Package 'PSPManalysis'

January 19, 2023

Type Package

Title Analysis of Physiologically Structured Population Models

Version 0.3.9

Description Performs demographic, bifurcation and evolutionary analysis of physiologically structured population models, which is a class of models that consistently translates continuous-time models of individual life history to the population level.
A model of individual life history has to be implemented specifying the individual-level functions that determine the life history, such as development and mortality rates and fecundity.
M.A. Kirkilionis, O. Diekmann, B. Lisser, M. Nool, B. Sommeijer & A.M. de Roos (2001) <doi:10.1142/S0218202501001264>.
O.Diekmann, M.Gyllenberg & J.A.J.Metz (2003) <doi:10.1016/S0040-5809(02)00058-8>.
A.M. de Roos (2008) <doi:10.1111/j.1461-0248.2007.01121.x>.

License GPL-3

Encoding UTF-8

NeedsCompilation yes

Imports rstudioapi (>= 0.11), pkgbuild (>= 1.1)

Suggests testthat, R.rsp

RoxygenNote 7.2.3

VignetteBuilder R.rsp

Depends R (>= 3.1)

Author Andre M. de Roos [aut, cre],
Ernst Hairer [ctb],
Gerhard Wanner [ctb]

Maintainer Andre M. de Roos <A.M.deRoos@uva.nl>

Repository CRAN

Date/Publication 2023-01-19 15:50:06 UTC

R topics documented:

`csbread` *Reads the state of a structured population model from a CSB file*

Description

`csbread` reads a CSB file, which is generated by `PSPMdemo`, `PSPMequi`, `PSPMecodyn` and `PSPMevodyn` to save the entire state of the environmental variables and physiologically structured populations during computations.

Usage

```
csbread(csbfile = NULL, state = -1)
```

Arguments

`csbfile`	(string, required)
	Name of the CSB file to be read with or without '.csb' extension.
`state`	(integer, optional)
	If not specified `csbread` will list the states that are stored in the CSB file. If specified, it should be the name or index of one of the states in the CSB file.

Details

output <- csbread(csbfile = NULL, state = -1)

Value

If a specific state is specified and found in the file, the state is returned as a list.

Examples

```
## Not run:
PSPMdemo("Medfly", c(2, 11, 0.1, 11, 15), clean = TRUE)
csbread("Medfly-PGR-0000")

csbread("Medfly-PGR-0000", 1)

## End(Not run)
```

PSPMclean *Deletes on request all files produced by the PSPManalysis package.*

Description

PSPMclean deletes all PSPManalysis result files (default) and/or all executables (hit 'F') in the current directory.

Usage

```
PSPMclean(str = NULL)
```

Arguments

str	Character (optional). Only valid argument is 'F'. If not or wrongly specified the user will be asked whether to do a full clean up or whether to quit the clean up.

Value

None.

Examples

```
## Not run:
PSPMclean()

PSPMclean("F")

## End(Not run)
```

PSPMdemo *Demographic analysis of a structured population model*

Description

PSPMdemo computes the population growth rate of a physiologically structured population model and its sensitivities with respect to all model parameters. PSPMdemo either carries out these computation for a single parameter set or varies one of the parameters over a range of values specified by the user

Usage

```
PSPMdemo(
  modelname = NULL,
  curvepars = NULL,
  parameters = NULL,
  options = NULL,
  clean = FALSE,
  force = FALSE,
  debug = FALSE,
  silent = FALSE
)
```

Arguments

modelname	(string, required)
	Basename of the file with model specification. The file should have extension ".h". For example, the model "Medfly" is specified in the file "Medfly.h". If the model is specified in R include the .R extension explicitly, i.e. specify the model name as "Medfly.R"
curvepars	(row vector, optional, can be left equal to its default NULL)
	Vector of length 5, specifying:
	curvepars[1]: the index of the parameter to vary (in case the model is specified in R, this can be a string with the name of the parameter as specified in the variable 'DefaultParameters')
	curvepars[2]: the initial value of the parameter
	curvepars[3]: the step size in the parameter value

curvepars[4]: lower threshold, below which value of the parameter the computation stops

curvepars[5]: upper threshold, above which value of the parameter the computation stops

parameters (row vector, optional, can be left equal to its default NULL)

Vector of length PARAMETER_NR (set in the model program file; This is the length of the variable 'DefaultParameters' if the model is specified in R), specifying the values for the model parameters to use in the computation. Vectors of other lengths, including an empty vector will be ignored.

options (row vector of strings, optional, can be left equal to its default NULL)

Vector with pairs of strings, consisting of an option name and a value (for example c("isort", "1")) or single options (i.e. c("test")). Possible option names and their values are:

"isort", "<index>": Index of i-state variable to use as ruling variable for sorting the structured populations

"report", "<value>": Interval between consecutive output of computed points to the console (>= 1). Minimum value of 1 implies output of every point

"test": Perform only a single integration over the life history, reporting dynamics of survival, R0, i-state and interaction variables

clean (Boolean, optional argument)

Specify clean = TRUE as argument to remove all the result files of the model before the computation

force (Boolean, optional argument)

Specify force = TRUE as argument to force a rebuilding of the model before the computation

debug (Boolean, optional argument)

Specify debug = TRUE as argument to compile the model in verbose mode and with debugging flag set

`silent` (Boolean, optional argument)

Specify silent = TRUE as argument to suppress reporting of compilation commands and results on the console

Details

output <- PSPMdemo(modelname = NULL, curvepars = NULL, parameters = NULL, options = NULL, clean = FALSE, force = FALSE, debug = FALSE, silent = FALSE)

Value

The output is a list containing the following elements:

`curvepoints`: Matrix with output for all computed points along the curve

`curvedesc`: Column vector with strings, summarizing the numerical details of the computed curve (i.e., initial point, parameter values, numerical settings used).

Examples

```
## Not run:
PSPMdemo("Medfly", c(2, 11, 0.1, 11, 16))

## End(Not run)
```

PSPMecodyn *Ecological dynamics of a structured population model computed using the Escalator Boxcar Train*

Description

`PSPMecodyn` computes the dynamics of a physiologically structured population model starting from an environmental and population state that is computed with `PSPMequi`. If starting from an arbitrary state is required, the list specifying the initial state should have the same layout as produced by `PSPMequi`.

Usage

```
PSPMecodyn(
  modelname = NULL,
  startstate = NULL,
  timepars = NULL,
  bifpars = NULL,
  parameters = NULL,
  options = NULL,
  clean = FALSE,
  force = FALSE,
  debug = FALSE,
  silent = FALSE
)
```

Arguments

`modelname` (string, required)

Basename of the file with the model specification. The file should have an extension ".h". For example, the model "PNAS2002" is specified in the file "PNAS2002.h". If the model is specified in R include the .R extension explicitly, i.e. specify the model name as "PNAS2002.R"

`startstate` (list, required)

The initial environmental and population state from which to start the simulation of the dynamics. This list should have the identical layout as a list returned by the function csbread(). As a minimum, the list should contain a vector 'Environment' specifying the initial values of the environmental variables, and a matrix 'Pop00' (assuming there is only a single population in the model), which specifies on each row the number and individual state variables of a cohort of while the different rows specify all the cohorts in the population.

`timepars` (row vector of length 4, required)

Vector of length 4 specifying the settings for the time integration:

`timepars[1]`: Cohort cycle time, i.e. time interval between starts of new boundary cohorts

`timepars[2]`: Output time interval, i.e. time interval between data output to .out file

`timepars[3]`: State output interval, i.e. time interval between complete state output to .csb file

`timepars[4]`: Maximum integration time, i.e. maximum time value until which to continue the integration

`bifpars` (row vector of length 6, optional)

Vector of length 6 specifying the settings for the bifurcation settings. If not specified a normal time integration is carried out.

`bifpars[1]`: Index of the bifurcation parameter

`bifpars[2]`: Starting value of the bifurcation parameter

`bifpars[3]`: Step size in the bifurcation parameter

`bifpars[4]`: Final value of the bifurcation parameter

`bifpars[5]`: Period of producing data output during each bifurcation interval

`bifpars[6]`: Period of producing state output during each bifurcation interval

`parameters` (row vector, optional, can be left equal to its default NULL)

Vector of length PARAMETER_NR (set in the model program file; This is the length of the variable 'DefaultParameters' if the model is specified in R), specifying the values for the model parameters to use in the computation. Vectors of other lengths, including an empty vector will be ignored.

`options` (row vector of strings, optional, can be left equal to its default NULL)

Vector with pairs of strings, consisting of an option name and a value (for example c("info", "1")). Possible option names and their values are:

`"info", "<index>"`: Level of performance information on the DOPRI5 integrator written to .err file (1, 2, 3 or 4)

`"report", "<index>"`: Interval between reporting of data output to console (> 0)

`clean` (Boolean, optional argument)

Specify clean = TRUE as argument to remove all the result files of the model before the computation

`force` (Boolean, optional argument)

Specify force = TRUE as argument to force a rebuilding of the model before the computation

`debug` (Boolean, optional argument)

Specify debug = TRUE as argument to compile the model in verbose mode and with debugging flag set

`silent` (Boolean, optional argument)

Specify silent = TRUE as argument to suppress reporting of compilation commands and results on the console

Details

output <- PSPMecodyn(modelname = NULL, startstate = NULL, timepars = NULL, bifpars = NULL, parameters = NULL, options = NULL, clean = FALSE, force = FALSE, debug = FALSE, silent = FALSE)

Value

The output is a list containing the following elements:

`curvepoints`: Matrix with output for all computed points along the curve

`curvedesc`: Column vector with strings, summarizing the numerical details of the computed curve (i.e., initial point, parameter values, numerical settings used)

Examples

```
## Not run:
initstate <- list(Environment = c(1.561e-04, 1.270e-04, 4.008e-06),
                  Pop00 = matrix(c(0.001, 0, 7.0, 1.0E-5, 300, 111),
                  ncol = 3, byrow = TRUE))
PSPMecodyn("PNAS2002", initstate, c(1, 1, 10, 100))

## End(Not run)
```

PSPMequi *Bifurcation analysis of a structured population model*

Description

PSPMequi computes bifurcation curves for a physiologically structured population model as a function of one or two parameters and detects bifurcation points along these curves. When computing equilibrium curves of a physiologically structured population model as a function of one parameter, PSPMequi can detect transcritical bifurcation points (branching points, BP) of both the structured populations as well as environment variables (BPE), limit points (LP) in the equilibrium curve and evolutionary stationary points (ESS). The location of these bifurcation points can subsequently be computed as a function of second parameter. In addition PSPMequi can compute the pairwise invasion plot (PIP) as a function of the resident and a mutant value for one evolving parameter.

Usage

```
PSPMequi(
  modelname = NULL,
  biftype = NULL,
  startpoint = NULL,
  stepsize = NULL,
  parbnds = NULL,
  parameters = NULL,
  minvals = NULL,
  maxvals = NULL,
  options = NULL,
  clean = FALSE,
  force = FALSE,
  debug = FALSE,
  silent = FALSE
)
```

Arguments

modelname	(string, required)
	Basename of the file with the model specification. The file should have an extension ".h". For example, the model "PNAS2002" is specified in the file "PNAS2002.h". If the model is specified in R include the .R extension explicitly, i.e. specify the model name as "PNAS2002.R"
biftype	(string, required)
	Type of bifurcation to compute: "BP", "BPE", "EQ", "LP", "ESS" or "PIP"
startpoint	(row vector, required)

The initial point from which to start the continuation of the curve

`stepsize` (double value, required)

Value of the step size in the first bifurcation parameter

`parbnds` (row vector, required)

Vector of length 3 for EQ continuation, length 6 for BP, BPE, LP and PIP continuation and 3+4*N for ESS continuation. The first triplet specifies: Each triples specifies:

`parbnds[1]`: the index of the first bifurcation parameter (in case the model is specified in R, this can be a string with the name of the parameter as specified in the variable 'DefaultParameters')

`parbnds[2]`: lower threshold, below which value of the first bifurcation parameter the computation stops

`parbnds[3]`: upper threshold, above which value of the first bifurcation parameter the computation stops

In case of two-parameter bifurcations, the second triplet specifies:

`parbnds[4]`: the index of the second bifurcation parameter (in case the model is specified in R, this can be a string with the name of the parameter as specified in the variable 'DefaultParameters')

`parbnds[5]`: lower threshold, below which value of the second bifurcation parameter the computation stops

`parbnds[6]`: upper threshold, above which value of the second bifurcation parameter the computation stops

In case of ESS continuation, consecutive sets of 4 values specify:
`parbnds[4*n]`: the index of population that is impacted by the parameter at its ESS value (in case the model is specified in R, this can be a string with the name of the parameter as specified in the variable 'DefaultParameters') `parbnds[4*n+1]`: the index of the parameter at its ESS value `parbnds[4*n+2]`: lower threshold, below which value of this ESS parameter the computation stops `parbnds[4*n+3]`: upper threshold, above which value of this ESS parameter the computation stops

`parameters` (row vector, optional, can be left equal to its default NULL)

Vector of length PARAMETER_NR (set in the model program file; This is the length of the variable 'DefaultParameters' if the model is specified in R), specifying the values for the model parameters to use in the computation. Vectors of other lengths, including an empty vector will be ignored.

`minvals` (row vector, optional, can be left equal to its default NULL)

Vector of length (ENVIRON_DIM + POPULATION_NR), specifying minimum values for the environmental variables and the population birth rates, at which computations will stop. Vectors of other lengths, including an empty vector will be ignored.

`maxvals` (row vector, optional, can be left equal to its default NULL)

Vector of length (ENVIRON_DIM + POPULATION_NR), specifying maximum values for the environmental variables and the population birth rates, at which computations will stop. Vectors of other lengths, including an empty vector will be ignored.

`options` (row vector of strings, optional, can be left equal to its default NULL)

Vector with pairs of strings, consisting of an option name and a value (for example c("popBP", "1")) or single options (i.e. c("test")). Possible option names and their values are:

`"envBP", "<index>"`: Index of environment variable, of which to continue the transcritical bifurcation

`"popBP", "<index>"`: Index of structured population, of which to continue the transcritical bifurcation

`"popEVO", "<index>"`: Index of structured population, for which to compute the selection gradient or perform PIP continuation

`"parEVO", "<index>"`: Index of parameter, for which to compute the selection gradient

`"envZE", "<index>"`: Index of environment variable in trivial equilibrium (can be used multiple times)

"popZE", "<index>": Index of structured population in trivial equilibrium (can be used multiple times)

"isort", "<index>": Index of i-state variable to use as ruling variable for sorting the structured populations

"report", "<value>": Interval between consecutive output of computed points to the console (>= 1). Minimum value of 1 implies output of every point

"noBP": Do not check for branching points while computing equilibrium curves

"noLP": Do not check for limit points while computing equilibrium curves

"single": Only compute the first point of the solution curve, do not continue the curve

"test": Perform only a single integration over the life history, reporting dynamics of survival, R0, i-state and interaction variables

clean (Boolean, optional argument)

Specify clean = TRUE as argument to remove all the result files of the model before the computation

force (Boolean, optional argument)

Specify force = TRUE as argument to force a rebuilding of the model before the computation

debug (Boolean, optional argument)

Specify debug = TRUE as argument to compile the model in verbose mode and with debugging flag set

silent (Boolean, optional argument)

Specify silent = TRUE as argument to suppress reporting of compilation commands and results on the console

Details

output <- PSPMequi(modelname = NULL, biftype = NULL, startpoint = NULL, stepsize = NULL, parbnds = NULL, parameters = NULL, minvals = NULL, maxvals = NULL, options = NULL, clean = FALSE, force = FALSE, debug = FALSE, silent = FALSE)

Value

The output is a list containing the following elements:

`curvepoints`: Matrix with output for all computed points along the curve

`curvedesc`: Column vector with strings, summarizing the numerical details of the computed curve (i.e., initial point, parameter values, numerical settings used)

`bifpoints`: Matrix with the located bifurcation points along the curve

`biftypes`: Column vector of strings, containing a description of the type of bifurcation for each of the located bifurcation points

Examples

```
## Not run:
PSPMequi("Indet_growth", "EQ", c(1, 0.22, 0), -0.1, c(6, 0.8, 1.0),
         options = c("popEVO", "0"), silent = TRUE)

## End(Not run)
```

`PSPMevodyn` *Evolutionary dynamics for a structured population model computed following the canonical equation*

Description

`PSPMevodyn` computes the dynamics of a physiologically structured population model over evolutionary time for an arbitrary number of evolving parameters. The evolutionary trajectory of these evolving parameters is determined by the canonical equation of Adaptive Dynamics, which is solved using a simple Euler integration scheme.

Usage

```
PSPMevodyn(
  modelname = NULL,
  startpoint = NULL,
  curvepars = NULL,
  evopars = NULL,
  covars = NULL,
```

```
  parameters = NULL,
  options = NULL,
  clean = FALSE,
  force = FALSE,
  debug = FALSE,
  silent = FALSE
)
```

Arguments

`modelname`	(string, required)
	Basename of the file with the model specification. The file should have an extension ".h". For example, the model "PNAS2002" is specified in the file "PNAS2002.h". If the model is specified in R include the .R extension explicitly, i.e. specify the model name as "PNAS2002.R"
`startpoint`	(row vector, required)
	The initial point from which to start the simulation of the dynamics over evolutionary time, including the initial values of the evolving parameters
`curvepars`	(row vector of length 2, required)
	Vector of length 2 specifying:
	`curvepars[1]`: the maximum step size in evolutionary time during the integration of the canonical equation
	`curvepars[2]`: the maximum evolutionary time at which to stop the integration of the canonical equation
`evopars`	(row vector of length n*4, required)
	Vector of length n*4 specifying:
	`evopars[1]`: the index of the structured population whose life history is influenced by the first evolving parameter
	`evopars[2]`: the index of the first evolution parameter (in case the model is specified in R, this can be a string with the name of the parameter as specified in the variable 'DefaultParameters')

`evopars[3]`: lower threshold, below which value of the first evolution parameter the computation stops

`evopars[4]`: upper threshold, above which value of the first evolution parameter the computation stops

.....

`evopars[n*4-3]`: the index of the structured population whose life history is influenced by the last evolving parameter

`evopars[n*4-2]`: the index of the last evolution parameter (in case the model is specified in R, this can be a string with the name of the parameter as specified in the variable 'DefaultParameters')

`evopars[n*4-1]`: lower threshold, below which value of the last evolution parameter the computation stops

`evopars[n*4]`: upper threshold, above which value of the last evolution parameter the computation stops

`covars` (row vector or matrix, optional, can be left equal to its default NULL)

Vector of length N*N or NxN matrix, where N is the number of evolving parameters. The vector or matrix elements specify the values of the covariance matrix in the selection gradients. Vectors of other lengths, including an empty vector will be ignored.

`parameters` (row vector, optional, can be left equal to its default NULL)

Vector of length PARAMETER_NR (set in the model program file), specifying the values for the model parameters to use in the computation. Vectors of other lengths, including an empty vector will be ignored.

`options` (row vector of strings, optional, can be left equal to its default NULL)

Vector with pairs of strings, consisting of an option name and a value (for example c("popZE", "1")) or single options (i.e. c("test")). Possible option names and their values are:

`"envZE", "<index>"`: Index of environment variable in trivial equilibrium (can be used multiple times)

"popZE", "<index>": Index of structured population in trivial equilibrium (can be used multiple times)

"isort", "<index>": Index of i-state variable to use as ruling variable for sorting the structured populations

"report", "<value>": Interval between consecutive output of computed points to the console (>= 1). Minimum value of 1 implies output of every point

"test": Perform only a single integration over the life history, reporting dynamics of survival, R0, i-state and interaction variables

clean (Boolean, optional argument)

Specify clean = TRUE as argument to remove all the result files of the model before the computation

force (Boolean, optional argument)

Specify force = TRUE as argument to force a rebuilding of the model before the computation

debug (Boolean, optional argument)

Specify debug = TRUE as argument to compile the model in verbose mode and with debugging flag set

silent (Boolean, optional argument)

Specify silent = TRUE as argument to suppress reporting of compilation commands and results on the console

Details

output <- PSPMevodyn(modelname = NULL, startpoint = NULL, curvepars = NULL, evopars = NULL, covars = NULL, parameters = NULL, options = NULL, clean = FALSE, force = FALSE, debug = FALSE, silent = FALSE)

Value

The output is a list containing the following elements:

curvepoints: Matrix with output for all computed points along the curve

curvedesc: Column vector with strings, summarizing the numerical details of the computed curve (i.e., initial point, parameter values, numerical settings used)

Examples

```
## Not run:
PSPMevodyn("Indet_growth", c(0.22, 0.03554, 1.0), c(0.05, 1),
           c(0, 6, 0.5, 1.5))

## End(Not run)
```

PSPMhelp *Opens the PSPManalysis manual*

Description

PSPMhelp opens the manual of the the PSPManalysis package in html format.

Usage

```
PSPMhelp()
```

Details

The manual is created in bookdown format. A PDF version can be downloaded via the PDF icon in the menu bar.

Value

None.

Examples

```
## Not run:
PSPMhelp()

## End(Not run)
```

PSPMind *Computes the individual life history of a physiologically structured population model in a given environment*

Description

PSPMind is a utility function to compute the individual life history as defined by a physiologically structured population model, given a specific set of values for the environmental variables

Usage

```
PSPMind(
  modelname = NULL,
  environment = NULL,
  parameters = NULL,
  options = NULL,
  clean = FALSE,
  force = FALSE,
  debug = FALSE,
  silent = FALSE
)
```

Arguments

modelname	(string, required)
	Basename of the file with the model specification. The file should have an extension ".h". For example, the model "PNAS2002" is specified in the file "PNAS2002.h". If the model is specified in R include the .R extension explicitly, i.e. specify the model name as "PNAS2002.R"
environment	(row vector, required)
	Vector of length ENVIRON_DIM (set in the model program file; This is the length of the variable 'EnvironmentState' if the model is specified in R), specifying the value of the environmental variables at which to calculate the individual life history. This vector can also be extended with values of the birth rates for all structured populations in the model, which would scale the output of the model with these birth rates.
parameters	(row vector, optional, can be left equal to its default NULL)
	Vector of length PARAMETER_NR (set in the model program file; This is the length of the variable 'DefaultParameters' if the model is specified in R), specifying the values for the model parameters to use in the computation. Vectors of other lengths, including an empty vector will be ignored.

`options` (row vector of strings, optional, can be left equal to its default NULL)

Vector with a pair of strings, consisting of an option name and a value (for example c("isort", "1")). The only possible option name and its values is:

`"isort"`, `"<index>"`: Index of i-state variable to use as ruling variable for sorting the structured populations

`clean` (Boolean, optional argument)

Specify clean = TRUE as argument to remove all the result files of the model before the computation

`force` (Boolean, optional argument)

Specify force = TRUE as argument to force a rebuilding of the model before the computation

`debug` (Boolean, optional argument) Specify debug = TRUE as argument to compile the model in verbose mode and with debugging flag set

`silent` (Boolean, optional argument)

Specify silent = TRUE as argument to suppress reporting of compilation commands and results on the console

Details

output <- PSPMind(modelname = NULL, environment = NULL, parameters = NULL, options = NULL, clean = FALSE, force = FALSE, debug = FALSE, silent = FALSE)

Value

The output is a structure with the population state as normally stored in the .csb output file of PSPMdemo, PSPMequi, PSPMecodyn and PSPMevodyn.

Examples

```
## Not run:
PSPMind("PNAS2002_5bs", c(1.30341E-05, 3.84655E-05, 4.00802E-06),
        options = c("isort", "1"), clean=TRUE, force=TRUE)

## End(Not run)
```

showpspm *Shows the model definition file of one of the example models provided with PSPManalysis*

Description

showpspm displays the file contents of one of the physiologically structured population models that is provided as an example.

Usage

```
showpspm(modelname = NULL)
```

Arguments

modelname	(string)
	Name of the example model to be displayed.

Details

showpspm(modelname = NULL)

Examples

```
## Not run:
showpspm("Medfly.R")

## End(Not run)
```

Index