Using R for Systems Understanding

A Dynamic Approach

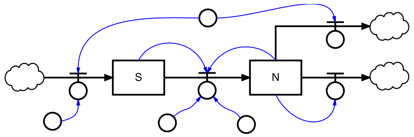
Thomas Petzoldt & Karline Soetaert

Technische Universität Dresden
Institute of Hydrobiology
Dresden, Germany
`thomas.petzoldt@tu-dresden.de`

Centre for Estuarine and Marine Ecology (CEME)
Netherlands Institute of Ecology (NIOO-KNAW)
Yerseke, The Netherlands
`k.soetaert@nioo.knaw.nl`

18th August 2011

Dynamic Systems

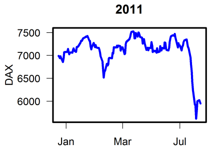


Evolution of systems in time (or / and space)

- Growth of organisms (and of my children),
- Economy, traffic, financial markets,
- Chemical reactions, spread of diseases,
- Movement of planets, stars, the universe.

Description

- Empirical with "pure statistics" (input-output, black box),
- Mechanistic (what's going on within the system):
  - Single objects (= agents, automata, individuals),
  - Populations and pools (⟶ differential equations).



Dynamic systems

- difficult to forecast in brain ⟶ weather, stock market
- non-linearity, indirect effects, feedback loops, oscillations,
- dampening or autocatalytic amplification?

⟶ stability, chaos, crash?

Modelling

- Systems understanding: most important processes,
- Simulate experiments before wasting time and money,
- Design experiments (and management)
  for best outcome, and improve statistical significance.

Differential equations in R: why and how

Why numerical solutions?

- Not all systems have an analytical solution,
- Numerical solutions allow discrete forcings, events, ...
- If standard tool for statistics, why additional software for dynamic simulations?

How in R?

- odesolve (Setzer, 2001):
  ⟶ two ODE solvers (lsoda, rk4),
- deSolve (Soetaert, Petzoldt, Setzer, 2009):
  ⟶ comprehensive set of solvers (ODE, DAE, PDE, DDE).
- Note: odesolve is deprecated, use deSolve!

Real systems need more than ODEs → additional features

Example problem	Type	In R?
algebraic constraints	DAE (diff. algebraic eq.)	(1)
time and space	PDE (partial diff. eq.)	(1,2,3)
time delays	DDE (delay diff. eq)	(1)
time dependent external control	forcing functions	(1)
abrupt changes of states (externally triggered)	events	(1)
abrupt changes of states (depending on state of the system)	roots + events	(1)
identify parameters	sensitivity, calibration	(4)

(1) deSolve – (2) rootSolve – (3) ReacTran – (4) FME

More additional features

Plotting is made easy with high-level plotting functions

- `plot`-, `image`- and `hist`- methods (S3)
- plotting multiple senarios simultaneously
- adding observed data
- "movie-like" output

Time-consuming models can be part R/part compiled code

- as fast as entire model in compiled code
- input - output handling as flexible as entire model in R

Case studies

Cultures and growth experiments

- physiological properties of organisms (e.g. growth rate),
- test of environmental factors (temperature, pH, salinity, toxicity),
- production of biomass, pharmaceuticals, beer, wine, whiskey . . .

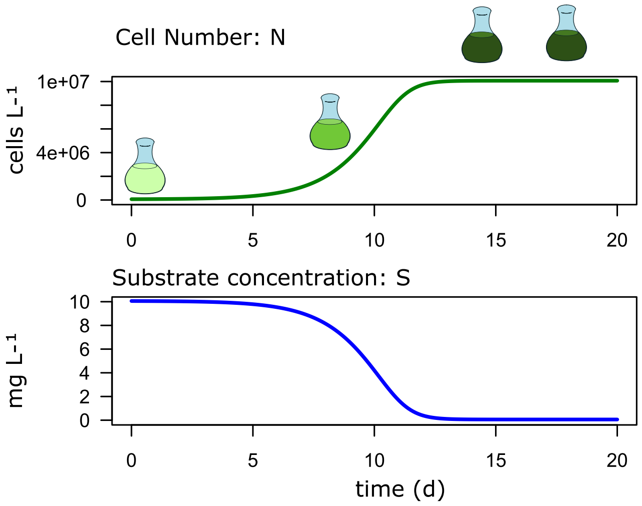
Experimentalists' questions to the modeller

- Determine optimal conditions for getting:
  - statistically significant effects in an experiment.
  - maximum yield of a product with minimum costs.
- Determine physiological parameters after the experiment.

Batch and chemostat cultures ...

- are very easy in R,
- but what with other reactors like "semi-batch"?

Substrate dependent growth in a batch



- cells grow until substrate (e.g. phosphorus) is exhausted.

Substrate limited growth model

Equations

$$f(S) = \frac{r \cdot S}{ks + S}$$

$$\frac{dS}{dt} = -\frac{1}{Y} \cdot f(S) \cdot N$$

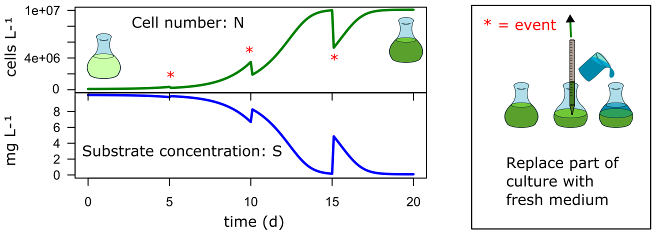
$$\frac{dN}{dt} = f(S) \cdot N$$

- initial values, parameters, time steps,
- numerical solution,
- visualization.

R Code

```
library(deSolve)
batch <- function(time, y, parms){
  with(as.list(c(y, parms)), {
    f  <- r * S / (ks + S)
    dS <- - 1/Y * f * N
    dN <-         f * N
    return(list(c(dS, dN)))
  })
}
y <- c(S = 10, N = 1e4)
parms <- c(r=1, ks=5, Y=1e6, S0=10)
times <- seq(0, 20, 0.1)
out <- ode(y, times, batch, parms)
plot(out)
```

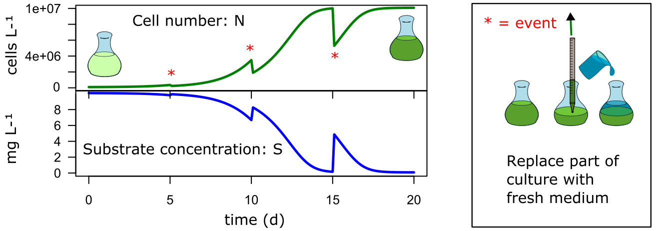
Semicontinuous culture (Semibatch I)



Discontinuous operation not trivial for ODE solvers

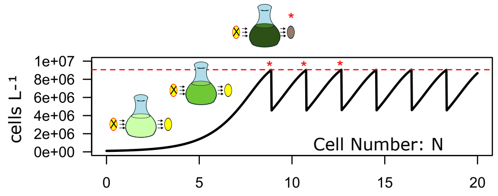
- Use very small time steps? → *inefficient*
- Use loops to glue separate solutions together?? → *programming*
- Good news: recent `deSolve` supports events!

Semicontinuous culture (Semibatch II)



```
etime <- seq(5, 20, 5)               # time points to triger events

eventfun <- function(t, y, parms) {              # event function
  with(as.list(c(y, parms)), {
    return(c(D * S0 + (1-D) * S, (1-D) * N))    # D = dilution rate
  })
}
out <- ode(y, times, batch, parms,
           events = list(func = eventfun, time = etime))
```

Semicontinuous culture III: Turbidostat mode



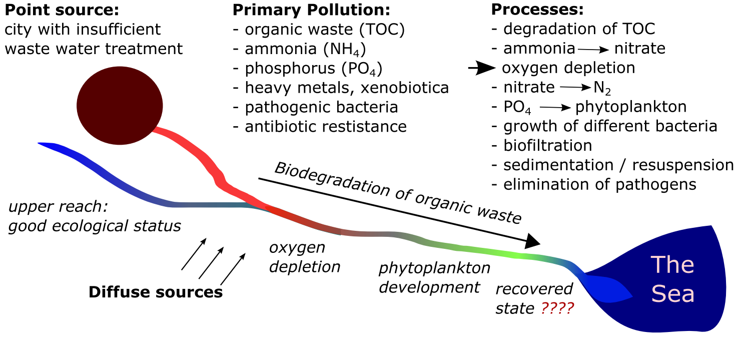
- Dilute culture when cell number N exceeds critical number (Detected with photometric or turbidity measurement).
- Use root-finding properties of the `deSolve` solvers.

```
crit <- 0.9e7   # critical cell number that triggers dilution event

rootfun <- function (t, y, pars) return(crit - y[2])

out <- ode(y, times, batch, parms,
  events = list(func = eventfun, root = TRUE), rootfun = rootfun)
```

Matter turnover and transport in a polluted river

- What are the main sources and effects of pollution?
- What can be done to improve water quality?

Matter turnover and transport in a polluted river



Point source:
city with insufficient
waste water treatment

Primary Pollution:
- organic waste (TOC)
- ammonia (NH_4)
- phosphorus (PO_4)
- heavy metals, xenobiotica
- pathogenic bacteria
- antibiotic restistance

Processes:
- degradation of TOC
- ammonia → nitrate
- oxygen depletion
- nitrate → N_2
- PO_4 → phytoplankton
- growth of different bacteria
- biofiltration
- sedimentation / resuspension
- elimination of pathogens

- Many processes in reality ...
- ... let's look at two processes for demonstation basic principles:
  1. oxygen consumtion by biological ammonia oxidation (nitrification)
  2. oxygen exchange between atmosphere and water (re-aeration)

Transport, Processes, Stoichiometry

$Y(x, n)$: State matrix

$T(x, n)$: Transport matrix

$P(x, k) = f(Y, c)$: Process matrix

$V(k, n)$: Stoichiometry matrix

with:

n: number of state variables (e.g. chemical species)
k: number of processes
x: space coordinate (here: river kilometers in 1D)
c: constants (model parameters in nonlinear functions)

Transport, Processes, Stoichiometry

$$\text{change} = \text{transport} + \text{processes} \cdot \text{stoichiometry}$$

$$Y' = T + P \cdot V$$

$$\begin{pmatrix} y'_{1,1} & \cdots & y'_{1,n} \\ y'_{2,1} & \cdots & y'_{2,n} \\ & \cdots\cdots & \\ y'_{x,1} & \cdots & y'_{x,n} \end{pmatrix} = \begin{pmatrix} t_{1,1} & \cdots & t_{1,n} \\ t_{2,1} & \cdots & t_{2,n} \\ & \cdots\cdots & \\ t_{x,1} & \cdots & t_{x,n} \end{pmatrix} + \begin{pmatrix} p_{1,1} & \cdots & p_{1,k} \\ p_{2,1} & \cdots & p_{2,k} \\ & \cdots\cdots & \\ p_{x,1} & \cdots & p_{x,k} \end{pmatrix} \cdot \begin{pmatrix} v_{1,1} & \cdots & v_{1,n} \\ v_{2,1} & \cdots & v_{2,n} \\ & \cdots\cdots & \\ v_{k,1} & \cdots & v_{k,n} \end{pmatrix}$$

Core elements of the river model

Transport (package ReacTran)

```
tran <- cbind(
  tran.1D(C = NH4, D = D, v = v, C.up = NH4up, C.down = NH4dwn, A = A, dx = Grid)$dC,
  tran.1D(C = NO3, D = D, v = v, C.up = NO3up, C.down = NO3dwn, A = A, dx = Grid)$dC,
  tran.1D(C = O2 , D = D, v = v, C.up = O2up,  C.down = O2dwn,  A = A, dx = Grid)$dC
)
```

Stoichiometry matrix

```
stoich <- matrix(c(
  # NH4  NO3  O2
    0,    0,   1,     # reaeration
   -1,   +1,  -4.57   # nitrification
  ),  nrow = 2, byrow = TRUE)
```

Process equations

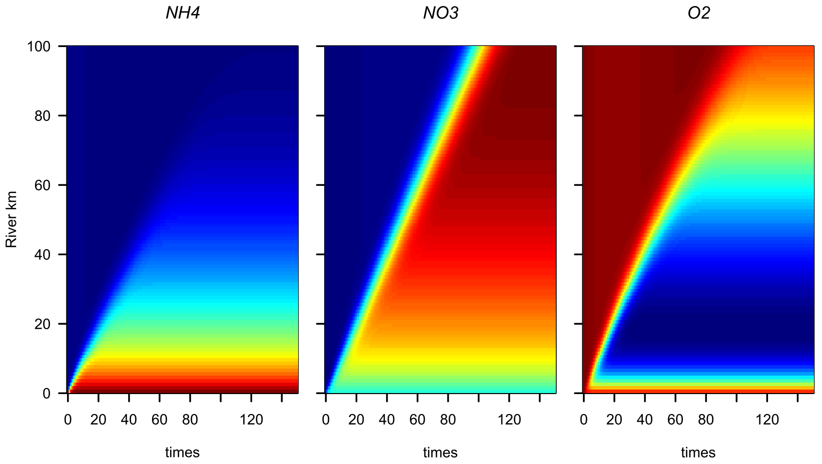
```
proc <- cbind(
  k2 * (O2sat - O2),               # re-aeration
  rMax * O2/(O2 + kO2) * NH4       # nitrification
)
```

State equation

```
dY   <- tran + proc %*% stoich
```

Outcome of the river model

```
image(out1D) # + some tuning to produce figure below
```



See also: `nitrification1D_ani.html`

Full scale case studies

At CEME (The Netherlands) , Roger Nzigou is using such a model to establish nutrient budgets for the Girone Estuary

At TU Dresden (Germany), Anna-Maria Ertel and Sabine Hacker use such models to analyse pollution turnover and bacteria transport in the Western Bug River (Ukraine)

Photo removed

River modellers in Europe

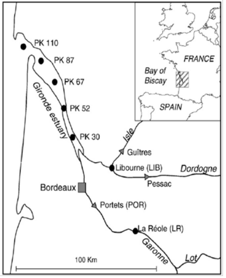
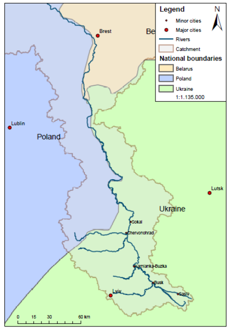


Photo removed

More about the Western Bug project, see:
Ertel et al. (2011) Env Earth Sci
DOI 10.1007/s12665-011-1289-0



The dynmod ecosystem – Model is solved, analysis begins . . .

- much can be done with R's standard and contributed packages,
- special packages for dynamic model analysis:

deSolve	now supports user-friendly plotting of results.
simecol	object oriented structuring of models and scenarios together with their data
simecolModels	a growing collection of models
FME	sensitivity analysis, parameter identification, confidence bands (MCMC)

- "knowledge base" packages:

marelac	datasets, constants, utilities for aquatic sciences
seacarb	seawater carbonate chemistry (Lavigne & Gattuso)
AquaEnv	integrated toolbox for aquatic chemical model generation
stoichcalc	handling of stoichiometric matrices (Reichert & Schuwirth, 2010)

- Sweave for report writing (Leisch, 2002).

Summary and Conclusions

R supplies a comprehensive ecosystem to the dynamic modeller

- powerful tools and many **prototypical examples**,
- efficient algorithms for more than only most common situations.
- comprehensive documentation: package docs, publications, books.

R's tools are now suited for both beginners and professional work.

Thank you!

More:

`http://desolve.r-forge.r-project.org`
(examples, PDFs, papers, books . . .).

Mailing list:

`mailto:r-sig-dynamic-models@r-project.org`
Special interest group for dynamic simulation models in R.

Acknowledgments

Citation

A lot of effort went in creating this software; please cite it when using it.

- to cite deSolve: [31], rootSolve [30], ReacTran [24]
- Some complex examples can be found in [27],
- A framework to fit differential equation models to data is FME [26],
- A framework for ecological modelling is simecol [14],
- . . . and don't forget the long history of original work referenced in the papers mentioned above, especially the original algorithms.

Acknowledgments

- None of this would be possible without the splendid work of the R Core Team [15],
- This presentation was created with Sweave [9],
- Creation of the packages made use of Rforge [32].

Bibliography I

[1] K. E. Brenan, S. L. Campbell, and L. R. Petzold.
Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations.
SIAM Classics in Applied Mathematics, 1996.

[2] P. N. Brown, G. D. Byrne, and A. C. Hindmarsh.
Vode, a variable-coefficient ode solver.
SIAM Journal on Scientific and Statistical Computing, 10:1038–1051, 1989.

[3] CWI.
Test set for initial value problem solvers, release 2.4, 2008.
`http://pitagora.dm.uniba.it/~testset/`.

[4] E. Hairer, S. P. Norsett, and G. Wanner.
Solving Ordinary Differential Equations I: Nonstiff Problems. Second Revised Edition.
Springer-Verlag, Heidelberg, 2009.

[5] E. Hairer and G. Wanner.
Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems. Second Revised Edition.
Springer-Verlag, Heidelberg, 2010.

[6] A. C. Hindmarsh.
ODEPACK, a systematized collection of ODE solvers.
In R. Stepleman, editor, *Scientific Computing, Vol. 1 of IMACS Transactions on Scientific Computation*, pages 55–64. IMACS / North-Holland, Amsterdam, 1983.

[7] W. Hundsdorfer and J. Verwer.
Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations. Springer Series in Computational Mathematics.
Springer-Verlag, Berlin, 2003.

[8] R. Lefever, G. Nicolis, and I. Prigogine.
On the occurrence of oscillations around the steady state in systems of chemical reactions far from equilibrium.
Journal of Chemical Physics, 47:1045–1047, 1967.

[9] F. Leisch.
Dynamic generation of statistical reports using literate data analysis.
In W. Härdle and B. Rönz, editors, *COMPSTAT 2002 – Proceedings in Computational Statistics*, pages 575–580, Heidelberg, 2002. Physica-Verlag.

Bibliography II

[10] E. Lorenz.
Deterministic non-periodic flows.
Journal of atmospheric sciences, 20:130–141, 1963.

[11] M. C. Mackey and L. Glass.
Oscillation and chaos in physiological control systems.
Science, 197:287–289, 1977.

[12] L. R. Petzold.
Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations.
SIAM Journal on Scientific and Statistical Computing, 4:136–148, 1983.

[13] T. Petzoldt.
R as a simulation platform in ecological modelling.
R News, 3(3):8–16, 2003.

[14] T. Petzoldt and K. Rinke.
simecol: An object-oriented framework for ecological modeling in R.
Journal of Statistical Software, 22(9):1–31, 2007.

[15] R Development Core Team.
R: A Language and Environment for Statistical Computing.
R Foundation for Statistical Computing, Vienna, Austria, 2011.
ISBN 3-900051-07-0.

[16] H. H. Robertson.
The solution of a set of reaction rate equations.
In J. Walsh, editor, *Numerical Analysis: An Introduction*, pages 178–182. Academic Press, London, 1966.

[17] O. Rossler.
An equation for continous chaos.
Physics Letters A, 57 (5):397–398, 1976.

[18] L. Shampine and S. Thompson.
Solving ddes in matlab.
App. Numer. Math., 37:441–458, 2001.

Bibliography III

[19] L. F. Shampine, I. Gladwell, and S. Thompson.
Solving ODEs with MATLAB.
Cambridge University Press, Cambridge, 2003.

[20] K. Soetaert.
rootSolve: Nonlinear root finding, equilibrium and steady-state analysis of ordinary differential equations, 2009.
R package version 1.6.

[21] K. Soetaert, J. R. Cash, and F. Mazzia.
bvpSolve: Solvers for Boundary Value Problems of Ordinary Differential Equations, 2010.
R package version 1.1.

[22] K. Soetaert and P. M. J. Herman.
A Practical Guide to Ecological Modelling. Using R as a Simulation Platform.
Springer-Verlag, New York, 2009.

[23] K. Soetaert and F. Meysman.
ReacTran: Reactive Transport Modelling in 1D, 2D and 3D, 2009.
R package version 1.1.

[24] K. Soetaert and F. Meysman.
Reactive transport in aquatic ecosystems: rapid model prototyping in the open source software R.
Environmental modelling and software, page in press, 2011.

[25] K. Soetaert and T. Petzoldt.
FME: A Flexible Modelling Environment for Inverse Modelling, Sensitivity, Identifiability, Monte Carlo Analysis, 2009.
R package version 1.0.

[26] K. Soetaert and T. Petzoldt.
Inverse modelling, sensitivity and monte carlo analysis in R using package FME.
Journal of Statistical Software, 33(3):1–28, 2010.

[27] K. Soetaert and T. Petzoldt.
Solving ODEs, DAEs, DDEs and PDEs in R.
Journal of Numerical Analysis, Industrial and Applied Mathematics, in press, 2011.

Bibliography IV

[28] K. Soetaert, T. Petzoldt, and R. Setzer.
R-package deSolve, Writing Code in Compiled Languages, 2009.
package vignette.

[29] K. Soetaert, T. Petzoldt, and R. W. Setzer.
deSolve: General solvers for initial value problems of ordinary differential equations (ODE), partial differential equations (PDE), differential algebraic equations (DAE), and delay differential equations (DDE), 2009.
R package version 1.7.

[30] K. Soetaert, T. Petzoldt, and R. W. Setzer.
Solving Differential Equations in R.
The R Journal, 2(2):5–15, December 2010.

[31] K. Soetaert, T. Petzoldt, and R. W. Setzer.
Solving differential equations in R: Package deSolve.
Journal of Statistical Software, 33(9):1–25, 2010.

[32] S. Theußl and A. Zeileis.
Collaborative Software Development Using R-Forge.
The R Journal, 1(1):9–14, May 2009.

[33] B. van der Pol and J. van der Mark.
Frequency demultiplication.
Nature, 120:363–364, 1927.