

OOP in Ecological Modelling

Petzoldt, Rinke, Kates

Motivation Problem Workflow Basic idea

OOP in R State Machine What's typical? simObj

Implementation Example 1 Example 11 Scoping Nesting Benchmark Application

Con clusions

Population ecology modelling with R A Comparison of Object Oriented Approaches

Thomas Petzoldt¹ Karsten Rinke² Louis Kates³

¹Institute of Hydrobiology Technische Universität Dresden, Germany

> ²Limnological Institute Universität Konstanz, Germany

> > ³GKX Associates Inc. Waterloo, ON, Canada

Second use-R Conference Vienna, 2006

Outline

OOP in Ecological

Modelling

Petzoldt, Rinke,

Kates

Problem

Workflow

Basic idea

OOP in R

Example I

Example II

Benchmark

Application

Scoping

Nesting

simObj

State Machine

What's typical?

Motivation

The power of R and its problems A typical workflow Basic idea

Approach

OOP in R Ecological models as state machine What's typical in Ecological Models The proposed simObj specification

Implementation

A simple example A slightly more complex example Problems with scoping rules Handling nested functions Benchmark A practical problem

Conclusions



◆□▶ ◆□▶ ◆∃▶ ◆∃▶ ∃目 のへで

A Basic Lake Model

OOP in Ecological Modelling

Petzoldt, Rinke, Kates

Motivation Problem Workflow Basic idea

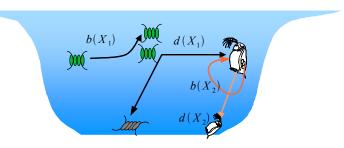
Approach OOP in R State Machine What's typical?

Implementatio Example I Example II Scoping Nesting Benchmark

simObi

Application

$\frac{dX}{dt} = b \cdot X - d \cdot X$ $b(X_1) = \frac{Light \cdot P_{max}}{Light \cdot K_1} \cdot \dots$



A Basic Lake Model

0 0

OOP in Ecological Modelling

DRESDEN

Petzoldt, Rinke, Kates

Motivation Problem Workflow

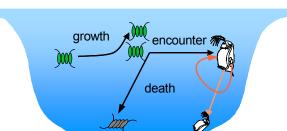
Workflow Basic idea

OOP in R State Machine What's typical? simObj

nplementation

Example I Example II Scoping Nesting Benchmark Application

on clusions



・ロト (個) (目) (目) (目) (日)



Example I

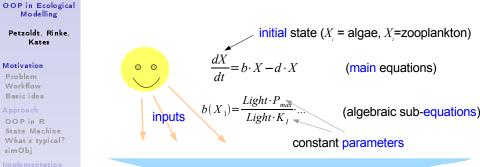
Example II

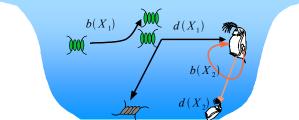
Benchmark

Application

Scoping Nesting

A Basic Lake Model





OOP in Ecological

R in Ecological Modelling

Modelling Petzoldt, Rinke

Problem

Kates

Workflow

Basic idea

OOP in R

simObj

Example I Example II

Scoping

Nesting

Benchmark

Application

State Machine

What's typical?

A great tool:

- Well suited to implement all types of models:
 - ODE (Lotka-Volterra "complete Lakes")

 - Individual-based
 - Grid-Based

DRESDEN **OOP** in Ecological

Modelling Petzoldt Rinke Kates

Problem

Basic idea

OOP in R State Machine What's typical? simObj

Example I Example II Scoping Nesting Benchmark Application

R in Ecological Modelling

A great tool:

- Well suited to implement all types of models:
 - ODE (Lotka-Volterra) "complete Lakes")

- Individual-based
- ▶ Grid-Based,

Problems:

- Different types of models
 - ► Different people, programming skills,
 - Few time for science no time for documentation.
 - Incompatible spaghetti-code.

TECHNISCHE R in Ecological Modelling

OOP in Ecological Modelling Petzoldt Binke

Kates

Problem

Workflow

Basic idea

OOP in R

Example I

Example II

Bench mark

Application

Scoping

Nesting

State Machine

What's typical?

A great tool:

- Well suited to implement all types of models:
 - ODE (Lotka-Volterra "complete Lakes")
 - Individual-based
 - ► Grid-Based,

Problems:

Different types of models

・ロト (個) (目) (目) (目) (日)

- ► Different people, programming skills,
- Few time for science no time for documentation.
- Incompatible spaghetti-code.
- Hack complete program to change only one parameter?
- Better write new code than re-use existing?



Modelling

Petzoldt, Rinke, Kates

Problem

Workflow

Basic idea

OOP in R

simObi

Example I

Example II

Benchmark

Application

Scoping

Nesting

State Machine

What's typical?

Workflow and requirements

Common tasks:

- Compare the same model with different data,
- Compare two different models with same data.

Typical application scenario:

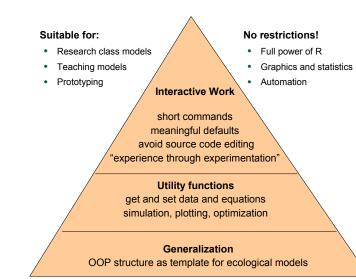
- Load the model,
- Run the model,
- Create scenarios,
- Compare scenarios.

Requirements:

- ► Ease of application,
- Meaningful defaults,
- Storage of results and settings.

Basic idea and goal

Provide a standard architecture and utility functions and propagate a common style.



Workflow and requirements

Common tasks:

- ► Compare the same model with different data,
- Compare two different models with same data.

Typical application scenario:

- Load the model,
- Run the model,
- Create scenarios,
- ► Compare scenarios.

Requirements:

- Ease of application,
- Meaningful defaults,
- Storage of results and settings.

▲□▶ ▲圖▶ ▲필▶ ▲필▶ - 원벨 - 위역은

Approach:

```
OOP in Ecological
Modelling
```

Kates

Problem

Basic idea

OOP in R

Example I

Example II

Bench mark

Application

Scoping

Nesting

State Machine

What's typical?

Approach

OOP template and package – simplify and unify ecological modelling with R

► Which OOP approaches are available?

- ▶ What is typical in ecological modelling?
- Provide an R Package with one selected OOP paradigm.

◆□▶ ◆□▶ ◆∃▶ ◆∃▶ ∃目 のへで

OOP in Ecological Modelling Petzoldt, Rinke, Kates

DRESDEN

Problem Workflow Basic idea

Approach OOP in R State Machine What's typical? simObj

mplementation Example I

Example II Scoping Nesting Benchmark Application

Con clusions

Motivation Problem Workflow Basic idea Approach OOP in R State Machine What's typical? simObj Implementation Example I Example I Scoping Nesting Benchmark

OOP in Ecological

Modelling

Petzoldt, Rinke

Kates

Application Conclusions



OOP in R

OOP in Ecological Modelling Petzoldt, Rinke

Problem

Workflow Basic idea

OOP in R

What's typical? simObi

Example I Example II Nesting Benchmark Application

- Kates

State Machine

Scoping

- R.oo: a contributed OOP system (Bengtsson, 2003) based on S3: method consistency, references, documentation facility,
- proto : class-less (prototype-based) OOP (Kates & Petzoldt, 2005): intentionally lightweight, delegation (prototype form of inheritance), references

S4 : the new standard OOP system (Chambers, 1998); ensures

Questions:

- ▶ Is there a best OOP system for ecological modelling?
- ► Does OOP kill performance?

Different types of data:

state variables.

► The main model

▶ input values,

▶ time steps

class)

parameters (constants),

Different types of functional information

Several OOP systems in R:

method consistency,

S3 : original class system of R,

▶ Does end user code depend on the OOP selected?

Tight relationship between methods (equations) and data

a set of (possibly nested) sub-models (sub-equations)



Characteristics of ecological models

OOP in Ecological Modelling

Petzoldt Rinke Kates

Problem Basic idea

OOP in R State Machine What's typical?

- simObi
- Example I Example II Scoping Nesting Benchmark Application

◆□▶ ◆□▶ ◆∃▶ ◆∃▶ ∃目 のへで TECHNISCHE DRESDEN

OOP in Ecological

OOP in Ecological Modelling

Petzoldt Rinke Kates

Problem Workflow Basic idea

OOP in R State Machine What's typical? simObj

Example I Example II Scoping Nesting Bench mark

solvers, integrators, visualization	(common within one model	Application
class)	Υ.	Conclusions



Modelling Petzoldt, Rinke Kates external inputs Problem Workflow Basic idea OOP in R initial State Machine main model outputs conditions What's typical? simObj Example I Example II Scoping time steps Nesting from, to, by **Benchmark** Nested subequation Application solver: t = 1+dt

The SimObj model specification



main:

equat

parms

times

init:

input

solve out:

simecol-package						
S4 class: simObj	Generics:					
function	- sim, plot, print					
tions: list of functions	- get/set -functions					
: data	Solvers					
: data	- lsoda-wrapper, rk4					
data	- iteration					
ts: data						
	Utility functions					
er: character	- approxTime					
data	- neighbours					

・ロト (個) (目) (目) (目) (日)



Implementation: S4 version

of the Lotka-Volterra model

lv <- new("OdeModel",</pre>

... with sub-equations:

equations = list(

f1 = function(x, K)

times = seq(0, 10, 0.1),

= c(x=0.5)

main = function (equations, x)

dx1 < f2(x[1], 0.1, 10)

model <- list(</pre>

{

},

),

)

init



Petzoldt, Rinke, Kates

```
Problem
Workflow
Basic idea
```

OOP in R State Machine What's typical? simObj

Implementation

Example I Example II Scoping Nesting Benchmark Application

x <- init with(as.list(parms), { dx1 < b * x[1] - e * x[1] * x[2] $dx_2 < - - d * x_2 + e * x_1 * x_2$ list(c(dx1, dx2))}) , ## birth encounter death = c(b=0.2, e=0.2, d=0.2),parms = seq(0, 100, 1),times = c(prey=0.5, predator=1) init)

main = function (time, init, parms)

S3, S4, R.oo, proto: The model objects are guite similar.

K - x.

f2 = function(x, r, K) r * x * f1(x, K)

```
TECHNISCHI
UNIVERSITAT
    DRESDEN
```

OOP in Ecological Modelling

```
Petzoldt Rinke
    Kates
```

Problem Basic idea

```
OOP in R
State Machine
What's typical?
simObi
```

```
Example I
Example II
Scoping
Nesting
Benchmark
```

Application

```
◆□▶ ◆□▶ ◆目▶ ◆目▶ 目目 のへで
                                                                        TECHNISCHE
A slightly more complex example ....
                                                                     DRESDEN
                                                                     OOP in Ecological
                                                                       Modelling
                                                                     Petzoldt Binke
                                                                         Kates
                                                                     Problem
                                                                     Basic idea
```

OOP in R State Machine What's typical? simObi Example I Example II Scoping Nesting Bench mark Application

A short example

```
OOP in Ecological
    Modelling
 Petzoldt, Rinke
```

Problem

Workflow

Basic idea

OOP in R

Example I

Scoping

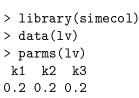
Nesting

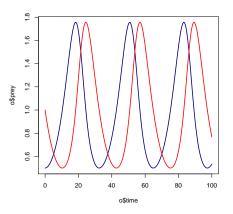
Benchmark

Application

simObj

Kates State Machine What's typical? Example II





> lv <- sim(lv) # pass-back modification > plot(lv)> o <- out(lv)

```
> plot (o$time, o$prey,
                            col="navy", lwd=2, type="1")
> lines(o$time, o$predator, col="red", lwd=2)
                               ◆□▶ ◆□▶ ◆目▶ ◆日▶ 三目目 のへで
```

More complex models:

Problems with scoping rules

- Lexical scoping in R
- Sub-equations assembled in a common structure (a list)
- ▶ How can these functions see each other ?
- Two possible approaches:
 - A) pass the whole object (or parts of it) down to the called function.
 - B) provide all necessary functions and data within a local environment.



OOP in Ecological Modelling Petzoldt, Rinke, Kates

Problem Workflow

Basic idea

OOP in R State Machine What's typical? simObj

Example I Example II Scoping Nesting

Benchmark Application



Benchmarks ... are more or less subjective

OOP in Ecological Modelling

Petzoldt Rinke Kates

Problem Workflow Basic idea

OOP in R State Machine What's typical?

simObi

Example I

Example II Scoping Nesting

Benchmark Application

A) Object Passing eqA <- list(f1 = function(eq, x, K)К-х, f2 = function(eq, x, r, K) r * x * eq\$f1(eq, x, K)) solverA <- function(eq) {</pre> eq\$f1(eq, 3, 4) + eq\$f2(eq, 1, 2, 3)}

solverA(eqA)

A) Object Passing

```
OOP in Ecological
              eqA <- list(
                f1 = function(eq, x, K)
                                           K - x,
Petzoldt, Rinke,
                f2 = function(eq, x, r, K) r * x * eq$f1(eq, x, K)
              )
              solverA <- function(eq) {</pre>
                eq$f1(eq, 3, 4) + eq$f2(eq, 1, 2, 3)
              7
              solverA(eqA)
What's typical?
              B) Temporary Environment
              eqB <- list(
                f1 = function(x, K)
                                       K - x,
                f2 = function(x, r, K) r * x * f1(x, K)
              )
              solverB <- function(eq) {</pre>
                eq <- putInEnv(eq, environment()) # a little trick</pre>
                f1(3,4) + f2(1,2,3)
              }
              solverB(eqB)
```



Modelling

Kates

Problem

Workflow

Basic idea

OOP in R

simObj

Example I

Example II

Benchmark

Application

Scoping

Nesting

State Machine

Benchmarks ... are more or less subjective

OOP in Ecological Modelling

Petzoldt Binke Kates

Problem

... and here is one:

Model	Size	nested	S 3	S 4	R.oo	proto	simecol	
Lotka-Volterra	small	no	3.5	3.6	3.6	3.9	3.7	(a)
Extended Lotka-	small	yes	4.8	4.8	4.9	5.1	4.8	(b)
Volterra								. ,
DEB (bioener-	medium	yes	2.8	2.8	2.9	3.0	2.7	(c)
getic Daphnia		-						. ,
model)								

Workflow Basic idea OOP in R State Machine What's typical?

Example I Example II Scoping

simObi

Nesting Benchmark

Application



OOP in Ecological Modelling

Petzoldt, Rinke, Kates

Problem Workflow

Basic idea

OOP in R State Machine What's typical simObj

Example I Example II Scoping Nesting Benchmark

Application



A practical application

and here is one:

(bioener-

Daphnia

Size

small

small

medium

of OOPs quite equal (with ecological models !)

Model

Volterra DEB

getic

model)

Lotka-Volterra

Extended Lotka-

Performance:

Demographically structured population dynamics model of Daphnia

Benchmarks ... are more or less subjective

nested

no

yes

yes

S3

3.5

4.8

2.8

S4

3.6

4.8

2.8

R.00

3.6

4.9

2.9

proto

3.9

5.1

3.0

simecol

(a)

(b)

(c)

3.7

4.8

2.7

Petzoldt Rinke Kates

Modelling

Problem Workflow Basic idea

OOP in R State Machine What's typical? simObi

Example I Example II Scoping Nesting Benchmark

Application

Abundance (Ind. L⁻¹) & Weight (µg) Day 235 0.15 10 Weight 8 Abundance Food (mgC L⁻¹ 0.1 6 4 0.05 2 0 5 10 15 20 25 30 Age class (d)

This model consists of two parts: individual level: bioenergetic approach (differential equations) population level: discrete age-structure (cohort-based) details, see Rinke & Vijverberg (2005)

Benchmarks ... are more or less subjective

OOP in Ecological Modelling

Petzoldt, Rinke

Kates

Problem

Workflow

Basic idea

OOP in R

Example I

Example II

Benchmark

Application

DRESDEN

Problem

OOP in R

Example I

Example II

Bench mark

Application Conclusions

Scoping

Nesting

State Machine

What's typical?

Scoping

Nesting

State Machine

What's typical?

and here is one:

Model	Size	nested	S 3	S 4	R.oo	proto	simecol	
Lotka-Volterra	small	no	3.5	3.6	3.6	3.9	3.7	(a)
Extended Lotka-	small	yes	4.8	4.8	4.9	5.1	4.8	(b)
Volterra								
DEB (bioener-	medium	yes	2.8	2.8	2.9	3.0	2.7	(c)
getic Daphnia								
model)								

Performance:

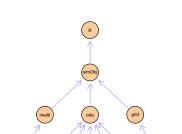
of OOPs quite equal (with ecological models !)

Reason:

- OOP used only to structure models.
- Excessive use of OOP features not necessary.
- Time consuming parts: variable assignments and numerics.

Conclusion: Use R – and OOP





・ロト (個) (目) (目) (目) (日)



Conclusion: Use R – and OOP

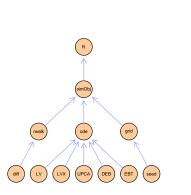
Modelling Petzoldt, Rinke, Kates

Notivation Problem Workflow Basic idea

OOP in R State Machine What's typical? simObj

Example I Example I Scoping Nesting Benchmark Application

Con clusions



► It's more important to use OOP at all than *the right* OOP.

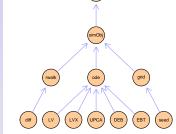
TECHNISCHE UNIVERSITÄT DRESDEN

OOP in Ecological Modelling

Petzoldt, Rinke, Kates Motivation Problem Workflow Basic idea Approach OOP in R State Machine What's typical? simObj Implementation Example I Example II Scoping Nesting Benchmark Application

Conclusion: Use R - and OOP

e,



- It's more important to use OOP at all than the right OOP.
- OOP helps to structure ecological models.
 R provides all mechanisms necessary.

$Conclusion: \ Use \ R-and \ OOP$



Petzoldt, Rinke Kates

Motivation Problem Workflow Basic idea

Approach OOP in R State Machine What's typical? simObj

mplementation Example I Example II Scoping Nesting Benchmark

Application Conclusions

R marcos rvalk colo grid df UV UX UPCA DEB EBT ceed

- ► It's more important to use OOP at all than *the right* OOP.
- OOP helps to structure ecological models. R provides all mechanisms necessary.
- The proposed OOP structure works without and with simecol.

Conclusions

OOP in Ecological Modelling

Petzoldt, Rinke Kates

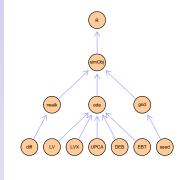
Motivation Problem Workflow Basic idea

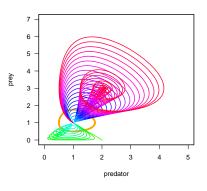
Approach OOP in R State Machine What's typical?

Implementation Example 1 Example 11 Scoping Nesting Benchmark Application

Conclusions

Conclusion: Use R – and OOP





・ロト (個) (目) (目) (目) (日)

- ► It's more important to use OOP at all than the right OOP.
- OOP helps to structure ecological models. R provides all mechanisms necessary.
- The proposed OOP structure works without and with simecol.

◆□▶ ◆□▶ ◆目▶ ◆目▶ 目目 のへで



OOP in Ecological Modelling

Petzoldt, Rinke, Kates

References **Object Passing**

Cellular Automata

- Bengtsson, H., 2003: The R.oo package object-oriented programming with references using standard R code. In: K. Hornik, F. Leisch, & A. Zeileis (eds.), Proceedings of the 3rd International Workshop on Distributed Statistical Computing, Vienna, Austria, http://www.maths.lth.se/help/R/R.oo/.
- Chambers, J. M., 1998: Programming with Data: A Guide to the S Language. Springer-Verlag, New York.
- Kates, L. & T. Petzoldt, 2005: The R Proto Package. Package vignette of the CRAN proto package and http://hhbio.wasser.tu-dresden.de/projects/proto/.
- Rinke, K. & J. Vijverberg, 2005: A model approach to evaluate the effect of temperature and food concentration on individual life-history and population dynamics of Daphnia. Ecological Modelling 186: 326-344.

TECHNISCHE DRESDEN

OOP in Ecological Modelling

Petzoldt, Rinke, Kates

References **Object** Passing Cellular Automata put in Env

Additional slides for discussion.

OOP in Ecological Modelling

Petzoldt Rinke

Kates

Object Passing

Cellular Automata

Pass the equation object down where it is needed.

x[i,] <- x[i-1,] + obj\$main(obj\$equations, x[i-1,]) * dt[i-1]

x <- matrix(NA, length(obj\$times), length(obj\$init))</pre>

dx1 <- equations\$eq1(equations, x[1], 0.1, 10)</pre>

eq1 = function(this, x, r, K) r * x * this\$f(x, K),

sim <- function(obj) {</pre>

x[1,] <- obj\$init;</pre> dt <- diff(obj\$times)

equations = list(

init = c(x=0.5)

times = seq(0, 10, 0.1),

obj\$out <- x

model <- list(</pre>

obj }

},

)

for (i in 2:length(obj\$times)) {

main = function (equations, x)

f = function(x, K) (K - x)),

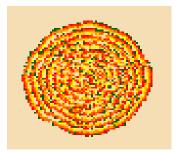


Stochastic cellular automaton

OOP in Ecological Modelling

Petzoldt Binke Kates

Object Passing Cellular Automata put In Env



source("http://www.simecol.de/data/ca.R") times(CA) < - c(to=80)sim(CA, animate=TRUE, col=mycolors(20), axes=F)

model <- sim(model)</pre> plot(model\$times, model\$out[,1], type="1")

```
putInEnv <- function(eq, e) {</pre>
OOP in Ecological
  Modelling
                  ## clone, very important to avoid "interferences"!!!
Petzoldt, Rinke,
                  eq <- as.list(unlist(eq))</pre>
    Kates
                  lapply(eq, "environment<-", e)</pre>
                  nn <- names(eq)</pre>
Object Passing
                  for (i in 1:length(eq)) {
Cellular Automata
putinEnv
                    assign(nn[i], eq[[i]], envir = e)
                  }
                  eq
                }
                eqB <- list(
                  f1 = function(x, y) x + y,
                  f2 = function(a, x, y) a * f1(x, y)
                )
                solverB <- function(eq) {</pre>
                  eq <- putInEnv(eq, environment())</pre>
                  f1(3,4) + f2(1,2,3)
                }
```