R topics documented:

- api_config_global .................................................. 4
- approx_equal .......................................................... 5
- compute_bloodmeal .................................................. 5
- compute_bloodmeal_simple ........................................... 6
- compute_emergents ................................................... 6
- compute_emergents.BH ............................................... 7
- compute_emergents.trace ............................................ 7
- compute_emergents.trace_deterministic ........................... 8
- compute_emergents.trace_stochastic ............................... 8
- compute_f .............................................................. 9
- compute_f.BQ .......................................................... 9
- compute_f.RM .......................................................... 10
- compute_f.trace ....................................................... 10
- compute_H .............................................................. 11
- compute_H.MOI ......................................................... 11
- compute_H.SIP ........................................................ 12
- compute_H.SIR ........................................................ 12
- compute_H.SIS ........................................................ 13
- compute_O .............................................................. 13
- compute_O.trace ...................................................... 14
- compute_oviposit ...................................................... 14
- compute_oviposit.BQ .................................................. 15
- compute_oviposit.BQ_deterministic ................................ 15
- compute_oviposit.BQ_stochastic .................................... 16
- compute_oviposit.RM .................................................. 16
- compute_oviposit.RM_deterministic ................................ 17
- compute_oviposit.RM_stochastic .................................... 17
- compute_oviposit.trace .............................................. 18
- compute_Psi ............................................................ 18
- compute_Psi.MOI ....................................................... 19
- compute_Psi.SIP ....................................................... 19
- compute_Psi.SIR ....................................................... 20
- compute_Psi.SIS ....................................................... 20
- compute_q .............................................................. 21
- compute_q.BQ .......................................................... 21
- compute_q.RM .......................................................... 22
- compute_q.trace ....................................................... 22
- compute_Wd ............................................................. 23
- compute_Wd.trace ..................................................... 23
- compute_wf ............................................................. 24
- compute_wf.MOI ....................................................... 24
- compute_wf.SIP ....................................................... 25
- compute_wf.SIR ....................................................... 25
- compute_wf.SIS ....................................................... 26
- compute_x .............................................................. 26
- compute_x.MOI ....................................................... 27
R topics documented:

- compute_x.SIP .......................................................... 27
- compute_x.SIR .......................................................... 28
- compute_x.SIS .......................................................... 28
- compute_xd .............................................................. 29
- compute_xd.trace ...................................................... 29
- compute_Z ............................................................... 30
- compute_Z.BQ .......................................................... 30
- compute_Z.RM .......................................................... 31
- compute_Z.trace ....................................................... 31
- distribute ............................................................... 32
- divmod ................................................................. 32
- draw_multinom ....................................................... 33
- get_config_alternative_trace ..................................... 33
- get_config_aqua_BH .................................................. 34
- get_config_aqua_trace .............................................. 35
- get_config_humans_MOI ............................................. 36
- get_config_humans_SIR ............................................. 37
- get_config_humans_SIS ............................................. 38
- get_config_mosquito_RM ........................................... 39
- get_config_mosquito_trace ....................................... 40
- get_config_visitor_trace ......................................... 41
- get_tmax .............................................................. 42
- get_tnow .............................................................. 42
- is_binary .............................................................. 42
- make_MicroMoB ...................................................... 43
- MicroMoB ............................................................. 43
- observe_pfpr .......................................................... 44
- observe_pfpr.SIP .................................................... 44
- observe_pfpr.SIS .................................................... 45
- output_aqua .......................................................... 46
- output_aqua.BH ...................................................... 46
- output_aqua.trace .................................................. 47
- output_mosquitoes .................................................. 47
- output_mosquitoes.RM ............................................. 48
- output_mosquitoes.trace ......................................... 48
- sample_stochastic_matrix ....................................... 49
- sample_stochastic_vector ....................................... 49
- setup_alternative_trace ......................................... 50
- setup_aqua_BH ....................................................... 50
- setup_aqua_trace ................................................... 51
- setup_humans_MOI .................................................. 51
- setup_humans_SIP ................................................... 53
- setup_humans_SIR ................................................... 54
- setup_humans_SIS ................................................... 55
- setup_mosquito_BQ .................................................. 56
- setup_mosquito_RM .................................................. 57
- setup_mosquito_trace ............................................. 58
- setup_visitor_trace ............................................... 58
api_config_global

Read global configuration options

Description

Read global configuration options

Usage

api_config_global(path)

Arguments

path file path to a JSON file
approx_equal

Check if two numeric values are approximately equal

Description
Check if two numeric values are approximately equal

Usage
approx_equal(a, b, tol = sqrt(.Machine$double.eps))

Arguments
- `a`: a numeric object
- `b`: a numeric object
- `tol`: the numeric tolerance

Value
a logical value

compute_bloodmeal

Compute bloodmeals taken by mosquitoes on hosts

Description
This should be run prior to any step functions to update components over a time step. It computes various quantities related to disease transmission between species using the generic interfaces (methods) provided by each component. It updates the EIR vector for the human component, and kappa, the net infectiousness of hosts for the mosquito component.

Usage
compute_bloodmeal(model)

Arguments
- `model`: an object from make_MicroMoB

Value
no return value
compute_bloodmeal_simple

*Compute bloodmeals taken by mosquitoes on hosts in simple models*

**Description**

The difference between this and `compute_bloodmeal` is that this function does not include any computations of alternative blood hosts or visitors and is suitable for models which only include mosquitoes and resident human populations.

**Usage**

```r
compute_bloodmeal_simple(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value

---

compute_emergents

*Compute number of newly emerging adults (λ)*

**Description**

This method dispatches on the type of `model$aqua`

**Usage**

```r
compute_emergents(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

a vector of length `p` giving the number of newly emerging adult in each patch
compute_emergents.BH

**compute_emergents.BH**

*Compute number of newly emerging adults from Beverton-Holt dynamics*

---

**Description**

This function dispatches on the second class attribute of `model$aqua` for stochastic or deterministic behavior.

**Usage**

```r
## S3 method for class 'BH'
compute_emergents(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

A vector of length 1 giving the number of newly emerging adult in each patch

---

compute_emergents.trace

*Compute number of newly emerging adults from forcing term*

---

**Description**

This function dispatches on the second class attribute of `model$aqua` for stochastic or deterministic behavior.

**Usage**

```r
## S3 method for class 'trace'
compute_emergents(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Details**

See `compute_emergents.trace_deterministic` and `compute_emergents.trace_stochastic`

**Value**

No return value
compute_emergents.trace_deterministic

*Description*

Compute number of newly emerging adults from forcing term (deterministic)

*Usage*

```r
## S3 method for class 'trace_deterministic'
compute_emergents(model)
```

*Arguments*

- `model` an object from `make_MicroMoB`

*Value*

a vector of length 1 giving the number of newly emerging adult in each patch

---

compute_emergents.trace_stochastic

*Description*

Draw a Poisson distributed number of emerging adults with mean parameter from the column of the trace matrix for this day.

*Usage*

```r
## S3 method for class 'trace_stochastic'
compute_emergents(model)
```

*Arguments*

- `model` an object from `make_MicroMoB`

*Value*

a vector of length 1 giving the number of newly emerging adult in each patch
compute_f

Compute mosquito feeding rate (f)

Description
This method dispatches on the type of model$mosquito

Usage
compute_f(model, B)

Arguments
model an object from make_MicroMoB
B a vector of length p giving total blood host availability by patch

Value
a vector of length p giving the per-capita blood feeding rate of mosquitoes in each patch

compute_f.BQ

Compute mosquito feeding rate for BQ model (f)

Description
Blood feeding rates are modeled as a Holling type 2 (rational) function of blood host availability.

\[ f(B) = f_x \frac{s_f B}{1 + s_f B} \]

Here \( f_x \) is the maximum blood feeding rate and \( s_f \) is a scaling parameter.

Usage
## S3 method for class 'BQ'
compute_f(model, B)

Arguments
model an object from make_MicroMoB
B a vector of length p giving total blood host availability by patch

Value
a vector of length p giving the per-capita blood feeding rate of mosquitoes in each blood feeding haunt
**compute_f.RM**

*Compute mosquito feeding rate for RM model (f)*

**Description**

This method simply returns the $f$ parameter of the mosquito object, because the RM model assumes a constant blood feeding rate.

**Usage**

```r
## S3 method for class 'RM'
compute_f(model, B)
```

**Arguments**

- **model**: an object from `make_MicroMoB`
- **B**: a vector of length $p$ giving total blood host availability by patch

**Value**

*a vector of length $p$ giving the per-capita blood feeding rate of mosquitoes in each patch*

---

**compute_f.trace**

*Compute null mosquito feeding rate (f)*

**Description**

Compute null mosquito feeding rate ($f$)

**Usage**

```r
## S3 method for class 'trace'
compute_f(model, B)
```

**Arguments**

- **model**: an object from `make_MicroMoB`
- **B**: a vector of length $p$ giving total blood host availability by patch

**Value**

*no return value*
compute_H

Compute human population strata sizes \((H)\)

Description

This method dispatches on the type of \texttt{model$human}.

Usage

\[ \text{compute}_H(\text{model}) \]

Arguments

\begin{itemize}
  \item \texttt{model} \hspace{1cm} an object from \texttt{make_MicroMoB}
\end{itemize}

Value

\[ \text{a vector of length} \ n \ \text{giving the size of each human population stratum} \]

compute_H.MOI

Compute human population strata sizes for MOI model \((H)\)

Description

Compute human population strata sizes for MOI model \((H)\)

Usage

\[ 
\#
\text{S3 method for class 'MOI''}
\text{compute}_H(\text{model})
\]

Arguments

\begin{itemize}
  \item \texttt{model} \hspace{1cm} an object from \texttt{make_MicroMoB}
\end{itemize}

Value

\[ \text{a vector of length} \ n \ \text{giving the size of each human population stratum} \]
compute_H.SIP

Compute human population strata sizes for SIP model ($H$)

Description

Compute human population strata sizes for SIP model ($H$)

Usage

```r
## S3 method for class 'SIP'
compute_H(model)
```

Arguments

- `model` an object from `make_MicroMoB`

Value

a vector of length $n$ giving the size of each human population stratum

compute_H.SIR

Compute human population strata sizes for SIR model ($H$)

Description

Compute human population strata sizes for SIR model ($H$)

Usage

```r
## S3 method for class 'SIR'
compute_H(model)
```

Arguments

- `model` an object from `make_MicroMoB`

Value

a vector of length $n$ giving the size of each human population stratum
**compute_H.SIS**

Compute human population strata sizes for SIS model (H)

**Description**

Compute human population strata sizes for SIS model (H)

**Usage**

```r
## S3 method for class 'SIS'
compute_H(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

a vector of length `n` giving the size of each human population stratum

---

**compute_O**

Compute available alternative blood hosts (O)

**Description**

This method dispatches on the type of `model$alternative`.

**Usage**

```r
compute_O(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

a vector of length `p` giving biting availability of other blood hosts at each patch
compute_0.trace  
*Compute available alternative blood hosts for trace model (O)*

Description

Compute available alternative blood hosts for trace model (O)

Usage

```r
## S3 method for class 'trace'
compute_0(model)
```

Arguments

- `model` an object from `make_MicroMoB`

Value

A vector of length `p` giving biting availability of other blood hosts at each patch

compute_oviposit  
*Compute number of eggs laid from oviposition for each patch*

Description

This method dispatches on the type of `model$mosquito`

Usage

```r
compute_oviposit(model)
```

Arguments

- `model` an object from `make_MicroMoB`

Value

A vector of length 1 giving the total number of eggs laid by adult mosquitoes in each aquatic habitat
compute_oviposit.BQ

Compute number of eggs laid from oviposition for each aquatic habitat for BQ model

Description
This method returns a vector of length 1.

Usage
## S3 method for class 'BQ'
compute_oviposit(model)

Arguments
model an object from make_MicroMoB

Details
see compute_oviposit.BQ_deterministic and compute_oviposit.BQ_stochastic

Value
a vector of length 1 giving the total number of eggs laid by adult mosquitoes in each aquatic habitat

compute_oviposit.BQ_deterministic

Compute number of eggs laid from oviposition for each patch for deterministic RM model

Description
Compute number of eggs laid from oviposition for each patch for deterministic RM model

Usage
## S3 method for class 'BQ_deterministic'
compute_oviposit(model)

Arguments
model an object from make_MicroMoB

Value
a vector of length 1 giving the total number of eggs laid by adult mosquitoes in each aquatic habitat
**compute_oviposit.BQ_stochastic**

*Compute number of eggs laid from oviposition for each patch for stochastic RM model*

**Description**

Compute number of eggs laid from oviposition for each patch for stochastic RM model.

**Usage**

```r
## S3 method for class 'BQ_stochastic'
compute_oviposit(model)
```

**Arguments**

- `model`: an object from `make_MicroMoB`.

**Value**

A vector of length `l` giving the total number of eggs laid by adult mosquitoes in each aquatic habitat.

---

**compute_oviposit.RM**

*Compute number of eggs laid from oviposition for each patch for RM model*

**Description**

This method returns a vector of length `p`.

**Usage**

```r
## S3 method for class 'RM'
compute_oviposit(model)
```

**Arguments**

- `model`: an object from `make_MicroMoB`.

**Details**

See `compute_oviposit.RM_deterministic` and `compute_oviposit.RM_stochastic`.

**Value**

A vector of length `p` giving the total number of eggs laid by adult mosquitoes in each patch.
**compute_oviposit.RM_deterministic**

*Compute number of eggs laid from oviposition for each patch for deterministic RM model*

---

**Description**

Compute number of eggs laid from oviposition for each patch for deterministic RM model

**Usage**

```r
## S3 method for class 'RM_deterministic'
compute_oviposit(model)
```

**Arguments**

- `model`: an object from `make_MicroMoB`

**Value**

a vector of length `p` giving the total number of eggs laid by adult mosquitoes in each patch

---

**compute_oviposit.RM_stochastic**

*Compute number of eggs laid from oviposition for each patch for stochastic RM model*

---

**Description**

Compute number of eggs laid from oviposition for each patch for stochastic RM model

**Usage**

```r
## S3 method for class 'RM_stochastic'
compute_oviposit(model)
```

**Arguments**

- `model`: an object from `make_MicroMoB`

**Value**

a vector of length `l` giving the total number of eggs laid by adult mosquitoes in each patch
compute_oviposit.trace

*Compute number of eggs laid from oviposition for each patch for null model*

**Description**

This method dispatches on the type of `model$mosquito`

**Usage**

```r
## S3 method for class 'trace'
compute_oviposit(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

A vector of length `p` giving the total number of eggs laid by adult mosquitoes in each patch

compute_Psi

*Compute time at risk matrix (Ψ)*

**Description**

The time at risk matrix is $Ψ = Θξ$. This method dispatches on the type of `model$human`.

**Usage**

```r
compute_Psi(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

A matrix with `n` rows and `p` columns, the time at risk matrix
**compute_Psi.MOI**  
*Compute time at risk matrix for MOI model (Ψ)*

**Description**  
Compute time at risk matrix for MOI model (Ψ)

**Usage**  
```r
## S3 method for class 'MOI'
compute_Psi(model)
```

**Arguments**
- `model`: an object from `make_MicroMoB`

**Value**
a matrix with \( n \) rows and \( p \) columns, the time at risk matrix

---

**compute_Psi.SIP**  
*Compute time at risk matrix for SIP model (Ψ)*

**Description**  
Compute time at risk matrix for SIP model (Ψ)

**Usage**  
```r
## S3 method for class 'SIP'
compute_Psi(model)
```

**Arguments**
- `model`: an object from `make_MicroMoB`

**Value**
a matrix with \( n \) rows and \( p \) columns, the time at risk matrix
Description
Compute time at risk matrix for SIR model ($\Psi$)

Usage
## S3 method for class 'SIR'
compute_Psi(model)

Arguments
model an object from make_MicroMoB

Value
a matrix with \(n\) rows and \(p\) columns, the time at risk matrix

Description
Compute time at risk matrix for SIS model ($\Psi$)

Usage
## S3 method for class 'SIS'
compute_Psi(model)

Arguments
model an object from make_MicroMoB

Value
a matrix with \(n\) rows and \(p\) columns, the time at risk matrix
**compute_q**

*Compute human blood feeding fraction (q)*

**Description**

This method dispatches on the type of `model$mosquito`

**Usage**

```r
compute_q(model, W, Wd, B)
```

**Arguments**

- `model` an object from `make_MicroMoB`
- `W` a vector of length `p` giving human availability by patch (`W`)
- `Wd` a vector of length `p` giving visitor availability by patch (`W_d`)
- `B` a vector of length `p` giving total blood host availability by patch (`B`)

**Value**

a vector of length `p` giving the proportion of bites taken on human hosts in each patch

---

**compute_q.BQ**

*Compute human blood feeding fraction for BQ model (q)*

**Description**

The human blood feeding fraction is simply the proportion of human hosts.

**Usage**

```r
## S3 method for class 'BQ'
compute_q(model, W, Wd, B)
```

**Arguments**

- `model` an object from `make_MicroMoB`
- `W` a vector of length `p` giving human availability by patch (`W`)
- `Wd` a vector of length `p` giving visitor availability by patch (`W_d`)
- `B` a vector of length `p` giving total blood host availability by patch (`B`)

**Value**

a vector of length `p` giving the proportion of bites taken on human hosts in each blood feeding haunt
### compute_q.RM

Compute human blood feeding fraction for RM model ($q$)

**Description**

This method simply returns the $q$ parameter of the mosquito object, because the RM model assumes a constant fraction of blood meals are taken on human hosts.

**Usage**

```r
## S3 method for class 'RM'
compute_q(model, W, Wd, B)
```

**Arguments**

- `model`: an object from make_MicroMoB
- `W`: a vector of length $p$ giving human availability by patch ($W$)
- `Wd`: a vector of length $p$ giving visitor availability by patch ($W_d$)
- `B`: a vector of length $p$ giving total blood host availability by patch ($B$)

**Value**

a vector of length $p$ giving the proportion of bites taken on human hosts in each patch

### compute_q.trace

Compute null human blood feeding fraction ($q$)

**Description**

Compute null human blood feeding fraction ($q$)

**Usage**

```r
## S3 method for class 'trace'
compute_q(model, W, Wd, B)
```

**Arguments**

- `model`: an object from make_MicroMoB
- `W`: a vector of length $p$ giving human availability by patch ($W$)
- `Wd`: a vector of length $p$ giving visitor availability by patch ($W_d$)
- `B`: a vector of length $p$ giving total blood host availability by patch ($B$)

**Value**

no return value
compute_Wd

**compute_Wd**  
Compute available visitors ($W_\delta$)

**Description**

This method dispatches on the type of `model$visitor`.

**Usage**

```r
compute_Wd(model)
```

**Arguments**

- `model` 
an object from `make_MicroMoB`

**Value**

a vector of length $p$ giving biting availability of visitors at each patch

---

compute_Wd.trace

**compute_Wd.trace**  
Compute available visitors for trace model ($W_\delta$)

**Description**

Compute available visitors for trace model ($W_\delta$)

**Usage**

```r
# S3 method for class 'trace'
compute_Wd(model)
```

**Arguments**

- `model` 
an object from `make_MicroMoB`

**Value**

a vector of length $p$ giving biting availability of visitors at each patch
**compute_wf**  
*Compute human biting weights (w_f)*

**Description**

This method dispatches on the type of `model$human`.

**Usage**

```r
compute_wf(model)
```

**Arguments**

- `model`  
an object from `make_MicroMoB`

**Value**

A vector of length \( n \) giving the biting weights of human hosts in each stratum.

---

**compute_wf.MOI**  
*Compute human biting weights for MOI model (w_f)*

**Description**

Compute human biting weights for MOI model (\( w_f \)).

**Usage**

```r
# S3 method for class 'MOI'
compute_wf(model)
```

**Arguments**

- `model`  
an object from `make_MicroMoB`

**Value**

A vector of length \( n \) giving the biting weights of human hosts in each stratum.
**compute_wf.SIP**

*Compute human biting weights for SIP model ($w_f$)*

**Description**

Compute human biting weights for SIP model ($w_f$)

**Usage**

```r
## S3 method for class 'SIP'
compute_wf(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

A vector of length $n$ giving the biting weights of human hosts in each stratum

**compute_wf.SIR**

*Compute human biting weights for SIR model ($w_f$)*

**Description**

Compute human biting weights for SIR model ($w_f$)

**Usage**

```r
## S3 method for class 'SIR'
compute_wf(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

A vector of length $n$ giving the biting weights of human hosts in each stratum
compute_x

### Description

In a Ross-Macdonald style transmission model, this is computed as

\[ x = cX \]

This method dispatches on the type of `model$human`.

#### Usage

```
compute_x(model)
```

#### Arguments

- `model` an object from `make_MicroMoB`

#### Value

- a vector of length \( n \) giving the net infectiousness of human hosts in each stratum
compute_x.MOI

Compute net infectiousness for MOI model \( (x) \)

**Description**

In the simple MOI (queueing) model here (M/M/inf), net infectiousness is considered not to vary with increasing MOI. It is calculated as

\[ c \cdot (1 - \frac{X_0}{H}) \]

where \( X_0 \) is the number of uninfected persons (multiplicity of infection of zero).

**Usage**

```r
## S3 method for class 'MOI'
compute_x(model)
```

**Arguments**

- `model`: an object from `make_MicroMoB`

**Value**

da vector of length \( n \) giving the net infectiousness of human hosts in each stratum

compute_x.SIP

Compute net infectiousness for SIP model \( (x) \)

**Description**

Compute net infectiousness for SIP model \( (x) \)

**Usage**

```r
## S3 method for class 'SIP'
compute_x(model)
```

**Arguments**

- `model`: an object from `make_MicroMoB`

**Value**

ada vector of length \( n \) giving the net infectiousness of human hosts in each stratum
compute_x.SIR

*Compute net infectiousness for SIR model (x)*

**Description**
Compute net infectiousness for SIR model (x)

**Usage**
```
## S3 method for class 'SIR'
compute_x(model)
```

**Arguments**
- `model`: an object from `make_MicroMoB`

**Value**
a vector of length \( n \) giving the net infectiousness of human hosts in each stratum

---

compute_x.SIS

*Compute net infectiousness for SIS model (x)*

**Description**
Compute net infectiousness for SIS model (x)

**Usage**
```
## S3 method for class 'SIS'
compute_x(model)
```

**Arguments**
- `model`: an object from `make_MicroMoB`

**Value**
a vector of length \( n \) giving the net infectiousness of human hosts in each stratum
compute_xd

Compute net infectiousness of visitors \((x_\delta)\)

Description

This method dispatches on the type of model$visitor.

Usage

compute_xd(model)

Arguments

model an object from make_MicroMoB

Value

a vector of length \(p\) giving net infectiousness of visitors at each patch

---

compute_xd.trace

Compute net infectiousness of visitors for trace model \((x_\delta)\)

Description

Compute net infectiousness of visitors for trace model \((x_\delta)\)

Usage

## S3 method for class 'trace'
compute_xd(model)

Arguments

model an object from make_MicroMoB

Value

a vector of length \(p\) giving net infectiousness of visitors at each patch
**compute_Z**

Compute density of infective mosquitoes ($Z$)

**Description**

This method dispatches on the type of `model$mosquito`. $Z$ is also known as the "sporozoite rate" in malariology.

**Usage**

```r
compute_Z(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

A vector of length $p$ giving the density of infected and infectious mosquitoes in each patch.

---

**compute_Z.BQ**

Compute density of infective mosquitoes for BQ model ($Z$)

**Description**

This method returns $Z$.

**Usage**

```r
## S3 method for class 'BQ'
compute_Z(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

A vector of length $p$ giving the density of infected and infectious mosquitoes in each blood feeding haunt.
compute_Z.RM

---

**compute_Z.RM**

*Compute density of infective mosquitoes for RM model (Z)*

**Description**

This method returns Z.

**Usage**

```r
## S3 method for class 'RM'
compute_Z(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

- a vector of length p giving the density of infected and infectious mosquitoes in each patch

---

compute_Z.trace

---

**compute_Z.trace**

*Compute null density of infective mosquitoes (Z)*

**Description**

Compute null density of infective mosquitoes (Z)

**Usage**

```r
## S3 method for class 'trace'
compute_Z(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

- no return value
distribute

Distribute items into bins as evenly as possible

Description

Distribute items into bins as evenly as possible.

Usage

distribute(n, p)

Arguments

n  number of bins
p  number of items

Value

A numeric vector of bin sizes

divmod

Division of integers

Description

Division of integers.

Usage

divmod(a, b)

Arguments

a  the dividend
b  the divisor

Value

A list with two elements, quo (quotient) and rem (remainder)
**draw_multinom**

*Draw a multinomially distributed random vector*

**Description**

Warning: this function does no argument checking. Ensure the arguments are as follows.

**Usage**

`draw_multinom(n, prob)`

**Arguments**

- `n` an integer giving the number of balls to distribute in bins
- `prob` a vector of probabilities for each bin, which must sum to one

**Value**

an integer vector of length equal to the length of `prob`

**Note**


---

**get_config_alternative_trace**

*Get parameters for trace driven alternative blood hosts*

**Description**

The JSON config file should have two entries:

- O: vector or matrix (see `time_patch_varying_parameter` for valid dimensions)

For interpretation of the entries, please read `setup_alternative_trace`.

**Usage**

`get_config_alternative_trace(path)`

**Arguments**

- `path` a file path to a JSON file
Value

a named list

Examples

# to see an example of proper JSON input, run the following
library(jsonlite)
par <- list(
  "O" = rep(1, 5)
)
toJSON(par, pretty = TRUE)

get_config_aqua_BH

Get parameters for aquatic (immature) model with Beverton-Holt dynamics

Description

The JSON config file should have two entries:

- stochastic: a boolean value
- molt: a scalar, vector, or matrix (row major)
- surv: a scalar, vector, or matrix (row major)
- K: a scalar, vector, or matrix (row major)
- L: a vector

Please see time_patch_varying_parameter for allowed dimensions of entries molt, surv, and K. L should be of length equal to the number of patches. For interpretation of the entries, please read setup_aqua_BH.

Usage

get_config_aqua_BH(path)

Arguments

path a file path to a JSON file

Value

a named list
Examples

# to see an example of proper JSON input, run the following
library(jsonlite)
p <- 5 # number of patches
t <- 10 # number of days to simulate
par <- list(
  "stochastic" = FALSE,
  "molt" = 0.3,
  "surv" = rep(0.5, 365),
  "K" = matrix(rpois(n = t * p, lambda = 100), nrow = p, ncol = t),
  "L" = rep(10, p)
)
toJSON(par, pretty = TRUE)

Description

The JSON config file should have two entries:

- stochastic: a boolean value
- lambda: a scalar, vector, or matrix (row major). It will be passed to time_patch_varying_parameter, see that function’s documentation for appropriate dimensions.

For interpretation of the entries, please read setup_aqua_trace.

Usage

get_config_aqua_trace(path)

Arguments

path a file path to a JSON file

Value

a named list

Examples

# to see an example of proper JSON input, run the following
library(jsonlite)
t <- 10 # number of days to simulate
par <- list(
  "stochastic" = FALSE,
  "lambda" = rpois(n = t, lambda = 10)
)
toJSON(par, pretty = TRUE)


get_config_humans_MOI  Get parameters for MOI human model

Description

The JSON config file should have 9 entries:

- stochastic: a boolean value
- theta: matrix (row major)
- wf: vector
- H: vector
- MOI: matrix (row major)
- b: scalar
- c: scalar
- r: scalar
- sigma: scalar

For interpretation of the entries, please read setup_humans_MOI.

Usage

get_config_humans_MOI(path)

Arguments

path a file path to a JSON file

Value

a named list

Examples

# to see an example of proper JSON input, run the following
library(jsonlite)
n <- 6 # number of human population strata
p <- 5 # number of patches
theta <- matrix(rexp(n*p), nrow = n, ncol = p)
theta <- theta / rowSums(theta)
H <- rep(10, n)
MOI <- matrix(0, nrow = 10, ncol = n)
MOI[1, ] <- H
par <- list("stochastic" = FALSE,
            "theta" = theta,
            "wf" = rep(1, n),
            "H" = H,
get_config_humans_SIR

"MOI" = MOI,
"b" = 0.55,
"c" = 0.15,
"r" = 1/200,
"sigma" = 1
)
toJSON(par, pretty = TRUE)

get_config_humans_SIR  Get parameters for SIR human model

Description

The JSON config file should have 8 entries:

- stochastic: a boolean value
- theta: matrix (row major)
- wf: vector
- H: vector
- SIR: matrix (row major)
- b: scalar
- c: scalar
- gamma: scalar

For interpretation of the entries, please read setup_humans_SIR.

Usage

get_config_humans_SIR(path)

Arguments

path  a file path to a JSON file

Value

a named list

Examples

# to see an example of proper JSON input, run the following
library(jsonlite)
n <- 6  # number of human population strata
p <- 5  # number of patches
theta <- matrix(rexp(n*p), nrow = n, ncol = p)
theta <- theta / rowSums(theta)
H <- rep(10, n)
SIR <- matrix(0, nrow = n, ncol = 3)
SIR[, 1] <- H
par <- list(
    "stochastic" = FALSE,
    "theta" = theta,
    "wf" = rep(1, n),
    "H" = H,
    "SIR" = SIR,
    "b" = 0.55,
    "c" = 0.15,
    "gamma" = 1/7
)
 toJSON(par, pretty = TRUE)

---

get_config_humans_SIS  Get parameters for SIS human model

Description
The JSON config file should have 8 entries:

- stochastic: a boolean value
- theta: matrix (row major)
- wf: vector
- H: vector
- X: vector
- b: scalar
- c: scalar
- r: scalar

For interpretation of the entries, please read setup_humans_SIS.

Usage
get_config_humans_SIS(path)

Arguments
path  a file path to a JSON file

Value
a named list
Examples

# to see an example of proper JSON input, run the following
library(jsonlite)

n <- 6  # number of human population strata
p <- 5  # number of patches
theta <- matrix(rexp(n*p), nrow = n, ncol = p)
theta <- theta / rowSums(theta)
H <- rep(10, n)
X <- rep(3, n)
par <- list(
  "stochastic" = FALSE,
  "theta" = theta,
  "wf" = rep(1, n),
  "H" = H,
  "X" = X,
  "b" = 0.55,
  "c" = 0.15,
  "r" = 1/200
)
toJSON(par, pretty = TRUE)

get_config_mosquito_RM

Get parameters for generalized Ross-Macdonald mosquito model

Description

The JSON config file should have 8 entries:

- stochastic: a boolean value
- f: scalar
- q: scalar
- eip: scalar or vector; see time_varying_parameter for valid formats
- p: scalar or vector; see time_varying_parameter for valid formats
- psi: matrix
- nu: scalar
- M: vector
- Y: vector
- Z: vector

For interpretation of the entries, please read setup_mosquito_RM.

Usage

get_config_mosquito_RM(path)
get_config_mosquito_trace

Arguments

path a file path to a JSON file

Value

a named list

Examples

# to see an example of proper JSON input, run the following
library(jsonlite)
t <- 10 # days to simulate
p <- 5 # number of patches
EIP <- rep(5, t)
p_surv <- 0.95
psi <- matrix(rexp(p^2), nrow = p, ncol = p)
psi <- psi / rowSums(psi)
par <- list(
  "stochastic" = FALSE,
  "f" = 0.3,
  "q" = 0.9,
  "eip" = EIP,
  "p" = p_surv,
  "psi" = psi,
  "nu" = 20,
  "M" = rep(100, p),
  "Y" = rep(20, p),
  "Z" = rep(5, p)
)
toJSON(par, pretty = TRUE)
get_config_visitor_trace

Value

a named list

Examples

# to see an example of proper JSON input, run the following
library(jsonlite)
par <- list(
  "oviposit" = rep(1, 5)
)
toJSON(par, pretty = TRUE)

description

The JSON config file should have two entries:

* Wd: vector or matrix (see time_patch_varying_parameter for valid dimensions)
* xd: vector or matrix (see time_patch_varying_parameter for valid dimensions)

For interpretation of the entries, please read setup_visitor_trace.

Usage

get_config_visitor_trace(path)

Arguments

path a file path to a JSON file

Value

a named list

Examples

# to see an example of proper JSON input, run the following
library(jsonlite)
par <- list(
  "Wd" = rep(1, 5),
  "xd" = rep(0.01, 365)
)
toJSON(par, pretty = TRUE)
**get_tmax**  
*Get maximum time of simulation from model object*

**Description**

Get maximum time of simulation from model object

**Usage**

`get_tmax(model)`

**Arguments**

- `model`: an object from `make_MicroMoB`

---

**get_tnow**  
*Get current time of simulation from model object*

**Description**

Get current time of simulation from model object

**Usage**

`get_tnow(model)`

**Arguments**

- `model`: an object from `make_MicroMoB`

---

**is_binary**  
*Does a numeric object consist of only zeros and ones?*

**Description**

Does a numeric object consist of only zeros and ones?

**Usage**

`is_binary(x)`

**Arguments**

- `x`: a numeric object
**Description**

The model object is a hashed environment. By default it contains a single list, `model$global` storing global state.

**Usage**

```r
make_MicroMoB(tmax, p, l = p)
```

**Arguments**

- `tmax` number of days to simulate
- `p` number of places
- `l` number of aquatic habitats (optional, will be set to `p` by default)

**Value**

an object of class `environment`

---

**MicroMoB**

*MicroMoB: Microsimulation for mosquito-borne pathogens*

**Description**

Discrete time simulation of mosquito-borne pathogen transmission

**Author(s)**

- **Maintainer**: Sean L. Wu <slwood89@gmail.com> ([ORCID](https://orcid.org/0000-0001-9380-4294))
- **Authors**:
  - David L. Smith <smitdave@uw.edu> ([ORCID](https://orcid.org/0000-0002-6678-7533))

**Other contributors**:

- Sophie Libkind [contributor]
observe_pfpr

Observe PfPR in human strata

Description

This method dispatches on the type of `model$human`.

Usage

`observe_pfpr(model, parameters)`

Arguments

- `model`: an object from `make_MicroMoB`
- `parameters`: a named list, should have elements `sens` (sensitivity), `spec` (specificity), and a vector of length equal to number of strata `testprop` which gives the proportion of each strata to be tested.

Value

An array of counts, with actual condition as first dimension and tested condition as the second dimension, and the third dimension is the human strata.

observe_pfpr.SIP

Observe PfPR in human strata for SIP model

Description

Observe PfPR in human strata for SIP model

Usage

```r
## S3 method for class 'SIP'
observe_pfpr(model, parameters)
```
**Arguments**

- `model` an object from `make_MicroMoB`
- `parameters` a named list, should have elements `sens` (sensitivity), `spec` (specificity), and a vector of length equal to number of strata `testprop` which gives the proportion of each strata to be tested.

**Value**

an array of counts, with actual condition as first dimension and tested condition as the second dimension, and the third dimension is the human strata

---

**Description**

Observe PfPR in human strata for SIS model

**Usage**

```r
## S3 method for class 'SIS'
observe_pfpr(model, parameters)
```

**Arguments**

- `model` an object from `make_MicroMoB`
- `parameters` a named list, should have elements `sens` (sensitivity), `spec` (specificity), and a vector of length equal to number of strata `testprop` which gives the proportion of each strata to be tested.

**Value**

an array of counts, with actual condition as first dimension and tested condition as the second dimension, and the third dimension is the human strata
output_aqua

*Get output for aquatic (immature) mosquito populations*

**Description**

This method dispatches on the type of model$aqua. It returns the current state of the aquatic component.

**Usage**

```r
output_aqua(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

a data.frame

---

output_aqua.BH

*Get output for aquatic (immature) mosquito populations with Beverton-Holt dynamics*

**Description**

Return a data.frame.

**Usage**

```r
## S3 method for class 'BH'
output_aqua(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

a data.frame with columns L (immature) and A (emerging pupae)
output_aqua.trace

Get output for aquatic (immature) mosquito populations with forced emergence

Description

This function returns an empty data.frame as trace models do not have endogenous dynamics.

Usage

```r
## S3 method for class 'trace'
output_aqua(model)
```

Arguments

- `model` an object from `make_MicroMoB`

Value

a data.frame

output_mosquitoes

Get output for mosquito populations

Description

This method dispatches on the type of `model$mosquito`. It returns the current state of the adult mosquito component.

Usage

```r
output_mosquitoes(model)
```

Arguments

- `model` an object from `make_MicroMoB`

Value

a data.frame
output_mosquitoes.RM  Get output for Ross-Macdonald mosquito populations

Description

Return a data.frame.

Usage

## S3 method for class 'RM'
output_mosquitoes(model)

Arguments

model  an object from make_MicroMoB

Value

a data.frame with columns M (all adult mosquitoes), Y (infected mosquitoes), and Z (infectious mosquitoes), and rows correspond to places.

output_mosquitoes.trace

Get output for null mosquito populations

Description

This function returns an empty data.frame as trace models do not have endogenous dynamics.

Usage

## S3 method for class 'trace'
output_mosquitoes(model)

Arguments

model  an object from make_MicroMoB

Value

a data.frame
sample_stochastic_matrix

Sample a stochastic matrix

Description

x is a matrix with arbitrary number of rows but whose columns are equal to the number of bins that the stochastic matrix prob parameterizes a distribution over. Each row of x gives a distribution of counts over bins and is resampled according to prob. It is conceptually similar to "stochastically" distributing the matrix as x %*% prob, which gives the expectation.

Usage

sample_stochastic_matrix(x, prob)

Arguments

x a matrix
prob a matrix, it must have number of columns equal to the number of columns of x and rows that sum to one

Value

a matrix whose dimensions equal the original x

sample_stochastic_vector

Sample a stochastic vector

Description

Given a vector of counts in cells, x and a stochastic matrix prob, each row of which describes a probability distribution of how that cell should be distributed among bins, sample destination bins for each cell count, and return a vector giving the number of counts in bins. It is conceptually similar to "stochastically" distributing the vector as x %*% prob, which gives the expectation.

Usage

sample_stochastic_vector(x, prob)

Arguments

x a vector
prob a matrix, it must have number of rows equal to x and rows that sum to one
**setup_aqua_BH**

*Setup aquatic (immature) mosquito model with Beverton-Holt dynamics*

**Description**
A single compartment for all aquatic stages is modeled which suffers density dependent mortality like the Beverton-Holt model.

**Usage**
```r
setup_aqua_BH(model, stochastic, molt, surv, K, L)
```

**Arguments**
- `model`: an object from `make_MicroMoB`
- `stochastic`: should the model update deterministically or stochastically?
- `molt`: proportion of immature stages which will mature and emerge as adults each day (may be time and patch varying see `time_patch_varying_parameter`)
- `surv`: daily survival probability (may be time and patch varying see `time_patch_varying_parameter`)
- `K`: carrying capacity (may be time and patch varying see `time_patch_varying_parameter`)
- `L`: initial number of immature mosquitoes

---

**setup_alternative_trace**

*Setup trace driven alternative blood hosts*

**Description**
This model complies with the visitors component interface. It adds a named list `model$alternative`.

**Usage**
```r
setup_alternative_trace(model, O = NULL)
```

**Arguments**
- `model`: an object from `make_MicroMoB`
- `O`: a time varying trace passed to `time_patch_varying_parameter` or `NULL` to set to 0 (no alternative blood hosts)

**Value**
no return value

---

**Value**
a vector of length equal to the number of columns of `prob`
**setup_aqua_trace**

**Details**

All parameters can be passed either as a vector of length equal to \( l \), a matrix with \( l \) rows and \( t_{\text{max}} \) columns, or a matrix with \( l \) rows and 365 columns.

**Value**

no return value

---

**setup_aqua_trace**

Setup aquatic (immature) mosquito model with trace (forced) emergence

---

**Description**

Emergence is passed as a (possibly time varying) parameter which is decoupled from the adult mosquito dynamics. This module assumes \( l \) and \( p \) are equivalent, as emergence rates are given for \( p \).

**Usage**

```
setup_aqua_trace(model, lambda, stochastic)
```

**Arguments**

- **model**
  - an object from `make_MicroMoB`
- **lambda**
  - daily emergence of mosquitoes, may be time and patch varying, see `time_patch_varying_parameter`
- **stochastic**
  - should the model update deterministically or stochastically?

**Value**

no return value

---

**setup_humans_MOI**

Setup humans with MOI (multiplicity of infection) pathogen model

---

**Description**

This is a queueing model (M/M/inf) of superinfection in humans.
Usage

```r
setup_humans_MOI(
  model,
  stochastic,
  theta,
  wf = NULL,
  H,
  MOI,
  b = 0.55,
  c = 0.15,
  r = 1/200,
  sigma = 1
)
```

Arguments

- **model**: an object from `make_MicroMoB`
- **stochastic**: should the model update deterministically or stochastically?
- **theta**: a time spent matrix
- **wf**: biting weights
- **H**: vector of strata population sizes
- **MOI**: a matrix giving the distribution of persons across strata (columns) and multiplicity of infection (rows).
- **b**: transmission efficiency (mosquito to human)
- **c**: transmission efficiency (human to mosquito)
- **r**: recovery rate (inverse of infectious duration)
- **sigma**: control non-independence of pathogen clearance; `sigma > 1` indicates competition (clearance is faster than independent) and `sigma < 1` indicates facilitation (clearance is slower than independent).

Value

- no return value

Note

The `step_humans` method for the MOI model will grow the MOI matrix (add rows) if an individual’s MOI exceeds the size of the matrix; therefore it’s a good idea to pad the input matrix with extra empty rows to avoid reallocating memory during the simulation as much as possible.
Description

A simple SIP (Susceptible-Infected-Protected) model

Usage

```r
setup_humans_SIP(
  model,
  stochastic, 
  theta,
  wf = NULL,
  SIP,
  b = 0.55,
  c = 0.15,
  r = 1/200, 
  rho = 0.07, 
  eta = 1/32 
)
```

Arguments

- `model` an object from `make_MicroMoB`
- `stochastic` should the model update deterministically or stochastically?
- `theta` a time spent matrix
- `wf` biting weights
- `SIP` matrix of strata (rows) by health states (SIP)
- `b` transmission efficiency (mosquito to human)
- `c` transmission efficiency (human to mosquito)
- `r` recovery rate (inverse of infectious duration)
- `rho` probability of treatment upon infection
- `eta` rate at which prophylaxis decays

Value

no return value
**setup_humans_SIR**

Setup humans with SIR infection model

**Description**

A simple SIR (Susceptible-Infected-Recovered) model

**Usage**

```r
setup_humans_SIR(
    model,  # model, an object from make_MicroMoB
    stochastic,  # should the model update deterministically or stochastically
    theta,  # a time spent matrix
    wf = NULL,  # biting weights
    H,  # vector of strata population sizes
    SIR,  # a matrix giving S, I, R counts (columns) for each strata (rows)
    b = 0.55,  # transmission efficiency (mosquito to human)
    c = 0.15,  # transmission efficiency (human to mosquito)
    gamma = 1/5  # rate of recovery
)
```

**Arguments**

- **model**: an object from `make_MicroMoB`
- **stochastic**: should the model update deterministically or stochastically?
- **theta**: a time spent matrix
- **wf**: biting weights
- **H**: vector of strata population sizes
- **SIR**: a matrix giving S, I, R counts (columns) for each strata (rows)
- **b**: transmission efficiency (mosquito to human)
- **c**: transmission efficiency (human to mosquito)
- **gamma**: rate of recovery

**Value**

no return value
**Description**

A simple SIS (Susceptible-Infected-Susceptible) model

**Usage**

```r
setup_humans_SIS(
  model,
  stochastic,
  theta,
  wf = NULL,
  H,
  X,
  b = 0.55,
  c = 0.15,
  r = 1/200
)
```

**Arguments**

- `model`: an object from `make_MicroMoB`
- `stochastic`: should the model update deterministically or stochastically?
- `theta`: a time spent matrix
- `wf`: biting weights
- `H`: vector of strata population sizes
- `X`: number of infectious persons in each strata
- `b`: transmission efficiency (mosquito to human)
- `c`: transmission efficiency (human to mosquito)
- `r`: recovery rate (inverse of infectious duration)

**Value**

no return value
**setup_mosquito_BQ**

*Setup blood feeding & oviposition (BQ) behavioral state mosquito model*

**Description**

This is a behavioral state model which allows for time varying EIP and survival probability. Mosquitoes transition between blood feeding (B) and oviposition (Q) depending on the success (or not) of those biological activities. It complies with the mosquito component interface, and may be simulated deterministically or stochastically.

**Usage**

```r
setup_mosquito_BQ(
  model,
  stochastic,
  eip,
  pB,
  pQ,
  psiQ,
  Psi_bb,
  Psi_bq,
  Psi_qb,
  Psi_qq,
  nu = 25,
  M,
  Y
)
```

**Arguments**

- `model`: an object from `make_MicroMoB`
- `stochastic`: should the model update deterministically or stochastically?
- `eip`: the Extrinsic Incubation Period (may be time varying see `time_varying_parameter`)
- `pB`: daily survival probability during blood feeding (may be time and patch varying see `time_patch_varying_parameter`)
- `pQ`: daily survival probability during oviposition (may be time and patch varying see `time_patch_varying_parameter`)
- `psiQ`: oviposition success probability (may be time and patch varying see `time_patch_varying_parameter`)
- `Psi_bb`: movement matrix from blood feeding haunts to blood feeding haunts (columns must sum to 1, p rows and columns)
- `Psi_bq`: movement matrix from blood feeding haunts to aquatic habitats (columns must sum to 1, 1 rows and p columns)
- `Psi_qb`: movement matrix from aquatic habitats to blood feeding haunts (columns must sum to 1, p rows and 1 columns)
setup_mosquito_RM

Psi_qq movement matrix from aquatic habitats to aquatic habitats (columns must sum to 1, 1 rows and columns)

nu number of eggs laid per oviposition

M number of susceptible mosquitoes (vector of length p + 1)

Y number of incubating mosquitoes (matrix with p + 1 rows and maxEIP + 1 columns)

Value

no return value

setup_mosquito_RM

Setup generalized Ross-Macdonald mosquito model

Description

This is a generalized RM model which allows for time varying EIP and survival probability. It com-
plies with the mosquito component interface, and may be simulated deterministically or stochasti-
cally.

Usage

setup_mosquito_RM(
  model,
  stochastic,
  f = 0.3,
  q = 0.9,
  eip,
  p,
  psi,
  nu = 25,
  M,
  Y,
  Z,
  N = NULL
)

Arguments

model an object from make_MicroMoB

stochastic should the model update deterministically or stochastically?

f the blood feeding rate

q the human blood feeding fraction

eip the Extrinsic Incubation Period (may be time varying see time_varying_parameter)

p daily survival probability (may be time and patch varying see time_patch_varying_parameter)

psi a mosquito dispersal matrix (rows must sum to 1)
**setup_visitor_trace**

**nu**  
number of eggs laid per oviposition

**M**  
total mosquito density per patch (vector of length p)

**Y**  
density of incubating mosquitoes per patch (vector of length p)

**Z**  
density of infectious mosquitoes per patch (vector of length p)

**N**  
1 by p matrix describing how eggs from mosquitoes in patches are distributed amongst aquatic habitats. If NULL it is the identity matrix of dimension 1.

**Value**

no return value

---

**setup_mosquito_trace**  
*Setup null mosquito model*

**Description**

This is a null model of mosquito dynamics that is only for testing/verifying aquatic models. It implements a single method `compute_oviposit.trace` and all other methods throw an error.

**Usage**

```r
setup_mosquito_trace(model, oviposit)
```

**Arguments**

- **model**  
an object from `make_MicroMoB`
- **oviposit**  
a vector of length p used as a return value for `compute_oviposit`

**Value**

no return value

---

**setup_visitor_trace**  
*Setup trace driven visitors*

**Description**

This model complies with the visitors component interface. It adds a named list `model$visitor`.

**Usage**

```r
setup_visitor_trace(model, Wd = NULL, xd = NULL)
```
**step_aqua**

**Arguments**

- **model**: an object from `make_MicroMoB`
- **Wd**: a time varying trace of visitor host availability passed to `time_patch_varying_parameter` or NULL to set to 0 (no visitors)
- **xd**: a time varying trace of visitor net infectiousness passed to `time_patch_varying_parameter` or NULL to set to 0 (no visitors)

**Value**

no return value

---

**step_aqua**  
*Update aquatic (immature) mosquito populations*

**Description**

This method dispatches on the type of `model$aqua`

**Usage**

`step_aqua(model)`

**Arguments**

- **model**: an object from `make_MicroMoB`

**Value**

no return value

---

**step_aqua.BH**  
*Update aquatic (immature) mosquito populations for Beverton-Holt dynamics*

**Description**

This function dispatches on the second class attribute of `model$aqua` for stochastic or deterministic behavior.

**Usage**

```r
## S3 method for class 'BH'
step_aqua(model)
```
Arguments
model an object from make_MicroMoB

Value
no return value

---

step_aqua.BH_deterministic

Update aquatic (immature) mosquito populations for deterministic Beverton-Holt dynamics

---

Description
Run a deterministic state update.

Usage
```r
## S3 method for class 'BH_deterministic'
step_aqua(model)
```

Arguments
model an object from make_MicroMoB

Value
no return value

---

step_aqua.BH_stochastic

Update aquatic (immature) mosquito populations for stochastic Beverton-Holt dynamics

---

Description
Run a stochastic state update.

Usage
```r
## S3 method for class 'BH_stochastic'
step_aqua(model)
```

Arguments
model an object from make_MicroMoB
Value
no return value

Description
This function does nothing as trace models do not have endogenous dynamics.

Usage
## S3 method for class 'trace'
step_aqua(model)

Arguments
model an object from make_MicroMoB

Value
no return value

step_humans
Update human population

Description
This method dispatches on the type of model$human.

Usage
step_humans(model)

Arguments
model an object from make_MicroMoB

Value
no return value
step_humans.MOI

Update MOI human model

Description
Update MOI human model

Usage

```r
## S3 method for class 'MOI'
step_humans(model)
```

Arguments

- `model` an object from `make_MicroMoB`

Value
no return value

---

step_humans.MOI_deterministic

Update MOI human model (deterministic)

Description
Update MOI human model (deterministic)

Usage

```r
## S3 method for class 'MOI_deterministic'
step_humans(model)
```

Arguments

- `model` an object from `make_MicroMoB`

Value
no return value
step_humans.MOI_stochastic

Update MOI human model (stochastic)

Description
Update MOI human model (stochastic)

Usage
## S3 method for class 'MOI_stochastic'
step_humans(model)

Arguments
model an object from make_MicroMoB

Value
no return value

step_humans.SIP

Update SIP human model

Description
Update SIP human model

Usage
## S3 method for class 'SIP'
step_humans(model)

Arguments
model an object from make_MicroMoB

Value
no return value
**step_humans.SIP_deterministic**

*Update SIP human model (deterministic)*

**Description**

Update SIP human model (deterministic)

**Usage**

```r
## S3 method for class 'SIP_deterministic'
step_humans(model)
```

**Arguments**

- `model`: an object from `make_MicroMoB`

**Value**

No return value

---

**step_humans.SIP_stochastic**

*Update SIP human model (stochastic)*

**Description**

Update SIP human model (stochastic)

**Usage**

```r
## S3 method for class 'SIP_stochastic'
step_humans(model)
```

**Arguments**

- `model`: an object from `make_MicroMoB`

**Value**

No return value
step_humans.SIR

**Description**

Update SIR human model

**Usage**

```r
## S3 method for class 'SIR'
step_humans(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value

---

step_humans.SIR_deterministic

**Description**

Update SIR human model (deterministic)

**Usage**

```r
## S3 method for class 'SIR_deterministic'
step_humans(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value
step_humans.SIS

*Update SIS human model*

Description

Update SIS human model

Usage

```r
## S3 method for class 'SIS'
step_humans(model)
```

Arguments

- `model`: an object from `make_MicroMoB`

Value

no return value

step_humans.SIS

*Update SIR human model (stochastic)*

Description

Update SIR human model (stochastic)

Usage

```r
## S3 method for class 'SIR_stochastic'
step_humans(model)
```

Arguments

- `model`: an object from `make_MicroMoB`

Value

no return value
### step_humans.SIS_deterministic

*Update SIS human model (deterministic)*

#### Description

Update SIS human model (deterministic)

#### Usage

```r
## S3 method for class 'SIS_deterministic'
step_humans(model)
```

#### Arguments

- `model`: an object from `make_MicroMoB`

#### Value

no return value

### step_humans.SIS_stochastic

*Update SIS human model (stochastic)*

#### Description

Update SIS human model (stochastic)

#### Usage

```r
## S3 method for class 'SIS_stochastic'
step_humans(model)
```

#### Arguments

- `model`: an object from `make_MicroMoB`

#### Value

no return value
step_mosquitoes

**Update mosquito population**

**Description**

This method dispatches on the type of `model$mosquito`

**Usage**

```r
step_mosquitoes(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value

---

step_mosquitoes.BQ

**Update blood feeding & oviposition (BQ) behavioral state mosquitoes**

**Description**

This function dispatches on the second argument of `model$mosquito` for stochastic or deterministic behavior.

**Usage**

```r
## S3 method for class 'BQ'
step_mosquitoes(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Details**

see `step_mosquitoes.BQ_deterministic` and `step_mosquitoes.BQ_stochastic`

**Value**

no return value
**step_mosquitoes.BQ_deterministic**

*Update blood feeding & oviposition (BQ) behavioral state mosquitoes (deterministic)*

**Description**

Update blood feeding & oviposition (BQ) behavioral state mosquitoes (deterministic)

**Usage**

```r
## S3 method for class 'BQ_deterministic'
step_mosquitoes(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value

---

**step_mosquitoes.BQ_stochastic**

*Update blood feeding & oviposition (BQ) behavioral state mosquitoes (stochastic)*

**Description**

Update blood feeding & oviposition (BQ) behavioral state mosquitoes (stochastic)

**Usage**

```r
## S3 method for class 'BQ_stochastic'
step_mosquitoes(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value
step_mosquitoes.RM  
**Update Ross-Macdonald mosquitoes**

**Description**
This function dispatches on the second argument of `model$mosquito` for stochastic or deterministic behavior.

**Usage**
```r
## S3 method for class 'RM'
step_mosquitoes(model)
```

**Arguments**
- `model`: an object from `make_MicroMoB`

**Details**
see `step_mosquitoes.RM_deterministic` and `step_mosquitoes.RM_stochastic`

**Value**
no return value

---

step_mosquitoes.RM_deterministic  
**Update Ross-Macdonald mosquitoes (deterministic)**

**Description**
Update Ross-Macdonald mosquitoes (deterministic)

**Usage**
```r
## S3 method for class 'RM_deterministic'
step_mosquitoes(model)
```

**Arguments**
- `model`: an object from `make_MicroMoB`

**Value**
no return value
### step_mosquitoes.RM_stochastic

*Update Ross-Macdonald mosquitoes (stochastic)*

**Description**

Update Ross-Macdonald mosquitoes (stochastic)

**Usage**

```r
## S3 method for class 'RM_stochastic'
step_mosquitoes(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value

### step_mosquitoes.trace

*Update null mosquito population*

**Description**

Update null mosquito population

**Usage**

```r
## S3 method for class 'trace'
step_mosquitoes(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value
strata_to_residency_counts

*Helper function for lumped population strata (counts)*

**Description**

If input is given as a matrix of population counts per strata (columns) and patch (rows), this function calculates the residency matrix and population size for the overall stratification of both residency and strata.

**Usage**

```r
strata_to_residency_counts(H_counts)
```

**Arguments**

- `H_counts`: a matrix of population counts

**Value**

a list with three elements:
- `J`: the residency matrix mapping elements in `H` to patches
- `H`: the overall population distribution over strata and patches

**Examples**

```r
# taken from package tests
J <- matrix(
  c(0.3, 0.5, 0.2,
    0.1, 0.6, 0.3), nrow = 3, ncol = 2, byrow = FALSE
)
H <- c(50, 60)
H_overall <- J %*% diag(H)
residency <- strata_to_residency_proportion(H_strata = H, J_strata = J)
```

---

strata_to_residency_proportion

*Helper function for lumped population strata (proportional assignment)*

**Description**

If input is given as a vector of population sizes per-strata, lumped over patches, and a separate matrix whose columns describe how each strata is distributed over patches, this function calculates the residency matrix and population size for the overall stratification of both residency and strata.
time_patch_varying_parameter

Usage

strata_to_residency_proportion(H_strata, J_strata)

Arguments

H_strata a vector of population size by strata
J_strata a matrix whose columns sum to one giving the distribution of strata (columns) populations over patches (rows)

Value

a list with three elements:

• assignment_indices: provides a mapping from patch (rows) and strata (columns) into the "unrolled" vector H
• J: the residency matrix mapping elements in H to patches
• H: the overall population distribution over strata and patches

Examples

# taken from package tests
J <- matrix(
  c(0.3, 0.5, 0.2,
    0.1, 0.6, 0.3), nrow = 3, ncol = 2, byrow = FALSE
)  
H <- c(50, 60)
# get the overall assignment of strata (cols) across patches (rows)
H_overall <- J %*% diag(H)
residency <- strata_to_residency_proportion(H_strata = H, J_strata = J)

---

time_patch_varying_parameter

Input parameters that may vary by time and patch

Description

Input parameters that may vary by time and patch

Usage

time_patch_varying_parameter(param, p, tmax)

Arguments

param if given a matrix, it must have nrows equal to p and ncols equal to either tmax or 365; if given a vector it must be of length p, tmax, or 365.
p number of patches
tmax number of time steps
**Description**

Input parameters that may vary by time

**Usage**

\[ \text{time\_varying\_parameter}(\text{param, tmax}) \]

**Arguments**

- **param**: a vector of length 1, \( \text{tmax} \), or 365.
- **tmax**: number of time steps

**Value**

a vector with \( \text{tmax} \) elements
<table>
<thead>
<tr>
<th>Function</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_config_global</td>
<td>4</td>
</tr>
<tr>
<td>approx_equal</td>
<td>5</td>
</tr>
<tr>
<td>array</td>
<td>44, 45</td>
</tr>
<tr>
<td>compute_bloodmeal</td>
<td>5, 6</td>
</tr>
<tr>
<td>compute_bloodmeal_simple</td>
<td>6</td>
</tr>
<tr>
<td>compute_emergents</td>
<td>6</td>
</tr>
<tr>
<td>compute_emergents.BH</td>
<td>7</td>
</tr>
<tr>
<td>compute_emergents.trace</td>
<td>7</td>
</tr>
<tr>
<td>compute_emergents.trace_deterministic</td>
<td>7, 8</td>
</tr>
<tr>
<td>compute_emergents.trace_stochastic</td>
<td>7, 8</td>
</tr>
<tr>
<td>compute_f</td>
<td>9</td>
</tr>
<tr>
<td>compute_f.BQ</td>
<td>9</td>
</tr>
<tr>
<td>compute_f.RM</td>
<td>10</td>
</tr>
<tr>
<td>compute_f.trace</td>
<td>10</td>
</tr>
<tr>
<td>compute_H</td>
<td>11</td>
</tr>
<tr>
<td>compute_H.MOI</td>
<td>11</td>
</tr>
<tr>
<td>compute_H.SIR</td>
<td>12</td>
</tr>
<tr>
<td>compute_H.SIS</td>
<td>13</td>
</tr>
<tr>
<td>compute_O</td>
<td>13</td>
</tr>
<tr>
<td>compute_O.trace</td>
<td>14</td>
</tr>
<tr>
<td>compute_oviposit</td>
<td>14, 58</td>
</tr>
<tr>
<td>compute_oviposit.BQ</td>
<td>15</td>
</tr>
<tr>
<td>compute_oviposit.BQ_deterministic</td>
<td>15, 15</td>
</tr>
<tr>
<td>compute_oviposit.BQ_stochastic</td>
<td>15, 16</td>
</tr>
<tr>
<td>compute_oviposit.RM</td>
<td>16</td>
</tr>
<tr>
<td>compute_oviposit.RM_deterministic</td>
<td>16, 17</td>
</tr>
<tr>
<td>compute_oviposit.RM_stochastic</td>
<td>16, 17</td>
</tr>
<tr>
<td>compute_oviposit.trace</td>
<td>18, 58</td>
</tr>
<tr>
<td>compute_Psi</td>
<td>18</td>
</tr>
<tr>
<td>compute_Psi.MOI</td>
<td>19</td>
</tr>
<tr>
<td>compute_Psi.SIP</td>
<td>19</td>
</tr>
<tr>
<td>compute_Psi.SIR</td>
<td>20</td>
</tr>
<tr>
<td>compute_Psi.SIS</td>
<td>20</td>
</tr>
<tr>
<td>compute_q</td>
<td>21</td>
</tr>
<tr>
<td>compute_q.BQ</td>
<td>21</td>
</tr>
<tr>
<td>compute_q.RM</td>
<td>22</td>
</tr>
<tr>
<td>compute_q.trace</td>
<td>22</td>
</tr>
<tr>
<td>compute_Wd</td>
<td>23</td>
</tr>
<tr>
<td>compute_Wd.trace</td>
<td>23</td>
</tr>
<tr>
<td>compute_wf</td>
<td>24</td>
</tr>
<tr>
<td>compute_wf.MOI</td>
<td>24</td>
</tr>
<tr>
<td>compute_wf.SIP</td>
<td>25</td>
</tr>
<tr>
<td>compute_wf.SIR</td>
<td>25</td>
</tr>
<tr>
<td>compute_wf.SIS</td>
<td>26</td>
</tr>
<tr>
<td>compute_x</td>
<td>26</td>
</tr>
<tr>
<td>compute_x.MOI</td>
<td>27</td>
</tr>
<tr>
<td>compute_x.SIP</td>
<td>27</td>
</tr>
<tr>
<td>compute_x.SIR</td>
<td>28</td>
</tr>
<tr>
<td>compute_x.SIS</td>
<td>28</td>
</tr>
<tr>
<td>compute_xd</td>
<td>29</td>
</tr>
<tr>
<td>compute_xd.trace</td>
<td>29</td>
</tr>
<tr>
<td>compute_Z</td>
<td>30</td>
</tr>
<tr>
<td>compute_Z.BQ</td>
<td>30</td>
</tr>
<tr>
<td>compute_Z.RM</td>
<td>31</td>
</tr>
<tr>
<td>compute_Z.trace</td>
<td>31</td>
</tr>
<tr>
<td>data.frame</td>
<td>46–48</td>
</tr>
<tr>
<td>distribute</td>
<td>32</td>
</tr>
<tr>
<td>divmod</td>
<td>32</td>
</tr>
<tr>
<td>draw_multinom</td>
<td>33</td>
</tr>
<tr>
<td>environment</td>
<td>43</td>
</tr>
<tr>
<td>get_config_alternative_trace</td>
<td>33</td>
</tr>
<tr>
<td>get_config_aqua_BH</td>
<td>34</td>
</tr>
<tr>
<td>get_config_aqua_trace</td>
<td>35</td>
</tr>
<tr>
<td>get_config_humans_MOI</td>
<td>36</td>
</tr>
<tr>
<td>get_config_humans_SIR</td>
<td>37</td>
</tr>
<tr>
<td>get_config_humans_SIS</td>
<td>38</td>
</tr>
<tr>
<td>get_config_mosquito_RM</td>
<td>39</td>
</tr>
<tr>
<td>get_config_mosquito_trace</td>
<td>40</td>
</tr>
<tr>
<td>get_config_visitor_trace</td>
<td>41</td>
</tr>
<tr>
<td>get_tmax</td>
<td>42</td>
</tr>
<tr>
<td>get_tnow</td>
<td>42</td>
</tr>
</tbody>
</table>
is_binary, 42
list, 34–38, 40, 41, 44, 45, 72, 73
make_MicroMoB, 5–31, 42, 43, 44–48, 50–71
MicroMoB, 43
MicroMoB-package (MicroMoB), 43
numeric, 5, 42
observe_pfpr, 44
observe_pfpr.SIP, 44
observe_pfpr.SIS, 45
output_aqua, 46
output_aqua.BH, 46
output_aqua.trace, 47
output_mosquitoes, 47
output_mosquitoes.RM, 48
output_mosquitoes.trace, 48
sample_stochastic_matrix, 49
sample_stochastic_vector, 49
setup_alternative_trace, 33, 50
setup_aqua_BH, 34, 50
setup_aqua_trace, 35, 51
setup_humans_MOI, 36, 51
setup_humans_SIP, 53
setup_humans_SIS, 38, 55
setup_mosquito_BQ, 56
setup_mosquito_RM, 39, 57
setup_mosquito_trace, 40, 58
setup_visitor_trace, 41, 58
step_aqua, 59
step_aqua.BH, 59
step_aqua.BH_deterministic, 60
step_aqua.BH_stochastic, 60
step_aqua.trace, 61
step_humans, 52, 61
step_humans.MOI, 62
step_humans.MOI_deterministic, 62
step_humans.MOI_stochastic, 63
step_humans.SIP, 63
step_humans.SIP_deterministic, 64
step_humans.SIP_stochastic, 64
step_humans.SIR, 65
step_humans.SIR_deterministic, 65
step_humans.SIR_stochastic, 66
step_humans.SIS, 66
step_humans.SIS_deterministic, 67
step_humans.SIS_stochastic, 67
step_mosquitoes, 68
step_mosquitoes.BQ, 68
step_mosquitoes.BQ_deterministic, 68, 69
step_mosquitoes.BQ_stochastic, 68, 69
step_mosquitoes.RM, 70
step_mosquitoes.RM_deterministic, 70, 70
step_mosquitoes.RM_stochastic, 70, 71
step_mosquitoes.trace, 71
strata_to_residency_counts, 72
strata_to_residency_proportion, 72
time_patch_varying_parameter, 33–35, 41, 50, 51, 56, 57, 59, 73
time_varying_parameter, 39, 56, 57, 74