paropt

Konrad Krämer

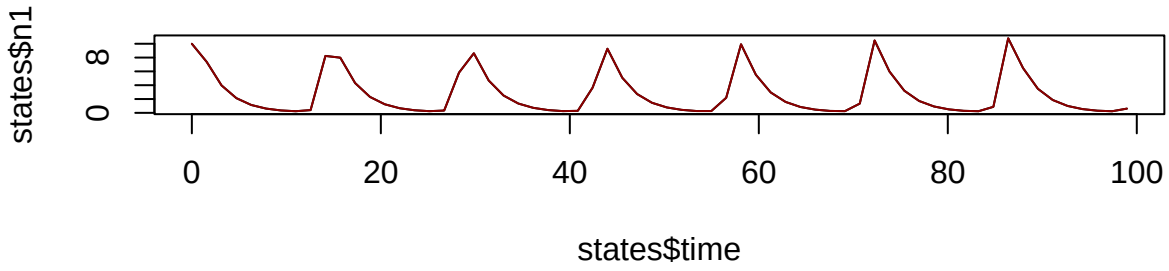
How to use paropt

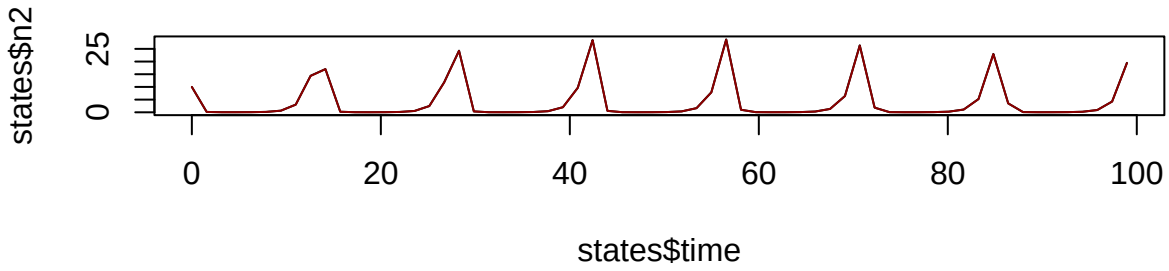
The package paropt is built in order to optimize parameters of ode-systems. The aim is to match the output of the **in silico** solution to previously measured states. The user has to supply an ode-system in the form of an R-function. The information about states and parameters are passed on as data.frames. In this vignette a predator-prey system is used as example to demonstrate how the functions of 'paropt' can be used.

```
# Optimize (all parameters are constant)
ode <- function(t, y, ydot, parameter) {
  a_db = at(parameter, 1)
  b_db = at(parameter, 2)
  c_db = at(parameter, 3)
  d_db = at(parameter, 4)
  predator_db = at(y,1)
  prey_db = at(y, 2)
  ydot[1] = predator_db*prey_db*c_db - predator_db*d_db
  ydot[2] = prey_db*a_db - prey_db*predator_db*b_db
}
path <- system.file("examples", package = "paropt")
states <- read.table(paste(path,"/states_LV.txt", sep = ""), header = TRUE)
lb <- data.frame(time = 0, a = 0.8, b = 0.3, c = 0.09, d = 0.09)
ub <- data.frame(time = 0, a = 1.3, b = 0.7, c = 0.4, d = 0.7)
set.seed(1)
res <- paropt::optimize(ode,
                        lb = lb, ub = ub,
                        reltol = 1e-06, abstol = c(1e-08, 1e-08),
                        error = 0.0001,
                        npop = 40, ngen = 1000,
                        states = states)

insilico <- res[[3]]
par(mfrow = c(2, 1))
plot(states$time, states$n1, type = "l")
points(insilico$time, insilico$n1, type = "l", col = "darkred")

plot(states$time, states$n2, type = "l")
points(insilico$time, insilico$n2, type = "l", col = "darkred")
```



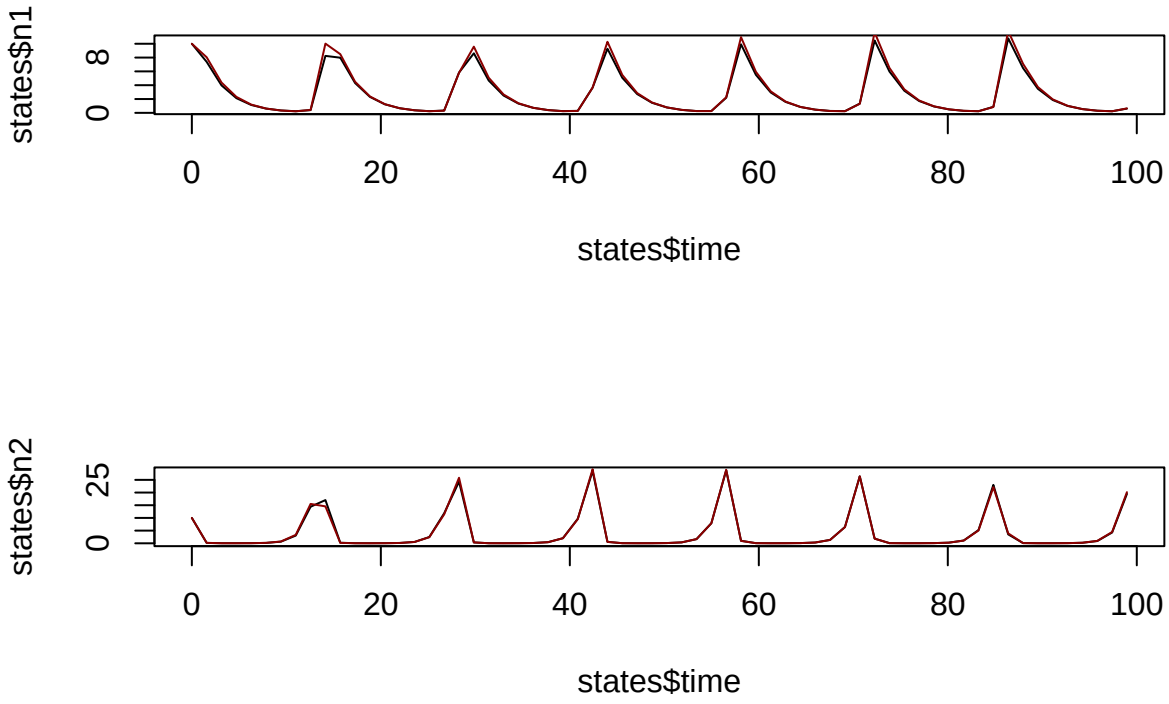



```r
# Optimize (parameter a,b and c are constant. d is variable!)
r <- function(a) {
  c(a, rep(NA, 3))
}

lb <- data.frame(time = c(0, 20, 60, 80),
                 a = r(0.8), b = r(0.3), c = r(0.09), d = 0.1)
ub <- data.frame(time = c(0, 20, 60, 80),
                 a = r(1.3), b = r(0.7), c = r(0.4), d = 0.6)
set.seed(1)
res <- paropt::optimize(ode,
                        lb = lb, ub = ub,
                        reltol = 1e-06, abstol = c(1e-08, 1e-08),
                        error = 0.0001,
                        npop = 40, ngen = 10000,
                        states = states)

insilico <- res[[3]]
par(mfrow = c(2, 1))
plot(states$time, states$n1, type = "l")
points(insilico$time, insilico$n1, type = "l", col = "darkred")

plot(states$time, states$n2, type = "l")
points(insilico$time, insilico$n2, type = "l", col = "darkred")
```



What happens during an evaluation of a parameterset

1. optimizer creates a bunch of random possible solutions within the parameter boundaries
2. Each solution is passed to the ode-solver which integrates along the time and returns the states at the time-points specified in the data.frame containing the state-information
3. For each parameter set an error is calculated
4. The velocity and subsequently the parameter-sets of the swarm are updated based on the errors

Particle swarm optimizer (PSO)

The PSO implementation used here has several key features. First of all, a bunch (number of particles defined by user) of possible parameter sets is created within the boundaries defined by the user. Each parameter set is called a particle. All particles together are called the swarm. Each possible parameter set is passed to the solver which integrates the system. The result is used to calculate the error. Thus, each particle has a current solution and a current personal best value. Furthermore, each particle possesses a neighborhood which consists of 0-3 other particles (for details see Akman, Akman, and Schaefer (2018)). The PSO uses a combination of its history (personal best value) and the information of the best particle of the neighborhood to change its current value. Notably, in this package a randomly adaptive topology is used. This means, that the neighborhood is recalculated each time when the global best solution (best solution of the entire swarm) has not improved within one generation.
Using this approach it is possible to optimize quite large parameter spaces. For instance, check the following publications Krämer, Brock, and Heyer (2022) and Krämer et al. (2022).

References

Akman, Devin, Olcay Akman, and Elsa Schaefer. 2018. "Parameter Estimation in Ordinary Differential Equations Modeling via Particle Swarm Optimization." *Journal of Applied Mathematics* 2018. https://doi.org/10.1155/2018/9160793.

Krämer, Konrad, Judith Brock, and Arnd G Heyer. 2022. "Interaction of Nitrate Assimilation and Photorespiration at Elevated CO2." *Frontiers in Plant Science* 13 (July). https://doi.org/10.3389/fpls.2022.897924.

Krämer, Konrad, Gabi Kepp, Judith Brock, Simon Stutz, and Arnd G. Heyer. 2022. "Acclimation to elevated CO2 affects the C/N balance by reducing de novo N-assimilation." *Physiologia Plantarum* 174 (1): 1–13. https://doi.org/10.1111/ppl.13615.