Package 'IxPopDyMod'

October 12, 2022

Title Framework for Tick Population and Infection Modeling

Version 0.2.0

Maintainer Myles Stokowski <mylesstokowski@gmail.com>

Description Code to specify, run, and then visualize and analyze the results of Ixodidae (hard-bodied ticks) population and infection dynamics models. Such models exist in the literature, but the source code to run them is not always available. 'IxPopDyMod' provides an easy way for these models to be written and shared.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

URL https://github.com/dallenmidd/IxPopDyMod

Depends R (>= 3.2.4)

Suggests knitr, rmarkdown

Imports magrittr, rlang, ggplot2, stringr, dplyr, tidyr, igraph, readr, yaml, tibble

Language en-US

NeedsCompilation no

Author Myles Stokowski [aut, cre], David Allen [aut] (<https://orcid.org/0000-0002-0712-9603>)

Repository CRAN

Date/Publication 2022-02-08 07:30:05 UTC

R topics documented:

`config` *Create a* `config` *object*

Description

Make a `config` object from the input parameters, and ensure that the inputs meet the requirements for the model. The returned object is a complete description of a model run scenario.

Usage

```
config(
  initial_population,
  transitions,
  parameters,
  predictors,
  steps,
  max_delay = 365L
)
```

Arguments

`initial_population`

Named numeric vector indicating starting population for each life stage. Life stages not specified are assumed to be 0.

`transitions` A `tibble` in which each row corresponds to a transition between two tick life stages, or a transition from a tick life stage to mortality.

from Tick life stage a transition is originating from, specified with a three character string. The final character specifies stage, with "e" = egg, "l" = larva, "n" = nymph, and "a" = adult. The middle character specifies infection, with "i" = infected, and "u" = uninfected. The first character is the current process or sub-stage, for example "q" = questing, "e" = engorged, and "r" = reproductive. We use "_" to indicate if any of these components is not relevant, for example "_" as the second character if we are ignoring infection.

to Tick life stage a transition is going to. May be specified with the same three character format as the "from" field. Alternatively, may be the one of the strings "m" or "per_capita_m" to indicate mortality.

transition_fun A string; the name of the function to use to calculate the value of the transition. Must either be a function included in the package or a custom function that has been loaded into the workspace.
Functions can take 0-2 predictors, and any number of parameters. Argument order matters - all transition functions must start with two predictor arguments (even if they are not used within the function), followed by any parameters. They must return a numeric vector. See `constant_fun`, `expo_fun` and `infect_fun` for examples for how to write custom functions.

delay If TRUE, transition is interpreted as a delay, if FALSE, transition is interpreted as a daily probability.

pred1 Specifies the first predictor to use in a transition function. One of NA, a string identical to a value of the "pred" column in the predictors table, or a pattern that matches at least one life stage.

pred2 Specifies the second predictor to use in a transition function. Format like pred1.

`parameters` A `tibble` of parameters to use in the transitions described in the transitions table. Each row corresponds to a parameter value that may be used in one or more transitions. Parameter values will be used in transitions where the "from" and "to" fields of the two (parameters and transitions) tables match.

from Used to identify the transitions that a parameter should be used for. Format like the "from" column in the transitions table, or a regex pattern that matches with one or more life stage strings.

to Used to identify the transitions that a parameter should be used for. Format like the "to" column in the transitions table, or a regex pattern that matches with one or more life stage strings.

param_name A string specifying the name of the argument in the function where you want to use a parameter.

host_spp Optional column, not needed for model configurations that do not dependent on host community. For a given row, NA if the parameter value

is not dependent on the host species. Otherwise, a string specifying the name of the host species that the parameter value pertains to.

param_value Numeric; the value of the parameter

`predictors` Optionally, a `tibble` of input data to be used as predictor values in transition functions, for example weather or host density.

pred String specifying the name of the predictor, e.g. "temp" or "host_den

pred_subcategory This column allows specifying predictors for which there are multiple values for a given j_day. Predictor values are sorted by this column in the config set up. This ensures that when accessing a predictor with multiple values for the same j_day, we get a vector of predictor values ordered by this column A typical use for this column is to specify the host density of each host species.

j_day Integer specifying the Julian day, or NA for predictors with constant value over time

value Numeric value of predictor

`steps` Numeric vector of length one indicating the duration to run the model over in days.

`max_delay` Numeric vector of length one. Determines the maximum number of days that a delayed transition can last.

Details

The delay column affects how a transition row is used in the model. In all cases, a transition row is evaluated with any parameters and predictors, resulting in a transition value, `t`. If there is another row with the same "from", but either "m" or "per_capita_m" for the "to" stage, this row will be evaluated as well, resulting in a mortality transition value, `m`. Only delay transitions support "per_capita_m".

In non-delay transitions (where `delay == FALSE`), ticks can either advance to the "to" stage, die, or remain in the "from" stage. In this case, `t` is interpreted as the probability that a tick in the "from" stage will advance to the "to" stage at the next time step. The survival rate, or the probability that a tick will remain in the same "from" life stage, is calculated as `1 - (t + m)`.

In delay transitions (where `delay == TRUE`), ticks can either advance to the "to" stage, or die - there is no survival. In this case, `t` is used to determine the number of days until ticks in the "from" stage will emerge as ticks in the "to" stage. `t` will be vectorized over each day from the current time step to `max_delay` days ahead. The duration of the transition (in days) will be the index `i` of the first element in `t` where the cumulative sum of `t[1:i]` is greater than or equal to 1.

Delay transitions support two modes of mortality, "m" and "per_capita_m". For transitions to "m", the mortality value `m` is interpreted as a daily probability of mortality for each day in the delay transition. This differs from transitions to "per_capita_m", where `m` is the total probability of mortality over the entire duration of the delay transition.'

Value

A `config` object

Examples

```
# We rebuild an example config from its constituent parts. This is successful as
# expected, because we're just making a config that's identical to an example.
do.call(config, config_ex_1)

# If we modify the config to something unsuitable, the function will complain.
# For example, if we modify the egg to larvae transition to use a different
# function that requires an additional parameter.

## Not run:
# We define a super simple function that takes two parameters.
prod_fun <- function(x, y, a, b) a * b

my_config <- config_ex_1
my_config$transitions[1, 3] <- 'prod_fun'

# this will throw an error, because a parameter is missing
do.call(config, my_config)
# config() will report that parameter "b" is missing for the exponential function.

# Adding the parameter should fix the config
my_config$parameters[9,] <- list(from = '__e', to = '__l', param_name = 'b',
                                 param_value = 1)

# Now, this should run without issues
do.call(config, my_config)

## End(Not run)
```

`config_ex_1` *Simple model configuration example*

Description

This model configuration uses only non-delay transitions, and no transitions depend on predictors (e.g. weather or host community). Parameter values are selected so that the population is stable over time.

Usage

```
config_ex_1
```

Format

A `config`

config_ex_2 *Simple model configuration example using delays*

Description

This model configuration uses delay transitions for all transitions except the adult to eggs transition. As in config_ex_1, no transitions depend on predictors, and the population is stable over time.

Usage

```
config_ex_2
```

Format

A config

constant_fun *Constant function*

Description

Constant function

Usage

```
constant_fun(x, y, a)
```

Arguments

x	Predictor 1 in transitions table. Not used in this function
y	Predictor 2 in transitions table. Not used in this function
a	Parameter a in parameters table.

Value

Numeric vector of length 1 equal to input parameter a

Examples

```
constant_fun(NULL, NULL, 1)
```

density_fun *Density dependent mortality*

Description

Density dependent mortality

Usage

```
density_fun(x, y, a, b, c, pref)
```

Arguments

x	Predictor 1 in transitions table. Numeric vector indicating host density for each of the host species. Length should be equal to the number of host species.
y	Predictor 2 in transitions table. Number of feeding ticks in life stages specified by predictor 2.
a	Parameter a in parameters table.
b	Parameter b in parameters table.
c	Parameter c in parameters table.
pref	Parameters named pref in parameters table. Numeric vector of length equal to the number of host species. Values are the preference for ticks in a given transition for each host species.

Value

Numeric vector of length 1, indicating mortality rate

Examples

```
density_fun(c(10, 20), 100, .1, .3, .2, c(.5, .8))
```

expo_fun *Exponential function*

Description

Exponential function

Usage

```
expo_fun(x, y, a, b)
```

Arguments

x	Predictor 1 in transitions table.
y	Predictor 2 in transitions table. Not used in this function.
a	Parameter a in parameters table.
b	Parameter b in parameters table.

Value

Numeric vector of length 1

Examples

```
expo_fun(.5, NULL, .1, .3)
```

---

feed_fun *Probability of actively questing and then finding a host*

Description

Probability of actively questing and then finding a host

Usage

```
feed_fun(x, y, a, pref, q, tmin, tmax)
```

Arguments

x	Predictor 1 in transitions table. Numeric vector indicating host density for each of the host species. Length should be equal to the number of host species.
y	Predictor 2 in transitions table. Numeric vector of length 1 indicating temperature.
a	Parameter a in parameters table.
pref	Parameters named pref in parameters table. Numeric vector of length equal to the number of host species. Values are the preference for ticks in a given transition for each host species.
q	Parameter q in parameters table. Used in Briere function.
tmin	Parameter tmin in parameters table. Indicates minimum temperature at which ticks actively quest.
tmax	Parameter tmax in parameters table. Indicates maximum temperature at which ticks actively quest.

Details

Product of binomial and Briere functions (prob of finding a host) * (prob of active questing)

Value

Numeric vector of length 1

Examples

```
feed_fun(10, 30, .001, .1, .5, 20, 40)
```

`find_n_feed` *Probability of finding a host and successfully feeding on it*

Description

Probability of finding a host and successfully feeding on it

Usage

```
find_n_feed(x, y, a, pref, feed_success)
```

Arguments

`x`	Predictor 1 in transitions table. Numeric vector indicating host density for each of the host species. Length should be equal to the number of host species.
`y`	Predictor 2 in transitions table. Not used in this function.
`a`	Parameter `a` in parameters table.
`pref`	Parameters named `pref` in parameters table. Numeric vector of length equal to the number of host species. Values are the preference for ticks in a given transition for each host species.
`feed_success`	Parameters named `feed success` in parameters table. Numeric vector of length equal to the number of host species. Values are the feeding success rate for ticks in a given transition while feeding on each host species.

Value

Numeric vector of length 1 indicating probability that ticks find any host and then successfully feed on that host.

Examples

```
find_n_feed(10, NULL, .1, 1, .5)
find_n_feed(runif(2) * 10, NULL, .1, runif(2), runif(2))
```

get_pred_from_table *Get a predictor from input data*

Description

Get a predictor from input data

Usage

```
get_pred_from_table(time, pred, table)
```

Arguments

time	Numeric vector of days to get data. Ignored if input is constant over time (as indicated by NA value in 'j_day' column)
pred	string specifying the name of the predictor, e.g. "host_den"
table	input predictors table

graph_lifecycle *Visualize transitions as a life cycle graph*

Description

Visualize transitions as a life cycle graph

Usage

```
graph_lifecycle(transitions)
```

Arguments

transitions	Tick transitions tibble

Details

This function could be used to visually confirm that a custom config has all the transitions intended

Value

None, plots a life cycle graph

Examples

```
graph_lifecycle(config_ex_1$transitions)
```

graph_population_each_group *Graph population size of each life stage over time*

Description

Graph population size of each life stage over time

Usage

```
graph_population_each_group(output)
```

Arguments

output	Model output; a tibble

Value

ggplot object

Examples

```
out <- run(config_ex_1)
graph_population_each_group(out)
```

graph_population_overall_trend *Graph overall trend in population*

Description

Graph overall trend in population

Usage

```
graph_population_overall_trend(output)
```

Arguments

output	Model output; a tibble

Details

See roughly whether population is increasing or decreasing. Calculates and plots the rate of change in number of adult ticks between consecutive days.

Value

ggplot object

Examples

```
# Make a new config that results in a population where some ticks remain
# in their life stage for multiple days.
my_config <- config_ex_1
my_config$parameters$param_value <- c(0.5, 0, 0.01, 0.95, 0.1, 0.8, 900, 0)
out <- run(my_config)
graph_population_overall_trend(out)
```

---

growth_rate *Calculate multiplicative growth rate of population*

Description

Calculate multiplicative growth rate of population

Usage

```
growth_rate(out)
```

Arguments

out	Model output data frame

Value

Numeric vector of length one representing growth rate

Examples

```
out <- run(config_ex_1)
growth_rate(out)
```

host_example_config *Configuration for showing how we can modify host community data*

Description

Configuration for showing how we can modify host community data

Usage

```
host_example_config
```

Format

A config

infect_example_config *Configuration for showing infection dynamics*

Description

Configuration for showing infection dynamics

Usage

```
infect_example_config
```

Format

A config

infect_fun *Probability that a feeding tick becomes engorged infected or uninfected*

Description

Probability that a feeding tick becomes engorged infected or uninfected

Usage

```
infect_fun(x, y, from_infected, to_infected, host_rc, pref)
```

Arguments

x	Predictor 1 in transitions table. Numeric vector indicating host density for each of the host species. Length should be equal to the number of host species.
y	Predictor 2 in transitions table. Not used in this function.
from_infected	Parameter from_infected in parameters table. Value should be 1 if transition is from an infected tick stage, 0 otherwise.
to_infected	Parameter to_infected in parameters table. Value should be 1 if transition is to an infected tick stage, 0 otherwise.
host_rc	Parameters named host_rc in parameters table. Numeric vector of length equal to the number of host species. Values are the host reservoir competence for each host species.
pref	Parameters named pref in parameters table. Numeric vector of length equal to the number of host species. Values are the preference for ticks in a given transition for each host species.

Details

Since density dependent mortality is subtracted later, in this function we assume that all feeding ticks feed successfully and become engorged.

Value

Numeric vector of length 1

Examples

```
infect_fun(10, NULL, 0, 0, .3, 1)
infect_fun(10, NULL, 0, 1, .3, 1)
infect_fun(10, NULL, 1, 1, .3, 1)
```

---

ogden2005 — *Configuration for Ixodes scapularis population dynamics model from Ogden et al. 2005*

Description

This model configuration recreates the Ixodes scapularis (blacklegged tick) population dynamics model from Ogden et al. 2005. This is a relatively complete model of tick population dynamics, including the effects of both temperature and the host community on tick life-stage transitions. We include this configuration to show that our package can be used to recreate existing models.

Usage

```
ogden2005
```

Format

A config

steps Number of time steps to run the model. Here each step corresponds to one day.

initial_population Named vector of initial population size. Here the population starts with 10000 questing adults.

transitions A tibble giving the transitions between tick life stages.

parameters A tibble with the parameters to the life-stage transitions functions.

predictors A tibble with the average temperature for each day, and density of hosts over the model run. Here the host community is stable with 20 deer and 200 rodents.

max_delay The number of time units used for the delay functions.

See Also

Ogden et al. (2005) doi: 10.1016/j.ijpara.2004.12.013

Examples

```
data(ogden2005)
## Not run:
output <- run(ogden2005)
graph_population_each_group(output)

## End(Not run)
```

---

ogden_feed_fun *Probability of actively questing times constant host finding probability*

Description

Probability of actively questing times constant host finding probability

Usage

```
ogden_feed_fun(x, y, a, q, tmin, tmax)
```

Arguments

x	Predictor 1 in transitions table. Numeric vector of length 1 indicating temperature.
y	Predictor 2 in transitions table. Not used in this function.
a	Parameter a in parameters table.
q	Parameter q in parameters table. Used in Briere function.
tmin	Parameter tmin in parameters table. Indicates minimum temperature at which ticks actively quest.
tmax	Parameter tmax in parameters table. Indicates maximum temperature at which ticks actively quest.

Details

(const prob of finding a host) * (prob of active questing)

Value

Numeric vector of length 1

See Also

Based on Ogden et al. (2005) doi: 10.1016/j.ijpara.2004.12.013

Examples

```
ogden_feed_fun(30, NULL, .03, .01, 10, 35)
```

---

read_config *create a config object from a YAML file*

Description

create a config object from a YAML file

Usage

```
read_config(file)
```

Arguments

file	YAML file to read

Value

A config object

Examples

```
## Not run:
read_config('cfg.yml')

## End(Not run)
```

run *Run the model*

Description

Run the model

Usage

```
run(cfg)
```

Arguments

cfg	An IxPopDyMod::config object

Value

Data frame of population of ticks of each life stage each day

Examples

```
run(config_ex_1)
```

run_all_configs *Run the model for each config*

Description

Simple convenience wrapper for calling run on each config in a list

Usage

```
run_all_configs(configs, parallel = FALSE)
```

Arguments

configs	List of config objects
parallel	Logical; if TRUE, run on all cores using parallel package.

Value

A list of data frame model outputs like those returned by run()

Examples

```
# run two example configs and save results

## Not run:
outputs <- run_all_configs(list(config_ex_1, config_ex_2))

## End(Not run)
```

`snow_cover_fun` *Mortality as a function of whether there is a snow on the ground*

Description

Mortality as a function of whether there is a snow on the ground

Usage

```
snow_cover_fun(x, y, no_snow_mort, snow_mort)
```

Arguments

`x`	amount of snow on ground
`y`	not used in this transition function
`no_snow_mort`	mortality with no snow on the ground
`snow_mort`	mortality with snow on the ground

`temp_example_config` *Configuration for showing how we can modify climate data*

Description

Configuration for showing how we can modify climate data

Usage

```
temp_example_config
```

Format

A `config`

vary_many_params *Generate copies of a* config *with all combinations of modified parameters*

Description

Generate copies of a config with all combinations of modified parameters

Usage

```
vary_many_params(cfg, param_rows, values_list)
```

Arguments

cfg	Base config to make modified copies of
param_rows	Numeric vector indicating the rows in the parameters table where parameter values should be modified. Length must equal length of values_list
values_list	List of numeric vectors. The values of a vector values_list[[i]] are the parameter values to use for the parameter identified by param_rows[[i]]

Value

A list of configs

Examples

```
# create new configs with different values for the parameter determining
# mortality of eggs (which is found in row 2) and that determining
# mortality of larvae (which is found in row 4)
cfgs <- vary_many_params(config_ex_1,
                         param_rows = c(2, 4),
                         values_list = list(c(0, 0.1), c(.99, .98)))

# inspect parameter rows 2 and 4 in each of the new configs to verify that we
# have the new values
lapply(cfgs, function(cfg) cfg$parameters[c(2, 4), 'param_value'])
```

vary_param *Generate copies of a* config *with a modified parameter*

Description

Create copies of a config with a modified parameter. These new configs can be used to see how that parameter affects the model

Usage

```
vary_param(
  cfg,
  param_row = NA,
  to = NA,
  from = NA,
  param_name = NA,
  host_spp = NA,
  values
)
```

Arguments

cfg	Base config to make modified copies of
param_row	Row number of parameter to vary, if this is specified arguments from, to, param_name, and host_spp are unneeded
to	The to life stage of the parameter to change.
from	The from life stage from of the parameter to change. If this is given, to and param_name are also needed.
param_name	The name of the parameter to change
host_spp	The host_spp identifying the parameter to change. Needed only if there are multiple rows in the parameter table with the same from, to and param_name, but different host_spp.
values	Numeric vector of values to use for parameter

Value

A list of configs

Examples

```
# create new configs with different values for the parameter determining
# mortality of eggs (which is found in row 2)
cfgs <- vary_param(config_ex_1, param_row = 2, values = c(0, 0.1, 0.2))

# inspect parameter row 2 in each of the new configs to verify that we have
```

```
# the new values
lapply(cfgs, function(cfg) cfg$parameters[[2, 'param_value']])
```

`winter_tick` *Configuration for winter tick population dynamics model*

Description

This is a model configuration based on a literature search on the factors affect the winter tick life cycle. Many of the transitions and parameters in this configuration are drawn from Drew and Samuel (1986). We include this configuration to show that our package is flexible for modeling multiple tick species with different life histories.

Usage

```
winter_tick
```

Format

An object of class `config` of length 6.

See Also

Drew and Samuel (1986) doi: 10.1139/z86105

Drew and Samuel (1985) doi: 10.7589/0090355821.3.274

Addison and McLaughlin (1988) doi: 10.2307/3282188

Ogden et al. (2005) doi: 10.1016/j.ijpara.2004.12.013

Examples

```
data(winter_tick)
## Not run:
output <- run(winter_tick)
graph_population_each_group(winter_tick)

## End(Not run)
```

`write_config` *Save a* `config` *object as files*

Description

Write a `config` object as a YAML file, and write all dataframe components (transitions, parameters, predictors) as csv files. All paths must be explicitly specified as arguments. This function will not allow overwriting files.

Usage

```
write_config(
  cfg,
  config_path,
  transitions_path,
  parameters_path,
  predictors_path
)
```

Arguments

`cfg`	A `config` object
`config_path`	Path to the output YAML config file
`transitions_path`	Path to output transitions csv
`parameters_path`	Path to output parameters csv
`predictors_path`	Path to output predictors csv

Value

None, writes config components to disk

Examples

```
## Not run:
write_config(config_ex_1, 'cfg.yml', 'trans.csv', 'params.csv',
             'predictors.csv')

## End(Not run)
```

Index