Package: MCSimMod (via r-universe)

May 22, 2026

Title Working with 'MCSim' Models

Version 1.2

Description Tools that facilitate ordinary differential equation (ODE) modeling in 'R'. This package allows one to perform simulations for ODE models that are encoded in the GNU 'MCSim' model specification language (Bois, 2009) <doi:10.1093/bioinformatics/btp162> using ODE solvers from the 'R' package 'deSolve' (Soetaert et al., 2010) <doi:10.18637/jss.v033.i09>.

Imports deSolve, methods, tools

URL https://CRAN.R-project.org/package=MCSimMod, https://github.com/USEPA/MCSimMod

License GPL-3 + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/Needs/dev devtools, styler (== 1.11.0), testthat, covr

Config/Needs/website r-lib/pkgdown

Config/testthat/edition 3

VignetteBuilder knitr

Repository https://usepa.r-universe.dev

Date/Publication 2026-01-22 19:49:55 UTC

RemoteUrl https://github.com/usepa/mcsimmod

RemoteRef HEAD

RemoteSha 5e74f063b5184ef74e2020c734e7c3ab5e463708

Contents

`compileModel` *Function to translate and compile MCSim model specification text*

Description

This function translates MCSim model specification text to C and then compiles the resulting C file to create a dynamic link library (DLL) file (on Windows) or a shared object (SO) file (on Unix).

Usage

```
compileModel(
  model_file,
  c_file,
  dll_name,
  dll_file,
  source_file = model_file,
  hash_file = NULL,
  verbose_output = FALSE
)
```

Arguments

`model_file`	Name of an MCSim model specification file.
`c_file`	Name of a C source code file to be created by compiling the MCSim model specification file.
`dll_name`	Name of a DLL or SO file without the extension (".dll" or ".so").
`dll_file`	Name of the same DLL or SO file with the appropriate extension (".dll" or ".so").
`source_file`	Name of the original source file to use for hash calculation. Defaults to `model_file` for backward compatibility. When `writeTemp=TRUE` in `createModel()`, this should be set to the original source file path to ensure hash tracking works correctly when the source file is separate from the compiled model file.
`hash_file`	Name of a file containing a hash key for determining if `source_file` has changed since the previous translation and compilation.
`verbose_output`	Boolean specifying whether to write translator messages to standard output. If value is TRUE, messages will be written to standard output; if value is FALSE, messages will be written to files in a temporary directory.

Value

No return value. Creates files and saves them in locations specified by function arguments.

createModel *Function to create an MCSimMod Model object*

Description

This function creates a `Model` object using an MCSim model specification file or an MCSim model specification string.

Usage

```
createModel(
  mName = character(0),
  mString = character(0),
  writeTemp = TRUE,
  verboseOutput = FALSE
)
```

Arguments

`mName`	Name of an MCSim model specification file, excluding the file name extension `.model`.
`mString`	A character string containing MCSim model specification text.
`writeTemp`	Boolean specifying whether to write model files to a temporary directory. If value is TRUE (the default), model files will be Written to a temporary directory; if value is FALSE, model files will be Written to the same directory that contains the model specification file.
`verboseOutput`	Boolean specifying whether to write translator messages to standard output. If value is TRUE, messages will be written to standard output; if value is FALSE, messages will be written to files in a temporary directory.

Value

Model object.

Examples

```
## Not run:
# Simple model
mod <- createModel("path/to/model")

# Load/compile the model
mod$loadModel()

# Update parameters (P1 and P2)
mod$updateParms(c(P1 = 3, P2 = 1))

# Define times for ODE simulation
```

```
times <- seq(from = 0, to = 24, by = 0.1)

# Run the simulation
out <- mod$runModel(times)

## End(Not run)
```

`Model-class` *MCSimMod Model class*

Description

A class for managing MCSimMod models.

Arguments

`mName`	Name of an MCSim model specification file, excluding the file name extension `.model`.
`mString`	A character string containing MCSim model specification text.

Details

Instances of this class represent ordinary differential equation (ODE) models. A `Model` object has both attributes (i.e., things the object "knows" about itself) and methods (i.e., things the object can "do"). Model attributes include: the name of the model (`mName`); a vector of parameter names and values (`parms`); and a vector of initial conditions (`Y0`). Model methods include functions for: translating, compiling, and loading the model (`loadModel`); updating parameter values (`updateParms`); updating initial conditions (`updateY0`); and running model simulations (`runModel`). So, for example, if `mod` is a Model object, it will have an attribute called `parms` that can be accessed using the R expression `mod$parms`. Similarly, `mod` will have a method called `updateParms` that can be accessed using the R expression `mod$updateParms()`. Use the `createModel()` function to create `Model` objects.

Fields

`mName` Name of an MCSim model specification file, excluding the file name extension `.model`.

`mString` Character string containing MCSim model specification text.

`initParms` Function that initializes values of parameters defined for the associated MCSim model.

`initStates` Function that initializes values of state variables defined for teh associated MCSim model..

`Outputs` Names of output variables defined for the associated MCSim model.

`parms` Named vector of parameter values for the associated MCSim model.

`Y0` Named vector of initial conditions for the state variables of the associated MCSim model.

`paths` List of character strings that are names of files associated with the model.

`writeTemp` Boolean specifying whether to write model files to a temporary directory. If value is TRUE, model files will be written to a temporary directory; if value is FALSE, model files will be written to the same directory that contains the model specification file.

`verboseOutput` Boolean specifying whether to write translator messages to standard output. If value is TRUE, messages will be written to standard output; if value is FALSE, messages will be written to files in a temporary directory.

`recompiled` Boolean specifying if the model has been recompiled due to change in source file

Methods

`cleanup(deleteModel = FALSE)` Delete files created during the translation and compilation steps performed by `loadModel`. If `deleteModel = TRUE`, delete the MCSim model specification file, as well.

`initialize(...)` Initialize the Model object using an MCSim model specification file (mName) or an MCSim model specification string (mString).

`loadModel(force = FALSE)` Translate (if necessary) the model specification text to C, compile (if necessary) the resulting C file to create a dynamic link library (DLL) file (on Windows) or a shared object (SO) file (on Unix), and then load all essential information about the Model object into memory (for use in the current R session).

`runModel(times, ...)` Perform a simulation for the Model object using the `deSolve` function `ode` for the specified `times`.

`updateParms(new_parms = NULL)` Reset the values of the parameters for the Model object to their default values and then update the values of any parameters named in the argument 'new_parms' using values provided in that argument.

`updateY0(new_states = NULL)` Reset the values of the initial conditions of state variables for the Model object to their default values and then update the values of the initial conditions of any state variables named in the argument 'new_states' using values provided in that argument.

Index