The Journal of Open Source Software

MCSimMod: An R Package for Working with Ordinary Differential Equation Models Encoded in the MCSim Model Specification Language

Dustin F. Kapraun [1¶], **Todd J. Zurlinden** [1], **Ryan D. Friese** [2], and **Andrew J. Shapiro** [1]

1 U.S. Environmental Protection Agency, U.S.A. **2** Pacific Northwest National Laboratory, U.S.A. ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Many physical and biological phenomena can be described using mathematical models based on ordinary differential equations (ODEs). In such a model, an ODE describes how a "state variable" changes (quantitatively) with respect to an independent variable (e.g., time or position). In general, a model can include several state variables, each with its own ODE, so the model can be expressed as a system of ODEs. Thus, if y is a vector of n state variables, an ODE model that describes the state of the system at t (i.e., at a specific time or value of the independent variable) can be expressed as

$$\frac{\mathrm{d}}{\mathrm{d}t}y(t) = f(y(t), \theta, t), \tag{1}$$

where f is a vector-valued function (of dimension n) and θ is a vector of parameters (i.e., constants or variables other than the state variables and the independent variable).

In order to obtain a specific solution for a system of ODEs, one must know the initial state of the system,

$$y_0 = y(t_0), \tag{2}$$

where t_0 is the inital value of the independent variable. Equation Equation 2 is often described as a statement of the "initial conditions" of the system, and solving a system of ODEs subject to such initial conditions is called solving an initial value problem (IVP).

For the R programming language and environment (R Core Team, 2024), there are at least two packages available that facilitate solving of IVPs. The R package `deSolve` (Soetaert et al., 2010) can be used to solve IVPs for ODEs that have been encoded either in R or in a compiled language, such as C or Fortran. For models encoded in a compiled language, one can compile the model source code to generate machine code that typically executes much more quickly than R code, which must be "interpreted" anew each time it's executed on a computer. The `deSolve` package includes functions that provide interfaces to well-documented, public-domain IVP solution routines implemented in FORTRAN, including four ODE integrators from the package ODEPACK, and in C, including solvers of the Runge-Kutta family. The R package `mrgsolve` (Baron, 2024) also includes functions that can be used to solve IVPs. These `mrgsolve` functions provide interfaces to IVP solution routines implemented in C++. Despite their different implementations, the packages `deSolve` and `mrgsolve` use many of the same underlying IVP solution algorithms.

The Journal of Open Source Software

Statement of need

Physiologically based pharmacokinetic (PBPK) models, which are a class of ODE models that describe absorption, distribution, metabolism, and excretion of a substance by a biological organism, are frequently used to inform human health risk assessments for environmental chemicals and the development of pharmaceuticals. For many years, the programming language ACSL and the associated programming environment acslX were the tools of choice for many scientists and researchers that work with PBPK models, but in 2015, the company that maintained acslX announced that it would no longer support or sell the acslX software. Prior to the decline of acslX, some PBPK modelers used the free and open-source software tools R and MCSim MCSim (usually separately) to perform computational modeling work, but once acslX became unavailable, many PBPK modelers began using R and MCSim together to implement PBPK models. (See, for example, PBPK models published by Pearce et al. (2017); Bernstein et al. (2021); and Campbell et al. (2023).) R and MCSim each have benefits and drawbacks when it comes to working with ODE models. R is a flexible and popular programming language and environment for analyzing data and generating graphics. However, because R is in an interpreted language, R statements must be translated into machine instructions each time they are executed on a computer. Consequently, complex calculations (such as those associated with PBPK model simulations) encoded in R are generally performed relatively slowly. MCSim is a more specialized software tool designed for implementing and calibrating mathematical models – it is not a general purpose programming tool like R. However, MCSim takes advantage of compiled languages (as acslX did) to perform model simulations quickly, making it an appealing choice when one needs to perform many simulations, as is typically the case for Monte Carlo (MC) analyses and Markov chain Monte Carlo (MCMC) parameter estimation. One can leverage the strengths of both R and MCSim by defining an ODE model in the relatively simple MCSim model specification language, translating the model to C using an MCSim utility (called "mod"), compiling the C model to obtain machine code, and then performing model simulations rapidly and easily by writing R scripts that make use of the compiled code through the `deSolve` package. Unfortunately, installing R, MCSim, and other required software and ensuring that they work together properly can be challenging and has presented obstacles for many in the PBPK modeling community. We developed the R package `MCSimMod` as an easy-to install, user-friendly software tool that takes advantage of the flexibility of R and the computational speed of MCSim to meet the needs of PBPK modelers (especially those already familiar with MCSim), but MCSimMod can be used to solve any IVP and it is therefore a valuable resource for anyone seeking to work with ODE models in R.

Package description

Overview

The `MCSimMod` package facilitates ODE modeling within the R environment. `MCSimMod` allows one to solve IVPs for ODE models that have been described in the MCSim (Bois, 2009) model specification language using ODE integration functions from the R package `deSolve` (Soetaert et al., 2010). This system enables users to take advantage of the flexibility and post hoc data analysis capabilities of the interpreted language R while also achieving computational speeds typically associated with compiled programming languages like C and FORTRAN. Furthermore, this system encourages modelers to use separate files for defining models and executing model simulations, which can, in turn, improve modularity and reusability of source code developed for modeling analyses.

We designed `MCSimMod` using the object-oriented programming paradigm, in which computer programs are based on objects that comprise attributes (i.e., data) and methods (i.e., functionality). In particular, the MCSimMod package defines a class called `Model` that provides a template for model objects. One creates a `Model` object (i.e., an instance of the `Model` class) to represent a given ODE model. As illustrated in Figure 1, 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()`.

Model

Attributes

Name	Description
mName	Character string that names the model
parms	Named vector of model parameter values
Y0	Named vector of initial conditions
Outputs	List of strings that are names of output variables
paths	List of strings that are names of associated files

Methods

Name	Description
loadModel()	Translates and compiles the model if necessary; loads the model for use in the current R session
updateParms()	Updates parameter values
updateY0()	Updates initial conditions
runModel()	Performs a simulation with the model

Figure 1: MCSimMod `Model` object schema.

Mathematics

Single dollars ($) are required for inline mathematics e.g. $f(x) = e^{\pi/x}$

Double dollars make self-standing equations:

$$\Theta(x) = \begin{cases} 0 \text{ if } x < 0 \\ 1 \text{ else} \end{cases}$$

You can also use plain LaTeXfor equations

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(x) e^{i\omega x} dx \tag{3}$$

and refer to Equation 3 from text.

Acknowledgements

We acknowledge contributions from ...

References

Baron, K. T. (2024). *Mrgsolve: Simulate from ODE-based models*. https://CRAN.R-project.org/package=mrgsolve

Bernstein, A. S., Kapraun, D. F., & Schlosser, P. M. (2021). A model template approach for rapid evaluation and application of physiologically based pharmacokinetic models for use in human health risk assessments: A case study on per- and polyfluoroalkyl substances. *Toxicological Sciences*, *182*(2), 215–228. https://doi.org/10.1093/toxsci/kfab063

Bois, F. Y. (2009). GNU MCSim: Bayesian statistical inference for SBML-coded systems biology models. *Bioinformatics*, *25*, 1453–1454. https://doi.org/10.1093/bioinformatics/btp162

Campbell, J. L., Clewell, H. J., Van Landingham, C., Gentry, P. R., & Andersen, M. E. (2023). Using available in vitro metabolite identification and time course kinetics for beta-chloroprene and its metabolite, (1-chloroethenyl) oxirane, to include reactive oxidative metabolites and glutathione depletion in a PBPK model for beta-chloroprene. *Frontiers in Pharmacology*, *14*. https://doi.org/10.3389/fphar.2023.1223808

Pearce, R. G., Setzer, R. W., Strope, C. L., Sipes, N. S., & Wambaugh, J. F. (2017). Httk: R package for high-throughput toxicokinetics. *Journal of Statistical Software*, *79*(4), 1--26. https://doi.org/10.18637/jss.v079.i04

R Core Team. (2024). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Soetaert, K., Petzoldt, T., & Setzer, R. W. (2010). Solving differential equations in R: Package deSolve. *Journal of Statistical Software*, *33*(9), 1–25. https://doi.org/10.18637/jss.v033.i09