Using R for mathematical modelling (the environment).

Karline Soetaert
Natural systems are very complex

- Scientists want to understand this complexity and make quantitative predictions

- Model = simplifications of the complex natural environment

- Test model to data

- Quantification of unmeasured processes

- Budgetting, interpolation in time/space

- Prediction of future behavior
Mathematical model based on mass balance conservation

\[ \frac{dA}{dt} = -F_1 \]
\[ \frac{dB}{dt} = F_1 - F_2 \]

⇒ Differential equations
Use R to solve mathematical mass balance models

Three different types of models/solutions – three main packages

- Integration (deSolve)
- Steady-state solution (rootSolve)
- Least-squares solutions (limSolve)

What was available + what is new

Two examples

- HIV model (dynamic / steady-state)
- Deep-water coral food web
Example 1: hiv dynamics

Large interest in viral infection
- Human disease
- Marine animals, algae, bacteria are affected
  ⇒ Important role in biogeochemical cycles
\[
\frac{dH}{dt} = \lambda - \rho \cdot H - \beta \cdot H \cdot V \\
\frac{dI}{dt} = \beta \cdot H \cdot V - \delta \cdot I \\
\frac{dV}{dt} = n \cdot \delta \cdot I - c \cdot V - \beta \cdot H \cdot V
\]

\(t\) = time \\
n, c, ... = parameter \\
H, I, V = state variable
Model formulation:

**Derivative**

\[
\frac{dC}{dt} = f(\Theta, C, t, u)
\]

**Initial condition**

\[C_{t=0} = C0\]

Model solution:

**Integration**

\[C(t) \quad \text{For } t > t_0\]
Previously on CRAN: odesolve (Setzer 2001)

- Nice interface

- Two integration routines:
  - RungeKutta, not meant to be used
  - lsoda, good for small, simple models

- Models implemented in R or compiled code DLL (fast)

**BUT:**

- Only simplest Ordinary Differential Equations (ODE)
- not flexible,
- not suited for large problems
Now on CRAN:

- **deSolve** (Soetaert, Petzoldt, Setzer)
  - Initial value problems
  - > 10 integration routines
  
  - Simple and complex ODE
  - Differential algebraic equations (DAE)
  - Partial differential equations: (PDE)
    - 1-D, 2-D, 3-D problems
  
  - Flexible; Sparse, banded, full Jacobian
  - Medium-sized to large problems (up to 80000 state variables)

- **bvpSolve** (Soetaert)
  - Boundary value problems
  - 2 solution methods
hiv <- function(time, y, pars) {
  with(as.list(c(pars, y)), {
  dH <- lam - rho*H - bet*H*V
  dI <- bet*H*V - delt*I
  dV <- n*delt*I - c*V - bet*H*V
  return(list(c(dH, dI, dV)))
  })
}

y <- c(H=100, I=150, V=50000)
times <- 0:60
pars <- c(bet=0.00002, rho=0.15, delt=0.55, c=5.5, lam=80, n=900)
out <- ode(y=y, parms=pars, times=times, func=hiv)
plot(out[, "time"], out[, "H"])

Introduction
Dynamic differential equations
Steady-state solutions
Linear models
History/Outlook

HIV dynamics
Solving dynamic differential equations
Initial value differential equations in R
The HIV/AIDS model in R

Healthy cells

Infected cells

Viral load
Problem: dynamic models require many data:

\[
\frac{dC}{dt} = \sum f_i(C, \Theta, t, u, ...) - \sum f_j(C, \Theta, t, u, ...)
\]

- Knowledge of initial values
- Time-variable forcing functions (external data, \( u \))

=> Not always available

Solution:

- Assume steady-state
  => Systems of nonlinear equations

- Calculate stability properties
Previously on CRAN:

- \textit{uniroot} solves for one root of one nonlinear equation within interval

We need:

- Find all roots within one interval

- Functions to estimate gradient matrices, Jacobians (stability)

- Solve roots of \(n\) nonlinear equations (steady-state analysis)
Now on CRAN:

- `rootSolve` (Soetaert)
  - `uniroot.all`, `jacobian`: stability analysis
  - `multiroot`: roots of general nonlinear functions (Newton-Raphson)

- `steady`, `steady.1D`, `steady.2D`, `steady.3D`, `runsteady`:
  - steady-state solvers
    - Fully compatible with integration routines from `deSolve`
    - Suited for large problems (~100 000 equations)
    - Sparse, banded, full Jacobian

```r
STD <- runsteady(y=y, func=hiv, parms=pars)
eigen( jacobian.full(y=STD$y, func=hiv, parms=pars) )$values
```
Problem:
- Mechanistic nonlinear models have many parameters ($\theta$):
  - Many are unknown
  - Need to be fitted to data
  - Data not always available
- Nonlinear equations may not be known

Solution:
- Avoid nonlinear equations
- No parameters
- The sources and sinks ($f_{i\rightarrow j}$) are the unknowns
  - Linear model
Example 2: Deep-water coral food webs

Corals are commonly found at ~ 800-1000 m water depth.

A large number of animals are living in the coral reefs

It is very expensive to do research there

⇒ Data are very fragmentary

⇒ Who is eating who? How much do they eat?

⇒ A model is needed to see the global picture
Problem:

number of equations <<< number of unknowns (under determined)

Coral food web: 51 equations ~ 140 unknowns

⇒ There is no unique solution
  (~ fitting a straight line through one point)

Solution 1. Add data from other sources to equalities
  => achieve overdeterminacy (1 solution)

Solution 2. Data from other sources as “inequalities”
equality equation:  
(in situ data, mass balance)  

inequality equation:  
(literature data, physiological constraints,..)  

\[
\begin{align*}
\text{Ex} &= f \\
Gx &\geq h
\end{align*}
\]

\[
\begin{pmatrix}
a & -b & 0 \\
\vdots & \ddots & \vdots \\
0 & \cdots & 0
\end{pmatrix}
\begin{pmatrix}
f_1 \\
\vdots \\
f_n
\end{pmatrix}
\geq
\begin{pmatrix}
C \\
1
\end{pmatrix}
\]

food web flows

» the matrix equations are solved for the vector with food web flows
Dealing with the underdeterminacy:

Coral: Solution is a 140-dimensional SPACE!
⇒Within this space, every point equally likely
⇒3 different ways of solving:

\[
E x = f \\
G x \geq h
\]

- **Parsimonious** selects one solution
- **Ranges** estimate of flow range
- **Random sampling** flow distribution in ensemble

\[
\begin{align*}
\min\|Ax \approx b\| & \quad \min(x) \\
\min(\sum x^2) & \quad \max(x)
\end{align*}
\]
Previously on CRAN

- `solve.qp`, *(quadprog)*: quadratic programming
  \[
  \min \|Ax \approx b\|, Ex = f, Gx \geq h
  \]

- `lp`, *(lpSolve)*: linear programming
  \[
  \min( \sum a_i x_i), Ex = f, Gx \geq h
  \]

But:
- `solve.qp` tends to fail for some problems
- `lp` requires x to be positive (linear programming)
- `lp` and `solve.qp` are not compatible
- No monte carlo sampling of underdetermined systems
- Implementing large matrices: error-prone
Now on CRAN:

**limSolve** (Soetaert, van Oevelen, van den Meersche)
- least squares,
- linear programming,
- least distance programming
- `xranges, xsample`: range estimation and random sampling

**LIM** (Soetaert, van Oevelen)
- Models are specified in text files
Flowto(CO2) = 100

coral -> CO2 = [0.2,0.4] * Flowto(coral)

“coral.input”

require(LIM)
ocoral.lim <- Setup(“coral.input”)  
Parsimonious <- Ldei(coral.lim)
Ranges <- Xranges(coral.lim)
Xs <- Xsample(coral.lim, iter=10000)
Plotranges(order(colMeans(Xs)),…)

1x10^{-5} 1x10^{-3} 1x10^{-1} 1x10^{0}  
Flow value (mmol C m^{-2} d^{-1})
Linear
limSolve
LIM
Steady-state
nonlinear
rootSolve
Dynamic
deSolve

\[
\frac{dC_j}{dt} = \sum_{i} \text{flow}_{i \rightarrow j} - \sum_{k} \text{flow}_{j \rightarrow k}
\]

\[
0 = \sum f_i(C, \Theta, ...) - \sum f_j(C, \Theta, ...)
\]

\[
\frac{dC}{dt} = \sum f_i(C, \Theta, t, u, ...) - \sum f_j(C, \Theta, t, u, ...)
\]
Before 2006: Fortran, Excel, Powerpoint, Sigmaplot, own software

End 2005. First acquaintance with R

End 2006. Decision to use R for our scientific programming / graphics

⇒ Implement functions not yet available
<table>
<thead>
<tr>
<th>History/Outlook</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
</tr>
<tr>
<td>Dynamic differential equations</td>
</tr>
<tr>
<td>Steady-state solutions</td>
</tr>
<tr>
<td>Linear models</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>History/Outlook</th>
</tr>
</thead>
<tbody>
<tr>
<td>Now and Future</td>
</tr>
</tbody>
</table>

- Three years later...
  - Basic solution methods available
  - 5 Solver packages (deSolve, rootSolve, bvpSolve, limSolve, LIM)
  - Specific model applications

- Reactive transport models, (ReacTran)
  - rivers, estuaries, lakes, sediments

- Toxicology, (ToxWebs)
  - toxic substances in marine organisms

- Ecological network analysis (NetIndices)

- ....
THANK YOU
