

Package ‘MixFishSim’

August 7, 2017

Title Mixed Fishery fleet dynamics simulation tool
Version 0.0.0.9000
Description A simulation framework for evaluating fleet dynamics in mixed fisheries.
Depends R (>= 3.3.1),
Imports spat
License What license is it under?
Encoding UTF-8
LazyData true
RoxygenNote 5.0.1
Suggests testthat

R topics documented:

baranov_f	1
create_fields	2
delay_diff	3
find_f	4
Index	5

baranov_f	<i>Baranov F</i>
-----------	------------------

Description

baranov_f provides the function to solve in [find_f](#) for estimating weekly fishing mortality from catch (*C*), biomass (*B*) and natural mortality (*M*). It’s based on the standard Baranov catch equation.

Usage

baranov_f(F, C, B, M)

Arguments

F	is the fishing mortality rate to solve.
C	is a Numeric vector detailing the catch at wk_t
B	is a Numeric vector of the biomass at wk_t
M	is a Numeric vector of the natural mortality rate at wk_t

Value

Returns nothing, is objective to be solved by `find_f`

Examples

```
## No examples
```

create_fields	<i>Create species distribution fields</i>
---------------	---

Description

`create_fields` parametrises and returns the spatio-temporal fields used for the spatial distribution of fish populations and movement in space and time for the simulations.

The spatio-temporal fields are generated using `spate.sim` function from the *spate* package using an advective-diffusion Stochastic Partial Differential Equation (SPDE). See *Lindgren 2011 and Sigrist 2015* for further detail.

Usage

```
create_fields(npt = 1000, t = 1, seed = 123, n.spp = NULL,
             spp.ctrl = NULL, plot.dist = FALSE, plot.file = getwd())
```

Arguments

npt	Numeric integer with the dimensions of the field in $npt * npt$
t	Numeric integer with the number of time-steps in the simulation
seed	(Optional) Numeric integer with the seed for the simulation
n.spp	Numeric integer with the number of species to be simulated. Each species must have an individual control list as detailed below.
spp.ctrl	List of controls to generate each species spatio-temporal distribution. Must be of the form <code>spp.ctrl = list(spp.1 = c(rho0 = 0.001, ...), spp.2 = c(rho0 = 0.001, ..),..)</code> and contain the following: <ul style="list-style-type: none"> • rho0 (≥ 0) Controls the range in a matern covariance structure. • sigma2 (≥ 0) Controls the marginal variance (i.e. process error) in the matern (≥ 0) covariance structure. • zeta (≥ 0) Damping parameter; regulates the temporal correlation. • rho1 (≥ 0) Range parameter for the diffusion process • gamma (≥ 0) Controls the level of anisotropy • alpha ($[0, \pi/2]$) Controls the direction of anisotropy

	<ul style="list-style-type: none"> • muX ($[-0.5, 0.5]$) x component of drift effect • muY ($[-0.5, 0.5]$) y component of drift effect • tau2 (≥ 0) Nugget effect (measurement error) • nu Smoothness parameter for the matern covariance function
plot.dist	Boolean, whether to plot the distributions to file
plot.file	path to save the plots of the species distributions

Value

Silently returns a list of spatial distributions with first level of the list being the population (1 -> n.spp) and the second being time (1 -> t). If plot.dist = TRUE it produces an image of the spatial distributions at each time step for each of the populations saved to the working directory (unless specified otherwise in plot.file)

Examples

```
fields <- create_fields(n.spp = 1, t = 2,
  spp.ctrl = list(
    'spp.1' = c('rho0' = 0.1, 'sigma2' = 1, 'zeta' = 0.1,
      'rho1' = 0.01, 'gamma' = 0.3, 'alpha' = pi/4,
      'muX' = -0.05, 'muY' = -0.05, 'tau2' = 0, 'nu' = 1.5)),
  plot.dist = TRUE, plot.file = getwd())
```

delay_diff	<i>Delay-difference (weekly)</i>
------------	----------------------------------

Description

delay_difference implements a two-stage delay-difference model with a weekly time-step after *Dichmont 2003*. Given the starting biomass, overall mortality and recruitment it returns the biomass in wk+1.

Usage

```
delay_diff(K = 0.3, F = NULL, M = 0.2, wt = 1, wtm1 = 0.1, R = NULL,
  B = NULL, Bm1 = NULL, al = NULL, alm1 = NULL)
```

Arguments

K	is a Numeric vector describing growth @param F is the weekly. Note: K is transformed to rho with $\rho = \exp(-K)$ for the model. estimate of instantaneous fishing mortality (obtained elsewhere, via find_f and baranov_f functions.
M	is a Numeric vector of the instantaneous rate of natural mortality for the population
wt	is a Numeric vector of the weight of a fish when fully recruited
wtm1	is a Numeric vector of the weight of a fish before its recruited
R	is a Numeric vector of the annual recruitment for the population in numbers
B	is the biomass of the population during wk_t
Bm1	is a Numeric vector of the biomass of the population in the previous week wk_{t-1}
al	is a Numeric vector of the proportion of recruits to the fishery in wk_t
alm1	is a Numeric vector of the proportion of recruits to the fishery in wk_{t-1}

Value

Returns the biomass at the beginning of the following week, wk_{t+1}

Examples

```
delay_diff(K = 0.3, F = 0.2, M = 0.2, wt = 1, wtm1 = 0.1, R = 1e6, B = 1e5,
Bm1 = 1e4, al = 0.5, alm1 = 0.1)
```

find_f	<i>find F (fishing mortality)</i>
--------	-----------------------------------

Description

find_f uses [uniroot](#) to find the fishing mortality rate given the catch, biomass and natural mortality using the [baranov_f](#) objective function.

Usage

```
find_f(C = C, B = B, M = M, FUN = baranov_f)
```

Arguments

C	is a Numeric vector detailing the catch at wk_t
B	is a Numeric vector of the biomass at wk_t
M	is a Numeric vector of the natural mortality rate at wk_t
FUN	is the objective function, here the Baranov equation baranov_f

Value

Gives the fishing mortality estimate F

Examples

```
find_f(C = 3000, B = 12000, M = 0.2, FUN = baranov_f)
```

Index

baranov_f, [1](#), [3](#), [4](#)

create_fields, [2](#)

delay_diff, [3](#)

find_f, [1-3](#), [4](#)

spate.sim, [2](#)

uniroot, [4](#)