and `stageQuantile(quantNum)(patchNum).csv`.

For more details about using this function to process CSV output see: vignette("data-analysis", package = "MGDrivE2")

**Value**

Writes output to files in write_dir

---

**track_hinf**  
*Make tracking matrix for human infection events*

**Description**

Create a matrix object for tracking incidence in human population to be passed to either `sim_trajectory_CSV` or `sim_trajectory_R`.

**Usage**

```
track_hinf(spn_T, S)
```

**Arguments**

- **spn_T** set of transitions
- **S** stoichiometry matrix

**Details**

The returned matrix can be passed to the `Sout` argument of `sim_trajectory_CSV` or `sim_trajectory_R`.

**Value**

A sparseMatrix object
Index

* datasets
  - mu_ts, 22
  - base_aquatic_gen, 3, 48, 52, 54
  - base_aquatic_stage, 4, 49, 53, 55
  - base_erlang, 5
  - base_erlang_F, 5
  - base_gen, 6
  - base_gen_F, 7
  - base_MQ, 8
  - base_MUH, 9
  - base_sum_F, 10
  - base_summarize_humans, 9, 51
  - batch_migration, 11, 12
  - calc_move_rate, 12
  - cubeMendelian, 14, 16, 18, 32, 34–44
  - equilibrium_lifecycle, 13, 16–18, 20, 32, 37, 38, 43, 44
  - equilibrium_SEI_SEIR, 14, 15, 20, 34, 35, 39, 40
  - equilibrium_SEI_SIS, 14, 17, 17, 32, 36, 41, 42
  - events, 23, 24, 26, 28
  - get_shape, 20
  - make_Q_SEI, 16, 19, 21
  - movement_prob2rate, 21
  - mu_ts, 22
  - ode, 26, 28, 47
  - sim_trajectory_base_CSV, 23
  - sim_trajectory_base_R, 24
  - sim_trajectory_CSV, 11, 23, 25, 28, 29, 31, 45–48, 57
  - sim_trajectory_R, 3, 4, 9, 11, 24, 26, 45–55, 57
  - solve_muAqua, 28
  - sparseMatrix, 57
  - split_aggregate_CSV, 5–7, 9, 10, 29
  - spn_hazards, 26, 27, 31, 40–48
  - spn_P_lifecycle_network, 3, 4, 14, 16, 19, 31–34, 37, 38, 39, 43, 48–50, 52–55, 57
  - spn_P_lifecycle_node, 3, 4, 14, 16, 19, 31–34, 37, 38, 39, 44, 48–50, 52–55, 57
  - spn_Post, 33, 38
  - spn_Pre, 33, 38
  - spn_S, 26, 27, 38, 45–48
  - spn_T_epiSEIR_network, 32–34, 39, 39, 40
  - spn_T_epiSEIR_node, 32–34, 39, 40
  - spn_T_epiSIS_network, 32–34, 39, 41, 42
  - spn_T_epiSIS_node, 32–34, 39, 41, 42
  - spn_T_lifecycle_network, 32–34, 39, 43, 44
  - spn_T_lifecycle_node, 32–34, 39, 43, 44
  - step_CLE, 26–28, 32, 45, 46–48
  - step_DM, 26, 28, 32, 45, 46, 47, 48
  - step_ODE, 26, 28, 32, 45, 46, 46, 48
  - step_PTS, 26–28, 32, 45–47, 47
  - summarize_eggs_gen, 3, 48
  - summarize_eggs_stage, 4, 49
  - summarize_females, 49
  - summarize_females_epi, 50
  - summarize_humans_epiSEIR, 10, 51
  - summarize_humans_epiSIS, 10, 51

58
summarize_larvae_gen, 3.52
summarize_larvae_stage. 4.53
summarize_males, 53
summarize_pupae_gen, 3.54
summarize_pupae_stage, 4.55
summarize_stats_CSV, 8.55

track_hinf, 26, 28, 57

unirout, 28