

Package ‘streambugs’

November 29, 2023

Type Package

Title Parametric Ordinary Differential Equations Model of Growth, Death, and Respiration of Macroinvertebrate and Algae Taxa

Version 1.4

Date 2023-11-29

Author Nele Schuwirth, Peter Reichert, Mikolaj Rybinski

Maintainer Nele Schuwirth <nele.schuwirth@eawag.ch>

Description Numerically solve and plot solutions of a parametric ordinary differential equations model of growth, death, and respiration of macroinvertebrate and algae taxa dependent on pre-defined environmental factors. The model (version 1.0) is introduced in Schuwirth, N. and Reichert, P., (2013) <DOI:10.1890/12-0591.1>. This package includes model extensions and the core functions introduced and used in Schuwirth, N. et al. (2016) <DOI:10.1111/1365-2435.12605>, Kattwinkel, M. et al. (2016) <DOI:10.1021/acs.est.5b04068>, Mondy, C. P., and Schuwirth, N. (2017) <DOI:10.1002/eap.1530>, and Paillex, A. et al. (2017) <DOI:10.1111/fwb.12927>.

URL <https://www.eawag.ch/en/departement/siam/projects/streambugs/>

BugReports <https://gitlab.com/NeleSchuwirth/streambugs/-/issues>

License GPL-3

Depends R (>= 3.5.0)

Imports deSolve (>= 1.20)

Suggests testthat

RoxygenNote 7.2.3

Encoding UTF-8

NeedsCompilation yes

SystemRequirements C99

Repository CRAN

Date/Publication 2023-11-29 16:40:02 UTC

R topics documented:

| | |
|---|-----------|
| streambugs-package | 2 |
| construct.statevariables | 2 |
| count.feeding.links | 3 |
| decode.statevarnames | 4 |
| foodweb.plot | 5 |
| plot.streambugs | 6 |
| run.streambugs | 7 |
| streambugs.example.model.extended | 9 |
| streambugs.example.model.toy | 9 |
| streambugs.get.sys.def | 10 |
| streambugs.write.sys.def | 11 |
| Index | 12 |

| | |
|--------------------|---|
| streambugs-package | <i>Simulator of development of macroinvertebrate and algae taxa</i> |
|--------------------|---|

Description

The main function of **streambugs** is `run.streambugs`. Remaining functions have auxiliary purpose to facilitate model creation, writing, and plotting of the simulation results, or to provide ready to use examples.

See Also

Useful links:

- <https://www.eawag.ch/en/departement/siam/projects/streambugs/>
- Report bugs at <https://gitlab.com/NeleSchuwirth/streambugs/-/issues>

| | |
|--------------------------|--|
| construct.statevariables | <i>Construct the streambugs ODE state variable names</i> |
|--------------------------|--|

Description

Construct encoded labels of streambugs ODE state variable names from reach names, habitat names, and "taxa" names for at least one of the POM, Algae, or Invertebrates groups.

Usage

```
construct.statevariables(
  Reaches,
  Habitats,
  POM = NULL,
  Algae = NULL,
  Invertebrates = NULL
)
```

Arguments

| | |
|---------------|--|
| Reaches | reach names character vector; duplicates are dropped |
| Habitats | habitats names character vector; duplicates are dropped |
| POM | optional ("taxa") character vector of POM (particulate organic matter) group; duplicates are dropped (note: at least one "taxon" name out of all groups has to be given) |
| Algae | optional taxa character vector of Algae group; duplicates are dropped (note: at least one taxon name out of all groups has to be given) |
| Invertebrates | optional taxa character vector of Invertebrates group; duplicates are dropped (note: at least one taxon name out of all groups has to be given) |

Value

vector with state variable names in the form of "Reach_Habitat_Taxon_Group"

Examples

```
Reaches      <- paste0("Reach",1:2)
Habitats     <- paste0("Hab",1:1)
y.names <- construct.statevariables(Reaches,Habitats,Invertebrates=c("Baetis","Ecdyonurus"))
```

count.feeding.links *Count feeding links between taxa in streambugs ODE.*

Description

Count number of global "Cons" stoichiometric interactions (feeding links) between streambugs ODE state variables (taxa) in all habitats.

Usage

```
count.feeding.links(y.names, par)
```

Arguments

`y.names` same as `y.names` in [run.streambugs](#)
`par` same as `par` in [run.streambugs](#)

Value

Integer number of feeding links, which is a number of feeding links in a single habitat times number of habitats and number of reaches.

Examples

```
m <- streambugs.example.model.toy()
count.feeding.links(m$y.names, m$par)

# feeding links count does not change w/ number of habitats (nor reaches)
m <- streambugs.example.model.toy(n.Habitats=10)
count.feeding.links(m$y.names, m$par)
```

`decode.statevarnames` *Decode the streambugs ODE state variable names*

Description

Extract reach names, habitat names, taxa names and optional group names from encoded labels of streambugs ODE state variable names.

Usage

```
decode.statevarnames(y.names)
```

Arguments

`y.names` vector with state variable names in the form of "Reach_Habitat_Taxon" or "Reach_Habitat_Taxon_Group"

Value

List with:

`$y.names` names of state variables (input argument)

`$y.reaches`, `$y.reaches`, `$y.habitats`, `$y.taxa`, **and** `$y.groups`: Names of, respectively, reaches, habitats, taxa, and of groups of each state variable

`$reaches`, `$habitats`, `$taxa`, `$groups`: Unique names of, respectively, reaches, habitats, taxa, and of groups of state variables

`$ind.fA`: Indices used for the areal fractions of each reach and habitat

Examples

```
y.names <- c("Reach1_Hab1_Baetis_Invertebrates", "Reach1_Hab1_Ecdyonurus_Invertebrates",
            "Reach2_Hab1_Baetis_Invertebrates", "Reach2_Hab1_Ecdyonurus_Invertebrates")
decode.statevarnames(y.names)
```

 foodweb.plot

Plot the streambugs "foodweb" graph.

Description

Plot the "foodweb" graph depicting interactions between ODE variables in a streambugs model.

Usage

```
foodweb.plot(
  y.names,
  par,
  file = NA,
  cex = 1,
  font = 1,
  title = "",
  lwd = 1,
  bg = colors()[1],
  lcol = colors()[555],
  ncrit = 8,
  lcrit = 20,
  survivals = NA,
  observed = NA,
  texts = TRUE,
  pointcol = FALSE,
  ...
)
```

Arguments

| | |
|---------|--|
| y.names | same as y.names in run.streambugs |
| par | same as par in run.streambugs |
| file | optional name of a PDF file to plot to |
| cex | same as cex in par , consumed by here by multiple text generating functions |
| font | same as font in par , consumed here by text as a font type for taxa names |
| title | optional title for the plot |
| lwd | same as lwd in par , consumed here by lines as a line width for the "food web" edges |
| bg | same as bg in par |

| | |
|-----------|--|
| lcol | same as col in <code>par</code> , consumed here by <code>lines</code> as a line color for the "food web" edges |
| ncrit | number of inverts in one line at which they are shifted up and down, alternating |
| lcrit | number of letters/characters of state names which are plotted at level 2 |
| survivals | vector with the entries "survived" or "extinct" for all state variables with names matching names of taxa state variables and "SusPOM" |
| observed | vector with the entries "never"/"notobserved", "observed", "sometimes", "always", or NA/"NA" for all state variables, with names matching names of taxa state variables and "SusPOM" |
| texts | if to plot as "food web" nodes as texts with taxa names; otherwise, plot points |
| pointcol | if to color text or point nodes using Eawag coloring scheme (eawagfarben); otherwise plot all in the same color |
| ... | additional arguments for the <code>pdf</code> graphics device, relevant only if file argument was given. |

Examples

```
model <- streambugs.example.model.toy()
foodweb.plot(model$y.names, model$par, cex=1.1, title="complete foodweb", ncrit=8,
             lcrit=7, lwd=2, bg="white", lcol="blue", font=2)
```

plot.streambugs *Plot the results of streambugs ODE run*

Description

Plot time series of all streambugs ODE state variables, for each reach, habitat and group, resulting from the `run.streambugs` function call.

Usage

```
## S3 method for class 'streambugs'
plot(x, y, inp = NA, ...)
```

Arguments

| | |
|-----|---|
| x | matrix with results derived by <code>run.streambugs</code> |
| y | same as par in <code>run.streambugs</code> |
| inp | same as inp in <code>run.streambugs</code> |
| ... | additional argument for the <code>plot</code> function call |

Examples

```
m <- streambugs.example.model.toy()
r <- run.streambugs(y.names=m$y.names, times=m$times, par=m$par, inp=m$inp, C=TRUE)
plot(x=r$res, y=m$par, inp=m$inp)
```

| | |
|-----------------|--------------------------------------|
| run.streamebugs | <i>Run the streamebugs ODE model</i> |
|-----------------|--------------------------------------|

Description

Numerically solve streamebugs ODE model (in either R or C version) for given parameters, inputs and time points, using the [ode](#) routine.

Usage

```
run.streamebugs(
  y.names,
  times,
  par,
  inp = NA,
  C = FALSE,
  file.def = NA,
  file.res = NA,
  file.add = NA,
  return.res.add = FALSE,
  tout.add = NA,
  verbose = TRUE,
  method = "lsoda",
  rtol = 1e-04,
  atol = 1e-04,
  ...
)
```

Arguments

| | |
|----------------|---|
| y.names | state variables names, either as a vector encoded in the form "Reach_Habitat_Taxon" or "Reach_Habitat_Taxon_Group", or a list as returned by decode.statevarnames function |
| times | vector with time points for which output is wanted; the first |
| par | vector with constant parameters and model inputs |
| inp | list with time-dependent parameters or model inputs with one list element for each parameter or input that includes a matrix where first column is the time and second the corresponding parameter or input value |
| C | identifier for C- or R-Version |
| file.def | file name for writing system definition |
| file.res | file name for results |
| file.add | file name for additional output (e.g. process rates) |
| return.res.add | returns res.add output additionally to res |
| tout.add | optional identifier for specific output times for the additional output, if NA all res.add is calculated for all times |

| | |
|---------|---|
| verbose | prints some outputs to console |
| method | method used by <code>ode</code> |
| rtol | argument of <code>ode</code> function defining relative error tolerance, either a scalar or an array of the same size as <code>y.names</code> |
| atol | argument of <code>ode</code> function defining absolute error, either a scalar or an array of the same size as <code>y.names</code> |
| ... | further arguments passed to <code>ode</code> |

Value

A list with:

`$res` matrix of class `streambugs` with up to as many rows as elements in `times` and as many columns as elements in `y.names`, plus an additional column for the time value. There will be a row for each element in `times` unless the FORTRAN routine "lsoda" returns with an unrecoverable error.

`$res.add` optional additional output matrix with process rates and taxon specific factors, present only if `return.res.add` input parameter is set to `TRUE`.

Model syntax

See "docs/Streambugs_syntax.pdf" file in the package installation folder: `system.file("docs", "Streambugs_syntax.pdf", package="streambugs")`.

Examples

```
m <- streambugs.example.model.toy()
# Display inputs: list of perturbed variables with time points and new values
m$inp
# Simulate
res.C.default <- run.streambugs(y.names = m$y.names, times = m$times,
  par = m$par, inp = m$inp, C = TRUE)
# Modify input (halve second perturbation size) and re-simulate
m$inp$Reach3_w[2,2] <- m$inp$Reach3_w[2,2] / 2
m$inp
res.C.modified <- run.streambugs(y.names = m$y.names, times = m$times,
  par = m$par, inp = m$inp, C = TRUE)
# Compare exemplary trajectory of organic matter in one of the habitats
var.name <- "Reach3_Hab1_POM1_POM"
plot(m$times, res.C.default$res[, var.name], type="l", col="red")
lines(m$times, res.C.modified$res[, var.name], col="green")
```

`streambugs.example.model.extended`*Set-up the streambugs extended model*

Description

Set-up state variables, parameters, input, and output times of the streambugs extended model. All these are defined and read from .dat files The model is ready to run with [run.streambugs](#).

Usage

```
streambugs.example.model.extended()
```

Value

List with:

`$name` name of the example

`$y.names` list with names of state variables as returned by the [decode.statevarnames](#) function

`$times, $par, $inp`: corresponding input parameters of the [run.streambugs](#) function

Model syntax

See "docs/Streambugs_syntax.pdf" file in the package installation folder: `system.file("docs", "Streambugs_syntax.pdf", package="streambugs")`.

Examples

```
model <- streambugs.example.model.extended()
# display values of microhabitat tolerance values for Lumbriculidae taxa
model$par[grepl("^Lumbriculidae_microhabtolval", names(model$par))]
```

`streambugs.example.model.toy`*Set-up the streambugs toy model*

Description

Set-up state variables, parameters, input, and output times of the streambugs toy model. The model is ready to run with [run.streambugs](#).

Usage

```
streambugs.example.model.toy(n.Reaches = 3, n.Habitats = 2)
```

Arguments

| | |
|------------|---------------------------------------|
| n.Reaches | Number of reaches in the toy example |
| n.Habitats | Number of habitats in the toy example |

Value

List with:

\$name name of the example

\$y.names list with names of state variables as returned by the [decode.statevarnames](#) function

\$times, \$par, \$inp: corresponding input parameters of the [run.streambugs](#) function

Model syntax

See "docs/Streambugs_syntax.pdf" file in the package installation folder: `system.file("docs", "Streambugs_syntax.pdf", package="streambugs")`.

Examples

```
model <- streambugs.example.model.toy()
# display values of the exponent "q" in the food limitation term; Note:
model$par[grepl("(.*_)?q$", names(model$par))]
```

```
streambugs.get.sys.def
```

Get system definition of the streambugs ODE model

Description

Get a structured representation of the streambugs ODE system definition.

Usage

```
streambugs.get.sys.def(y.names, par, inp = NA)
```

Arguments

| | |
|---------|---|
| y.names | state variables names, either as a vector encoded in the form "Reach_Habitat_Taxon" or "Reach_Habitat_Taxon_Group", or a list as returned by decode.statevarnames function |
| par | vector with constant parameters and model inputs |
| inp | list with time-dependent parameters or model inputs with one list element for each parameter or input that includes a matrix where first column is the time and second the corresponding parameter or input value |

Value

List with definition of the model including input state variables, parameters, and inputs, as well as the derived model structure including global parameters, environmental conditions of reaches and habitats, initial conditions, taxa properties, stoichiometric coefficients of all processes, and process definitions for each state variable.

Examples

```
m <- streambugs.example.model.toy()
sys.def <- streambugs.get.sys.def(y.names=m$y.names, par=m$par, inp=m$inp)
# Get initial conditions for all state variables
sys.def$par.initcond$parvals
# Get stoichiometric coefficients of consumption within the food web
sys.def$par.stoich.web$Cons
# Get stoichiometric coefficients of death process for each taxon
# (transforming them into a dead organic matter "POM")
sys.def$par.stoich.taxon$Death
```

streambugs.write.sys.def

Write system definition of the streambugs ODE model

Description

Write system definition of the streambugs ODE model into a human-readable text file.

Usage

```
streambugs.write.sys.def(sys.def, file = NA)
```

Arguments

| | |
|---------|--|
| sys.def | system definition generated by the function streambugs.get.sys.def |
| file | file name |

Examples

```
m <- streambugs.example.model.toy()
sys.def <- streambugs.get.sys.def(y.names=m$y.names, par=m$par, inp=m$inp)
file.name <- tempfile(m$name, fileext=".dat")
streambugs.write.sys.def(sys.def, file.name)
file.show(file.name, delete.file=TRUE)
```

Index

`construct.statevariables`, 2
`count.feeding.links`, 3

`decode.statevarnames`, 4, 7, 9, 10

`foodweb.plot`, 5

`lines`, 5, 6

`ode`, 7, 8

`par`, 5, 6
`pdf`, 6
`plot`, 6
`plot.streambugs`, 6

`run.streambugs`, 2, 4–6, 7, 9, 10

`streambugs (streambugs-package)`, 2
`streambugs-package`, 2
`streambugs.example.model.extended`, 9
`streambugs.example.model.toy`, 9
`streambugs.get.sys.def`, 10, 11
`streambugs.write.sys.def`, 11

`text`, 5