Package ‘MicroMoB’

January 17, 2023

Type Package

Title Discrete Time Simulation of Mosquito-Borne Pathogen Transmission

Version 0.1.2

Description Provides a framework based on S3 dispatch for constructing models of mosquito-borne pathogen transmission which are constructed from submodels of various components (i.e. immature and adult mosquitoes, human populations). A consistent mathematical expression for the distribution of bites on hosts means that different models (stochastic, deterministic, etc.) can be coherently incorporated and updated over a discrete time step.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.1.9000

Imports abind, jsonlite

Suggests knitr, rmarkdown, testthat (>= 3.0.0), ggplot2, data.table, callr, httr, readr, withr, plumber

VignetteBuilder knitr

URL https://dd-harp.github.io/MicroMoB/

https://github.com/dd-harp/MicroMoB

BugReports https://github.com/dd-harp/MicroMoB/issues

NeedsCompilation yes

Author Sean L. Wu [aut, cre] (https://orcid.org/0000-0002-5781-9493),
 David L. Smith [aut] (https://orcid.org/0000-0003-4367-3849),
 Sophie Libkind [ctb]

Maintainer Sean L. Wu <slwood89@gmail.com>

Repository CRAN

Date/Publication 2023-01-17 03:40:02 UTC
R topics documented:

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

compute_x.SIP .................................................. 28
compute_x.SIR .................................................. 28
compute_x.SIS .................................................. 29
compute_xd ...................................................... 29
compute_xd.trace .............................................. 30
compute_Z ...................................................... 30
compute_Z.BQ .................................................. 31
compute_Z.RM .................................................. 31
compute_Z.trace ............................................... 32
distribute ...................................................... 32
divmod .......................................................... 33
draw_multinom ................................................... 33
get_config_alternative_trace ................................. 34
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_eip_mosquito_RM ......................................... 42
get_f_mosquito_RM ........................................... 42
get_kappa_mosquito_RM ...................................... 43
get_K_aqua_BH ................................................ 43
get_lambda_aqua_trace ...................................... 44
get_molt_aqua_BH ............................................ 44
get_nu_mosquito_RM .......................................... 45
get_psi_mosquito_RM ......................................... 45
get_p_mosquito_RM ........................................... 46
get_q_mosquito_RM ........................................... 46
get_surv_aqua_BH ............................................ 47
get_tmax ....................................................... 47
get_tnow ....................................................... 48
is_binary ......................................................... 48
make_MicroMoB ................................................ 49
MicroMoB ........................................................ 49
observe_pfpr .................................................... 50
observe_pfpr.SIP ............................................. 50
observe_pfpr.SIS ............................................. 51
output_aqua .................................................... 51
output_aqua.BH ............................................... 52
output_aqua.trace ........................................... 52
output_mosquitoes ........................................... 53
output_mosquitoes.RM ....................................... 53
output_mosquitoes.trace ................................... 54
sample_stochastic_matrix ................................. 54
sample_stochastic_vector ................................. 55
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>setup_alternative_trace</td>
<td>55</td>
</tr>
<tr>
<td>setup_aqua_BH</td>
<td>56</td>
</tr>
<tr>
<td>setup_aqua_trace</td>
<td>56</td>
</tr>
<tr>
<td>setup_humans_MOI</td>
<td>57</td>
</tr>
<tr>
<td>setup_humans_SIP</td>
<td>58</td>
</tr>
<tr>
<td>setup_humans_SIR</td>
<td>59</td>
</tr>
<tr>
<td>setup_humans_SIS</td>
<td>60</td>
</tr>
<tr>
<td>setup_mosquito_BQ</td>
<td>61</td>
</tr>
<tr>
<td>setup_mosquito_RM</td>
<td>62</td>
</tr>
<tr>
<td>setup_mosquito_trace</td>
<td>63</td>
</tr>
<tr>
<td>setup_visitor_trace</td>
<td>63</td>
</tr>
<tr>
<td>set_eip_mosquito_RM</td>
<td>64</td>
</tr>
<tr>
<td>set_f_mosquito_RM</td>
<td>65</td>
</tr>
<tr>
<td>set_kappa_mosquito_RM</td>
<td>65</td>
</tr>
<tr>
<td>set_K_aqua_BH</td>
<td>66</td>
</tr>
<tr>
<td>set_lambda_aqua_trace</td>
<td>66</td>
</tr>
<tr>
<td>set_molt_aqua_BH</td>
<td>67</td>
</tr>
<tr>
<td>set_nu_mosquito_RM</td>
<td>67</td>
</tr>
<tr>
<td>set_psi_mosquito_RM</td>
<td>68</td>
</tr>
<tr>
<td>set_p_mosquito_RM</td>
<td>68</td>
</tr>
<tr>
<td>set_q_mosquito_RM</td>
<td>69</td>
</tr>
<tr>
<td>set_surv_aqua_BH</td>
<td>69</td>
</tr>
<tr>
<td>step_aqua</td>
<td>70</td>
</tr>
<tr>
<td>step_aqua.BH</td>
<td>70</td>
</tr>
<tr>
<td>step_aqua.BH_deterministic</td>
<td>71</td>
</tr>
<tr>
<td>step_aqua.BH_stochastic</td>
<td>71</td>
</tr>
<tr>
<td>step_aqua.trace</td>
<td>72</td>
</tr>
<tr>
<td>step_humans</td>
<td>72</td>
</tr>
<tr>
<td>step_humans.MOI</td>
<td>73</td>
</tr>
<tr>
<td>step_humans.MOI_deterministic</td>
<td>73</td>
</tr>
<tr>
<td>step_humans.MOI_stochastic</td>
<td>74</td>
</tr>
<tr>
<td>step_humans.SIP</td>
<td>74</td>
</tr>
<tr>
<td>step_humans.SIP_deterministic</td>
<td>75</td>
</tr>
<tr>
<td>step_humans.SIP_stochastic</td>
<td>75</td>
</tr>
<tr>
<td>step_humans.SIR</td>
<td>76</td>
</tr>
<tr>
<td>step_humans.SIR_deterministic</td>
<td>76</td>
</tr>
<tr>
<td>step_humans.SIR_stochastic</td>
<td>77</td>
</tr>
<tr>
<td>step_humans.SIS</td>
<td>77</td>
</tr>
<tr>
<td>step_humans.SIS_deterministic</td>
<td>78</td>
</tr>
<tr>
<td>step_humans.SIS_stochastic</td>
<td>78</td>
</tr>
<tr>
<td>step_mosquitoes</td>
<td>79</td>
</tr>
<tr>
<td>step_mosquitoes.BQ</td>
<td>79</td>
</tr>
<tr>
<td>step_mosquitoes.BQ_deterministic</td>
<td>80</td>
</tr>
<tr>
<td>step_mosquitoes.BQ_stochastic</td>
<td>80</td>
</tr>
<tr>
<td>step_mosquitoes.RM</td>
<td>81</td>
</tr>
<tr>
<td>step_mosquitoes.RM_deterministic</td>
<td>81</td>
</tr>
<tr>
<td>step_mosquitoes.RM_stochastic</td>
<td>82</td>
</tr>
<tr>
<td>step_mosquitoes.trace</td>
<td>82</td>
</tr>
</tbody>
</table>
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**

```r
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 ($\lambda$)

**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 adults in each patch.

---

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 adults 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

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

**Description**

Return the column of the lambda matrix for this day.

**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**

*Compute number of newly emerging adults from forcing term (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**

```r
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**

```r
## 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 patch.

---

**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 `model$human`.

**Usage**

```r
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.MOI

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

*Usage*
```r
## S3 method for class 'MOI'
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.SIP

*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_0

*Compute available alternative blood hosts (O)*

Description

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

Usage

```r
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_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_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**

```r
## 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

```r
## 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 1 giving the total number of eggs laid by adult mosquitoes in each aquatic habitat
compute_oviposit.RM

Description
This method returns a vector of length p.

Usage
## 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

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

Usage
## 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
## S3 method for class 'RM_stochastic'
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 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
## 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

Description

The time at risk matrix is $\Psi = \Theta \xi$. 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

Description

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

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

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

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

**Usage**

```r
## 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
compute_Psi.SIS  Compute time at risk matrix for SIS model ($\Psi$)

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

Usage
```r
## 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  

**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 \((W_{\delta})\)
- **B** a vector of length \(p\) giving total blood host availability by patch \((B)\)

**Value**

no return value

---

compute_Wd  

**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 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  

Description

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

Usage

## 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  

Description

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

Usage

## 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
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

Description

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

Usage

```r
## S3 method for class 'SIS'
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

**compute_x**  
*Compute net infectiousness of humans* \((x)\)

**Description**

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

\[
x = cX
\]

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

**Usage**

```r
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/\(\infty\)), net infectiousness is considered not to vary with increasing MOI. It is calculated as

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

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**

a 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
a 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
```r
## 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**  
```r  
## 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_δ)*

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

**Usage**  
```r  
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**

*Description*

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

**Usage**

```r
## 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**

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

Description
This method returns Z.

Usage
## 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 null density of infective mosquitoes (Z)

Description

Compute null density of infective mosquitoes (Z)

Usage

## 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**

```r
# to see an example of proper JSON input, run the following
library(jsonlite)
par <- list(
    "0" = 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`. 
get_config_aqua_trace

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)

get_config_aqua_trace  Get parameters for aquatic (immature) model with forced emergence

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
get_config_humans_MOI

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,
  "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
get_config_humans_SIS

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)

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.
get_config_mosquito_trace

Get parameters for null mosquito model

Usage

get_config_mosquito_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 # 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

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(
    "oviposit" = rep(1, 5)
)
toJSON(par, pretty = TRUE)

---

get_config_visitor_trace

*Get parameters for trace driven visitors*

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_eip_mosquito_RM**  
*Get extrinsic incubation period for Ross-Macdonald mosquito model*

**Description**
Get extrinsic incubation period for Ross-Macdonald mosquito model

**Usage**
```r
get_eip_mosquito_RM(model, times)
```

**Arguments**
- `model`: an object from `make_MicroMoB`
- `times`: vector of times to return

**Value**
no return value

---

**get_f_mosquito_RM**  
*Get feeding rate for Ross-Macdonald mosquito model*

**Description**
Get feeding rate for Ross-Macdonald mosquito model

**Usage**
```r
get_f_mosquito_RM(model)
```

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

**Value**
a vector
**get_kappa_mosquito_RM**  
*Get kappa for Ross-Macdonald mosquito model*

**Description**  
Get kappa for Ross-Macdonald mosquito model

**Usage**  
`get_kappa_mosquito_RM(model)`

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

**Value**  
a vector

---

**get_K_aqua_BH**  
*Get carrying capacity for Beverton-Holt aquatic mosquito model*

**Description**  
Get carrying capacity for Beverton-Holt aquatic mosquito model

**Usage**  
`get_K_aqua_BH(model, times, places)`

**Arguments**  
- `model` an object from `make_MicroMoB`
- `times` vector of times to get values
- `places` vector of places to get values

**Value**  
a matrix
get_lambda_aqua_trace  
*Get daily emergence for Beverton-Holt aquatic mosquito model*

**Description**
Get daily emergence for Beverton-Holt aquatic mosquito model

**Usage**
```
get_lambda_aqua_trace(model, times, places)
```

**Arguments**
- `model`: an object from `make_MicroMoB`
- `times`: vector of times to get values
- `places`: vector of places to get values

**Value**
a matrix

get_molt_aqua_BH  
*Get daily maturation probability for Beverton-Holt aquatic mosquito model*

**Description**
Get daily maturation probability for Beverton-Holt aquatic mosquito model

**Usage**
```
get_molt_aqua_BH(model, times, places)
```

**Arguments**
- `model`: an object from `make_MicroMoB`
- `times`: vector of times to get values
- `places`: vector of places to get values

**Value**
a matrix
**get_nu_mosquito_RM**

Get number of eggs laid per oviposition for Ross-Macdonald mosquito model

**Description**

Get number of eggs laid per oviposition for Ross-Macdonald mosquito model

**Usage**

`get_nu_mosquito_RM(model)`

**Arguments**

model  
an object from `make_MicroMoB`

**Value**

a vector

---

**get_psi_mosquito_RM**

Get mosquito dispersal matrix for Ross-Macdonald mosquito model

**Description**

Get mosquito dispersal matrix for Ross-Macdonald mosquito model

**Usage**

`get_psi_mosquito_RM(model)`

**Arguments**

model  
an object from `make_MicroMoB`

**Value**

a matrix
get_p_mosquito_RM

*Description*
Get daily survival probability for Ross-Macdonald mosquito model

*Usage*
get_p_mosquito_RM(model, times, places)

*Arguments*
- **model**
  - an object from `make_MicroMoB`
- **times**
  - vector of times to get values
- **places**
  - vector of places to get values

*Value*
a matrix

get_q_mosquito_RM

*Description*
Get human blood feeding fraction for Ross-Macdonald mosquito model

*Usage*
get_q_mosquito_RM(model)

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

*Value*
a vector
get_surv_aqua_BH

Get daily survival probability for Beverton-Holt aquatic mosquito model

Description
Get daily survival probability for Beverton-Holt aquatic mosquito model

Usage
get_surv_aqua_BH(model, times, places)

Arguments

- model: an object from make_MicroMoB
- times: vector of times to get values
- places: vector of places to get values

Value
a matrix

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

Value
a logical value
Description

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

Usage

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

Description

Discrete time simulation of mosquito-borne pathogen transmission

Author(s)

Maintainer: Sean L. Wu <slwood89@gmail.com> (ORCID)

Authors:

• David L. Smith <smitdave@uw.edu> (ORCID)

Other contributors:

• Sophie Libkind [contributor]

See Also

Useful links:

• https://dd-harp.github.io/MicroMoB/
• https://github.com/dd-harp/MicroMoB
• Report bugs at https://github.com/dd-harp/MicroMoB/issues
**Observe PfPR in human strata**

**Description**

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

**Usage**

```r
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 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.
**observe_pfpr.SIS**  
*Observe PfPR in human strata for SIS model*

**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

## 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

## 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
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

Value

a vector of length equal to the number of columns of prob

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

setup_alternative_trace(model, O = NULL)

Arguments

model an object from make_MicroMoB
0    a time varying trace passed to time_patch_varying_parameter or NULL to set to 0 (no alternative blood hosts)

Value

no return value
### setup_aqua_BH

Set *up 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

**Details**

All parameters can be passed either as a vector of length equal to 1, a matrix with 1 rows and `tmax` columns, or a matrix with 1 rows and 365 columns.

**Value**

no return value

### setup_aqua_trace

Set *up 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**

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

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).
**setup_humans_SIP**

**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

**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 SIR (Susceptible-Infected-Recovered) model

**Usage**

```r
setup_humans_SIR(
  model,
  stochastic,
  theta,
  wf = NULL,
  H,
  SIR,
  b = 0.55,
  c = 0.15,
  gamma = 1/5
)
```

**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)
- **Psi_qq**: movement matrix from aquatic habitats to aquatic habitats (columns must sum to 1, p rows and columns)
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 complies with the mosquito component interface, and may be simulated deterministically or stochastically.

**Usage**

```r
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)
\( \nu \)  
number of eggs laid per oviposition

\( \mathbb{M} \)  
total mosquito density per patch (vector of length \( p \))

\( \mathbb{Y} \)  
density of incubating mosquitoes per patch (vector of length \( p \))

\( \mathbb{Z} \)  
density of infectious mosquitoes per patch (vector of length \( p \))

\( \mathbb{N} \)  
\( 1 \times p \) 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)
```
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

---

**set_eip_mosquito_RM**

Set extrinsic incubation period for Ross-Macdonald mosquito model

Description

Change the extrinsic incubation period parameter eip for some set of times. The new values eip should either be a scalar or a vector of length equal to the length of times.

Usage

```r
set_eip_mosquito_RM(model, eip, times)
```

Arguments

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

- **eip**: new extrinsic incubation period values

- **times**: vector of times to set the new values

Value

- no return value
**set_f_mosquito_RM**  
*Set feeding rate for Ross-Macdonald mosquito model*

**Description**  
Change the feeding rate parameter $f$.

**Usage**  
```r  
set_f_mosquito_RM(model, f)  
```

**Arguments**
- `model` an object from `make_MicroMoB`
- `f` new blood feeding rate

**Value**  
no return value

**set_kappa_mosquito_RM**  
*Set kappa for Ross-Macdonald mosquito model*

**Description**  
Change $\kappa$.

**Usage**  
```r  
set_kappa_mosquito_RM(model, kappa)  
```

**Arguments**
- `model` an object from `make_MicroMoB`
- `kappa` new value of $\kappa$

**Value**  
no return value
set_K_aqua_BH  Set carrying capacity for Beverton-Holt aquatic mosquito model

Description
Change the carrying capacity parameter K for some times and places. The parameter K is stored internally as a matrix so that times and places are used to modify a submatrix, therefore the new value K should either be a scalar value to update the entire submatrix or a matrix of places rows and times columns.

Usage
set_K_aqua_BH(model, K, times, places)

Arguments
model an object from make_MicroMoB
K new carrying capacity
times vector of times to set the new values
places vector of places to set the new values

Value
no return value

set_lambda_aqua_trace  Set daily emergence for trace (forced) aquatic mosquito model

Description
Change the daily emergence parameter lambda for some times and places. The parameter lambda is stored internally as a matrix so that times and places are used to modify a submatrix, therefore the new value lambda should either be a scalar value to update the entire submatrix or a matrix of places rows and times columns.

Usage
set_lambda_aqua_trace(model, lambda, times, places)

Arguments
model an object from make_MicroMoB
lambda new emergence
times vector of times to set the new values
places vector of places to set the new values
**set_molt_aqua_BH**

**Value**

no return value

---

**Description**

Change the daily maturation probability parameter molt for some times and places. The parameter molt is stored internally as a matrix so that times and places are used to modify a submatrix, therefore the new value molt should either be a scalar value to update the entire submatrix or a matrix of places rows and times columns.

**Usage**

```
set_molt_aqua_BH(model, molt, times, places)
```

**Arguments**

- **model**: an object from make_MicroMoB
- **molt**: new daily maturation probability
- **times**: vector of times to set the new values
- **places**: vector of places to set the new values

**Value**

no return value

---

**set_nu_mosquito_RM**

**Set number of eggs laid per oviposition for Ross-Macdonald mosquito model**

---

**Description**

Change the number of eggs laid per oviposition parameter nu.

**Usage**

```
set_nu_mosquito_RM(model, nu)
```

**Arguments**

- **model**: an object from make_MicroMoB
- **nu**: new number of eggs laid per oviposition
**set_p_mosquito_RM**

Set mosquito dispersal matrix for Ross-Macdonald mosquito model

**Description**
Change the mosquito dispersal matrix parameter psi.

**Usage**
```
set_psi_mosquito_RM(model, psi)
```

**Arguments**
- **model**: an object from make_MicroMoB
- **psi**: new mosquito dispersal matrix

**Value**
no return value

---

**set_p_mosquito_RM**

Set daily survival probability for Ross-Macdonald mosquito model

**Description**
Change the daily survival probability parameter p for some times and places. The parameter p is stored internally as a matrix so that times and places are used to modify a submatrix, therefore the new value p should either be a scalar value to update the entire submatrix or a matrix of places rows and times columns.

**Usage**
```
set_p_mosquito_RM(model, p, times, places)
```

**Arguments**
- **model**: an object from make_MicroMoB
- **p**: new human blood feeding fraction
- **times**: vector of times to set the new values
- **places**: vector of places to set the new values

**Value**
no return value
**set_q_mosquito_RM**  
*Set human blood feeding fraction for Ross-Macdonald mosquito model*

**Description**

Change the human blood feeding fraction parameter $q$.

**Usage**

```r
set_q_mosquito_RM(model, q)
```

**Arguments**

- `model`: an object from `make_MicroMoB`
- `q`: new human blood feeding fraction

**Value**

no return value

---

**set_surv_aqua_BH**  
*Set daily survival probability for Beverton-Holt aquatic mosquito model*

**Description**

Change the daily survival probability parameter $\text{surv}$ for some times and places. The parameter $\text{surv}$ is stored internally as a matrix so that `times` and `places` are used to modify a submatrix, therefore the new value $\text{surv}$ should either be a scalar value to update the entire submatrix or a matrix of `places` rows and `times` columns.

**Usage**

```r
set_surv_aqua_BH(model, surv, times, places)
```

**Arguments**

- `model`: an object from `make_MicroMoB`
- `surv`: new daily survival probability
- `times`: vector of times to set the new values
- `places`: vector of places to set the new values

**Value**

no return value
### step_aqua

**Description**

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

**Usage**

```r
step_aqua(model)
```

**Arguments**

- `model` an object from `make_MicroMoB`

**Value**

no return value

### step_aqua.BH

**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
step_aqua.trace

Update aquatic (immature) mosquito populations for forced emergence

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
**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**

```r
## 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**

```r
## 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

Update SIR human model

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

Update SIR human model (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.SIR_stochastic

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

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_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
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

---

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
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
# 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.
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
**time_varying_parameter**

**Value**
- a matrix with \( p \) rows and \( t_{\text{max}} \) columns

---

**time_varying_parameter**

*Input parameters that may vary by time*

---

**Description**
Input parameters that may vary by time

**Usage**

\[
\text{time_varying_parameter}(\text{param}, t_{\text{max}})
\]

**Arguments**

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

**Value**
- a vector with \( t_{\text{max}} \) elements
Index

api_config_global, 5
approx_equal, 5
array, 50, 51

compute_bloodmeal, 6, 6
compute_bloodmeal_simple, 6
compute_emergents, 7
compute_emergents.BH, 7
compute_emergents.trace, 8
compute_emergents.trace_deterministic, 8, 8
compute_emergents.trace_stochastic, 8, 9
compute_f, 9
compute_f.BQ, 10
compute_f.RM, 10
compute_f.trace, 11
compute_H, 11
compute_H.MOI, 12
compute_H.SIP, 12
compute_H.SIR, 13
compute_H.SIS, 13
compute_O, 14
compute_O.trace, 14
compute_oviposit, 15, 63
compute_oviposit.BQ, 15
compute_oviposit.BQ_deterministic, 15, 16
compute_oviposit.BQ_stochastic, 15, 16
compute_oviposit.RM, 17
compute_oviposit.RM_deterministic, 17, 17
compute_oviposit.RM_stochastic, 17, 18
compute_oviposit.trace, 18, 63
compute_Psi, 19
compute_Psi.MOI, 19
compute_Psi.SIP, 20
compute_Psi.SIR, 20
compute_Psi.SIS, 21
compute_q, 21

compute_q.BQ, 22
compute_q.RM, 22
compute_q.trace, 23
compute_Wd, 23
compute_Wd.trace, 24
compute_wf, 24
compute_wf.MOI, 25
compute_wf.SIP, 25
compute_wf.SIR, 26
compute_wf.SIS, 26
compute_x, 27
compute_x.MOI, 27
compute_x.SIP, 28
compute_x.SIR, 28
compute_x.SIS, 29
compute_xd, 29
compute_xd.trace, 30
compute_Z, 30
compute_Z.BQ, 31
compute_Z.RM, 31
compute_Z.trace, 32
data.frame, 51–54
distribute, 32
divmod, 33
draw_multinom, 33

environment, 49

get_config_alternative_trace, 34
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_eip_mosquito_RM, 42
get_f_mosquito_RM, 42
INDEX

get_\text{K}_aqua\_BH, 43
get_\text{kappa}_mosquito\_RM, 43
get_\text{lambda}_aqua\_trace, 44
get_\text{molt}_aqua\_BH, 44
get_\text{nu}_mosquito\_RM, 45
get_\text{p}_mosquito\_RM, 46
get_\text{psi}_mosquito\_RM, 45
get_\text{q}_mosquito\_RM, 46
get_\text{surv}_aqua\_BH, 47
get_\text{tmax}, 47
get_\text{tnow}, 48

\text{is\_binary}, 48

\text{list}, 34–36, 38–41, 50, 51, 83, 84

\text{make\_MicroMoB}, 6–32, 42–48, 49, 50–82
\text{matrix}, 43, 44, 47
\text{MicroMoB}, 49
\text{MicroMoB\_package (MicroMoB), 49}

\text{numeric}, 5, 48

\text{observe\_pfpr}, 50
\text{observe\_pfpr\_SIP}, 50
\text{observe\_pfpr\_SIS}, 51
\text{output\_aqua}, 51
\text{output\_aqua\_BH}, 52
\text{output\_aqua\_trace}, 52
\text{output\_mosquitoes}, 53
\text{output\_mosquitoes\_RM}, 53
\text{output\_mosquitoes\_trace}, 54
\text{sample\_stochastic\_matrix}, 54
\text{sample\_stochastic\_vector}, 55
\text{set\_eip\_mosquito\_RM}, 64
\text{set\_f\_mosquito\_RM}, 65
\text{set\_K\_aqua\_BH}, 66
\text{set\_kappa\_mosquito\_RM}, 65
\text{set\_lambda\_aqua\_trace}, 66
\text{set\_molt\_aqua\_BH}, 67
\text{set\_nu\_mosquito\_RM}, 67
\text{set\_p\_mosquito\_RM}, 68
\text{set\_psi\_mosquito\_RM}, 68
\text{set\_q\_mosquito\_RM}, 69
\text{set\_surv\_aqua\_BH}, 69
\text{setup\_alternative\_trace}, 34, 55
\text{setup\_aqua\_BH}, 34, 56
\text{setup\_aqua\_trace}, 35, 56
\text{setup\_humans\_MOI}, 36, 57
\text{setup\_humans\_SIP}, 58
\text{setup\_humans\_SIR}, 37, 59
\text{setup\_humans\_SIS}, 38, 60
\text{setup\_mosquito\_BQ}, 61
\text{setup\_mosquito\_RM}, 39, 62
\text{setup\_mosquito\_trace}, 40, 63
\text{setup\_visitor\_trace}, 41, 63
\text{step\_aqua}, 70
\text{step\_aqua\_BH}, 70
\text{step\_aqua\_BH\_deterministic}, 71
\text{step\_aqua\_BH\_stochastic}, 71
\text{step\_aqua\_trace}, 72
\text{step\_humans}, 58, 72
\text{step\_humans\_MOI}, 73
\text{step\_humans\_MOI\_deterministic}, 73
\text{step\_humans\_MOI\_stochastic}, 74
\text{step\_humans\_SIP}, 74
\text{step\_humans\_SIP\_deterministic}, 75
\text{step\_humans\_SIP\_stochastic}, 75
\text{step\_humans\_SIR}, 76
\text{step\_humans\_SIR\_deterministic}, 76
\text{step\_humans\_SIR\_stochastic}, 77
\text{step\_humans\_SIS}, 77
\text{step\_humans\_SIS\_deterministic}, 78
\text{step\_humans\_SIS\_stochastic}, 78
\text{step\_mosquitoes}, 79
\text{step\_mosquitoes\_BQ}, 79
\text{step\_mosquitoes\_BQ\_deterministic}, 79, 80
\text{step\_mosquitoes\_BQ\_stochastic}, 79, 80
\text{step\_mosquitoes\_RM}, 81
\text{step\_mosquitoes\_RM\_deterministic}, 81, 81
\text{step\_mosquitoes\_RM\_stochastic}, 81, 82
\text{step\_mosquitoes\_trace}, 82
\text{strata\_to\_residency\_counts}, 83
\text{strata\_to\_residency\_proportion}, 83
\text{time\_patch\_varying\_parameter}, 34, 35, 41, 55–57, 61, 62, 64, 84
\text{time\_varying\_parameter}, 39, 61, 62, 85