Exercise 1: Lake Phytoplankton Model

ETH Zurich Course 701-0426-00L: Modelling Aquatic Ecosystems (Schuwirth/Reichert)

26.02.2020

Goals:

- Review basic elements of R needed to handle the technical aspects of the exercises.
- Understand the basic structure and functioning of the R package 'ecosim'.
- Be able to implement the simple lake phytoplankton model described in section 11.1 of the manuscript and to run simulations.
- Understand the behaviour of the solutions of this model.

Notes

- To conduct the exercises, install the newest version of R on your computer from http://r-project.org. We recommend to use RStudio as an editor for R: http://rstudio.com. The required packages `ecosim`, `stoichcalc` and `deSolve` can be installed by executing the commands
  ```
  install.packages("stoichcalc")
  install.packages("deSolve")
  install.packages("ecosim")
  ```
  Note that the last command installs the required packages also if the are not yet installed. However, only installing them explicitly guarantees that the newest version is installed (because required packages are not re-installed if the are already installed).
- The example files can be downloaded from the course homepage http://www.eawag.ch/forschung/siam/lehre/modaqecosys.

Task 1: Introduction to R

Become familiar with R. See presentation and separate documentation.

Other useful resources can be found at http://r-project.org, in particular:

- http://cran.r-project.org/manuals.html
- http://cran.r-project.org/other-docs.html
- http://cran.r-project.org/doc/manuals/r-release/R-intro.pdf

Task 2: Introduction to the package `ecosim`

See presentation, documentation in section 16 of the manuscript and the manual `ecosim.pdf` at http://cran.r-project.org/package=ecosim.

Task 3: Implementation of a simple lake phytoplankton model with constant driving forces

Carefully study the implementation of the model described in section 11.1 given below for the case of constant driving forces. Become familiar with the creation of objects of the classes `process`, `reactor`, and `system` and

link their entries to the system description provided in section 11.1. In particular, check the implementation of the object of the class `process` with the process table notation introduced in the course.

Run the model implementation based on the accompanying file `exercise_1.R`.

Task 4: Perform simuations with the simple lake phytoplankton model

Study how to do simulations and plots once a model has been defined as an object of the class `system` as given below.

Perform simulations with the model and try different plotting options based on the accompanying file `exercise_1.R` and on your own ideas. Interpret the results.

Task 5: Extend the model to periodic driving forces and do simulations

Study the extension of the model to periodic driving forces as documented below.

Run the model extensions based on the accompanying file `exercise_1.R`, perform simulations, and plot and interpret the results.

Task 6: Homework: Simple sensitivity analysis

Do simulations with modified values of the parameters $C_{\text{in,HPO}_4^{2-}}$, $k_{\text{gro,ALG}}$, $k_{\text{death,ALG}}$, $K_{\text{HPO}_4^{2-},\text{ALG}}$ using the function `calcsens()`. Type `?calcsens` to get the help file for this function and to see the default values. By default each parameter given in the list of `param.sens` is increased by a factor of 2 and decreased by a factor of 1/2. Try to understand the model responses to changes in the model input ($C_{\text{in,HPO}_4^{2-}}$) and process parameters ($k_{\text{gro,ALG}}$, $k_{\text{death,ALG}}$, $K_{\text{HPO}_4^{2-},\text{ALG}}$) under constant environmental conditions.

Questions

1. How can you derive the total (net) transformation rate of $C_{\text{HPO}_4^{2-}}$ and C_{ALG} from the process table (Table 11.1) and the process rates (Table 11.2)? Hint: see equation (4.1) in the manuscript. What are the units?
2. Look at equilibrium concentrations for both state variables (given at the beginning of the R-script and by the equations 11.11 and 11.12). On which parameters or model input variables do the equilibrium concentrations for the state variables depend?
3. Look at the state variables $C_{\text{HPO}_4^{2-}}$ and C_{ALG}. Which of them is more sensitive to the parameter $K_{\text{HPO}_4^{2-},\text{ALG}}$ and which of them is more sensitive to $C_{\text{in,HPO}_4^{2-}}$? Do you understand why?

Solution of Task 3

We first have to load the package `ecosim` (and install if it is not yet installed). Note that this will automatically load (and install) the packages `deSolve` that is used to numerically integrate the system of ordinary differential equations that mathematically describes the dynamics of the system, and the package `stoichcalc` that will be used in the exercises 3 to 5 for calculating process stoichiometries.

```
# load required packages:

if ( !require("ecosim") ) {install.packages("ecosim"); library(ecosim) }
```

```
## Loading required package: ecosim
```

```
## Loading required package: deSolve
```

```
## Loading required package: stoichcalc
```

Next we define a named list of model parameters:

```
# definition of model parameters:

param    <- list(k.gro.ALG   = 0.5,      # 1/d
                 k.death.ALG = 0.1,      # 1/d
                 K.HPO4      = 0.002,    # gP/m3
                 alpha.P.ALG = 0.003,    # gP/gDM
                 A           = 5e+006,   # m2
                 h.epi       = 5,        # m
                 Q.in        = 5,        # m3/s
                 C.HPO4.in   = 0.04,     # gP/m3
                 C.HPO4.ini  = 0.004,    # gP/m3
                 C.ALG.ini   = 0.1)      # gDM/m3
```

The following function calculates the equilibrium concentrations as a function of the list of model parameters defined above according to the equations (11.10) - (11.14) in the manuscript:

```
# function to calculate equilibrium concentrations:

equilib <- function(param)
{
   eq <- c(C.HPO4=param$C.HPO4.in,C.ALG=0)
   dil <- param$Q.in*86400/(param$h.epi*param$A)
   if ( param$k.gro.ALG > param$k.death.ALG+dil &
          param$C.HPO4.in > param$K.HPO4/(param$k.gro.ALG/(param$k.death.ALG+dil)-1) )
   {
     eq["C.HPO4"] <- param$K.HPO4/(param$k.gro.ALG/(param$k.death.ALG+dil)-1)
     eq["C.ALG"]  <- dil/(param$k.death.ALG+dil)/param$alpha.P.ALG*
                       (param$C.HPO4.in-param$K.HPO4/(param$k.gro.ALG/(param$k.death.ALG+dil)-1))
   }
   return(eq)
}
```

Next, we define the processes of growth and death of algae as objects of the class `process` of the package `ecosim`. Each process is defined by its name, rate, and stoichiometry. The rate is defined as an expression that can use parameters (defined for the object of class `system` below), concentrations defined in objects of class `reactor` that are part of the object of class `system`. To define the stoichiometry a named list of expressions must be provided that identifies the substance or organism concentrations as the names and contains the stoichiometric coefficients as expressions.

```
# definition of transformation processes

# growth of algae:

gro.ALG   <- new(Class  = "process",
                 name   = "Growth of algae",
                 rate   = expression(k.gro.ALG
                                     *C.HPO4/(K.HPO4+C.HPO4)
                                     *C.ALG),
                 stoich = list(C.ALG  = expression(1),              # gDM/gDM
                               C.HPO4 = expression(-alpha.P.ALG)))  # gP/gDM

# death of algae:

death.ALG <- new(Class = "process",
                 name   = "Death of algae",
                 rate   = expression(k.death.ALG*C.ALG),
                 stoich = list(C.ALG  = expression(-1)))            # gDM/gDM
```

Next, we define the mixed box describing the epliminion of the lake as an object of the class `reactor` of the package `ecosim`.

```
# definition of reactor to describe the epilimnion of the lake:

epilimnion <-
   new(Class            = "reactor",
       name             = "Epilimnion",
       volume.ini       = expression(A*h.epi),
       conc.pervol.ini  = list(C.HPO4 = expression(C.HPO4.ini),     # gP/m3
                               C.ALG  = expression(C.ALG.ini)),     # gDM/m3
       inflow           = expression(Q.in*86400),                   # m3/d
       inflow.conc      = list(C.HPO4 = expression(C.HPO4.in),
                               C.ALG  = 0),
       outflow          = expression(Q.in*86400),
       processes        = list(gro.ALG,death.ALG))
```

Finally, we combine the reactor, the parameters, and the desired output times in an object of class `system` of the package `ecoval`.

```
# definition of the system consisting of a single reactor:

system.11.1.a <- new(Class    = "system",
                     name     = "Lake",
                     reactors = list(epilimnion),
                     param    = param,
                     t.out    = seq(0,365,by=1))
```

Note that this object contains all definitions of the configuration of reactors (in this case just a single one), the processes active in each reactor, the model parameters, and the output time points. Any simulations carried out will refer to the definitions in this object, and not to the external variables that we used to set up the elements of the system.

Solution of Task 4

Calculate and print equilibrium solutions using the parameter values from the object `system.11.1.a`

```
# calculate and print equilibrium solutions:

print(equilib(system.11.1.a@param))
```

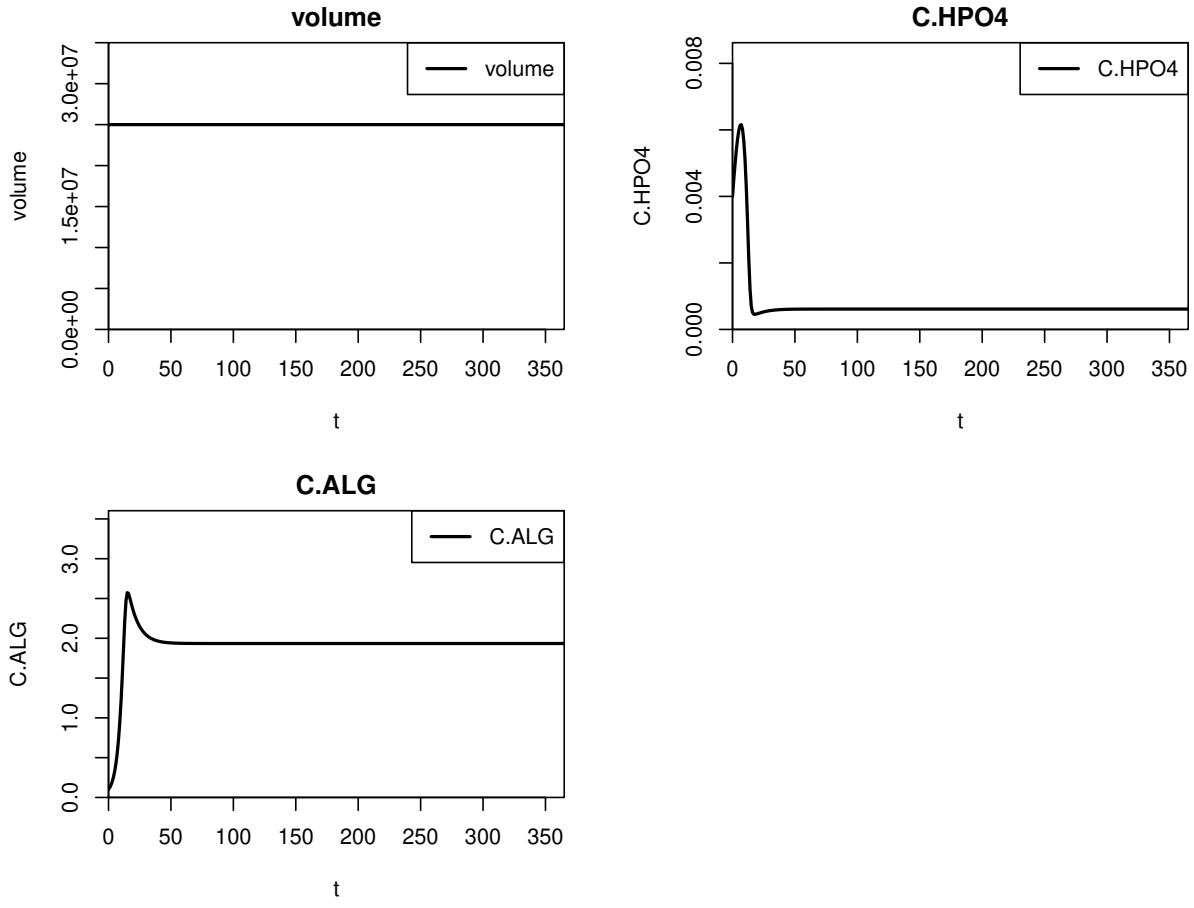
```
##       C.HPO4        C.ALG
## 0.0006128763 1.9344289971
```

Perform a simumlation and store the results by using the function `calcres` of the package `ecosim`:

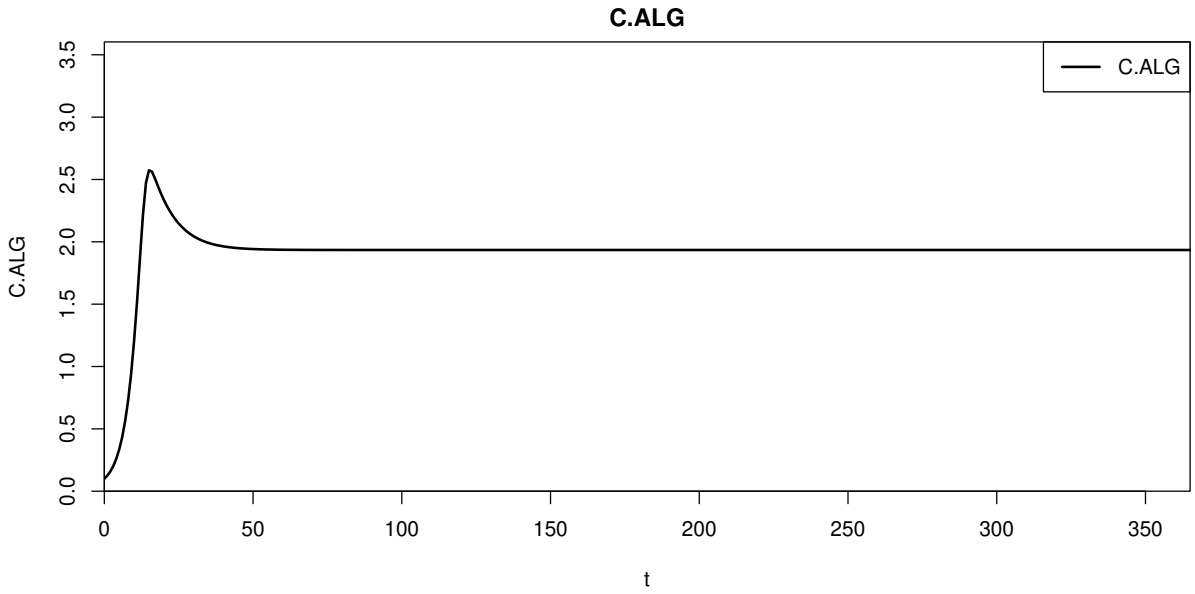
```
# perform simulation:

res.11.1.a <- calcres(system.11.1.a)
```

Plot results using the plotting function `plotres` of the package `ecosim`:

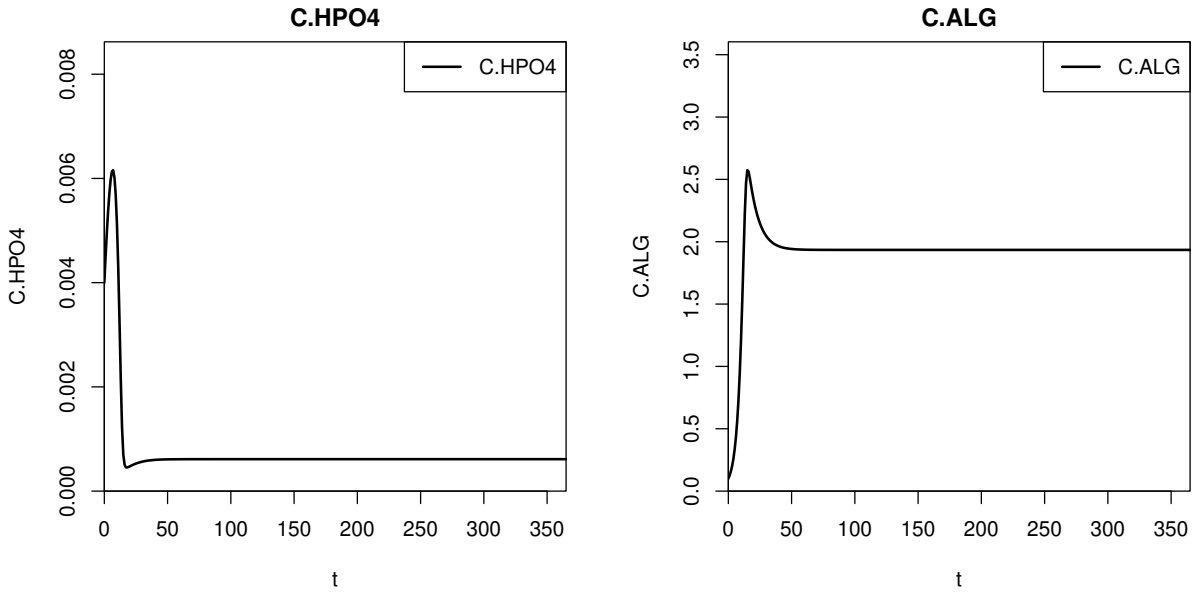
```
# plot results wit default options:

plotres(res.11.1.a)
```




```
# plot only results for the concentration of algae:

plotres(res=res.11.1.a,colnames="C.ALG")
```




```
# plot results for phosphate and algae:

plotres(res=res.11.1.a,colnames=list("C.HPO4","C.ALG"))
```

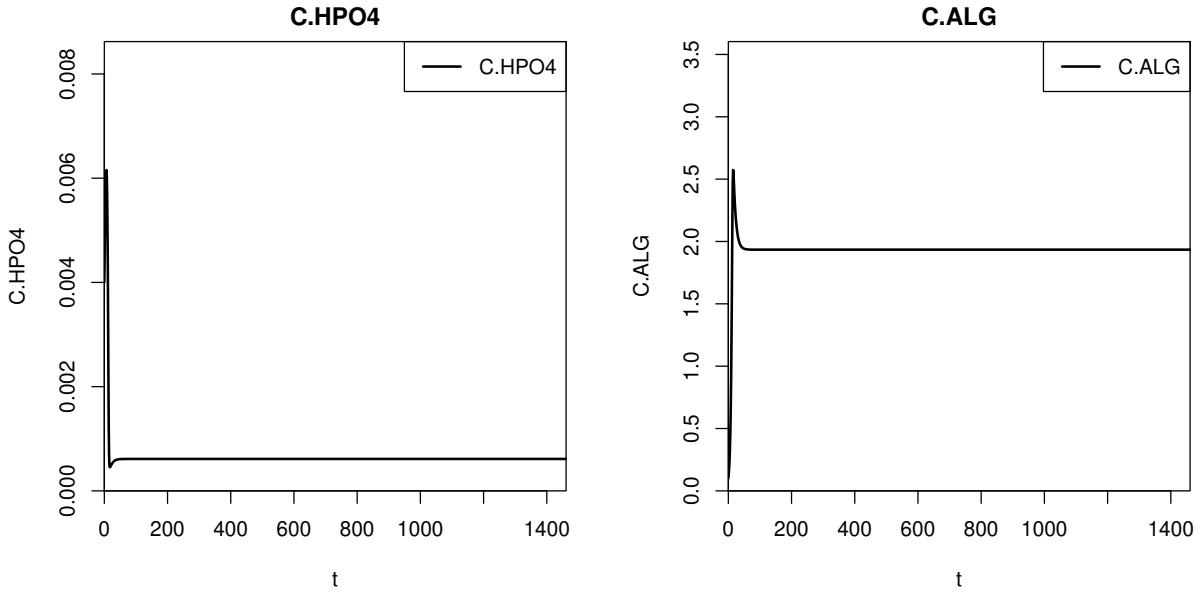



```
# plot results to a file:

plotres(res      = res.11.1.a,
        colnames = list("C.HPO4","C.ALG"),
        file     = "exercise_1_results_a1.pdf",
        width    = 10,
        height   = 5)
```

```
## pdf
##   2
```

Simulate for 4 years and plot results:

```
# change simulation time to 4 years:

system.11.1.a@t.out <- seq(0,4*365.25,by=1)

# calculate results for the system with modified simulation time

res.11.1.a.4y <- calcres(system.11.1.a)

# plot results

plotres(res=res.11.1.a.4y,colnames=list("C.HPO4","C.ALG"))
```




```
# plot results tp pdf

plotres(res      = res.11.1.a.4y,
        colnames = list("C.HPO4","C.ALG"),
        file     = "exercise_1_results_a2.pdf",
        width    = 10,
        height   = 5)
```

```
## pdf
##   2
```

Solution of Task 5

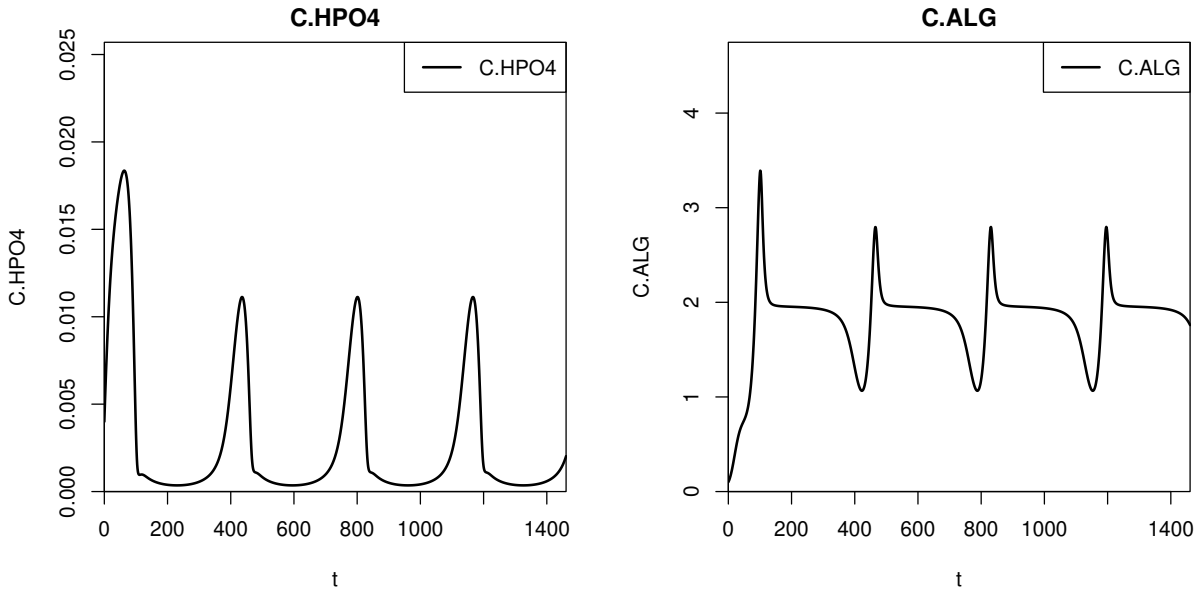
Adapt system definitions:

```
# extend system definitions:

system.11.1.b <- system.11.1.a

# extend growth of algae by environmental factors:

gro.ALG.ext <-
   new(Class  = "process",
       name   = "Growth of algae extended",
       rate   = expression(k.gro.ALG
                           *exp(beta.ALG*(T-T0))
                           *C.HPO4/(K.HPO4+C.HPO4)
                           *log((K.I+I0)
                                /(K.I+I0*exp(-(lambda.1+lambda.2*C.ALG)*h.epi)))
                            /((lambda.1+lambda.2*C.ALG)*h.epi)
                           *C.ALG),
       stoich = list(C.ALG  = 1,                              # gDM/gDM
                     C.HPO4 = expression(-alpha.P.ALG)))      # gP/gDM

# re-define processes in the reactor "epilimnion":

epilimnion@processes <- list(gro.ALG.ext,death.ALG)

# make environmental conditions (light and temperature) time dependent:

epilimnion@cond <- list(I0 = expression(0.5*(I0.min+I0.max)+
                                        0.5*(I0.max-I0.min)*
                                        cos(2*pi/365.25*(t-t.max))),    # W/m2
                        T  = expression(0.5*(T.min+T.max)+
                                        0.5*(T.max-T.min)*
                                        cos(2*pi/365.25*(t-t.max))))  # degC

# re-define the reactor "epilimnion" in the system definition:

system.11.1.b@reactors <- list(epilimnion)

# extend model parameters:

param  <- c(param,
            list(beta.ALG    = 0.046,     # 1/degC
                 T0          = 20,        # degC
                 K.I         = 30,        # W/m2
                 lambda.1    = 0.10,      # 1/m
                 lambda.2    = 0.10,      # m2/gDM
                 t.max       = 230,       # d
                 I0.min      = 25,        # W/m2
                 I0.max      = 225,       # W/m2
                 T.min       = 5,         # degC
                 T.max       = 25))       # degC

# increase algal growth rate to compensate for new limitatations:
```

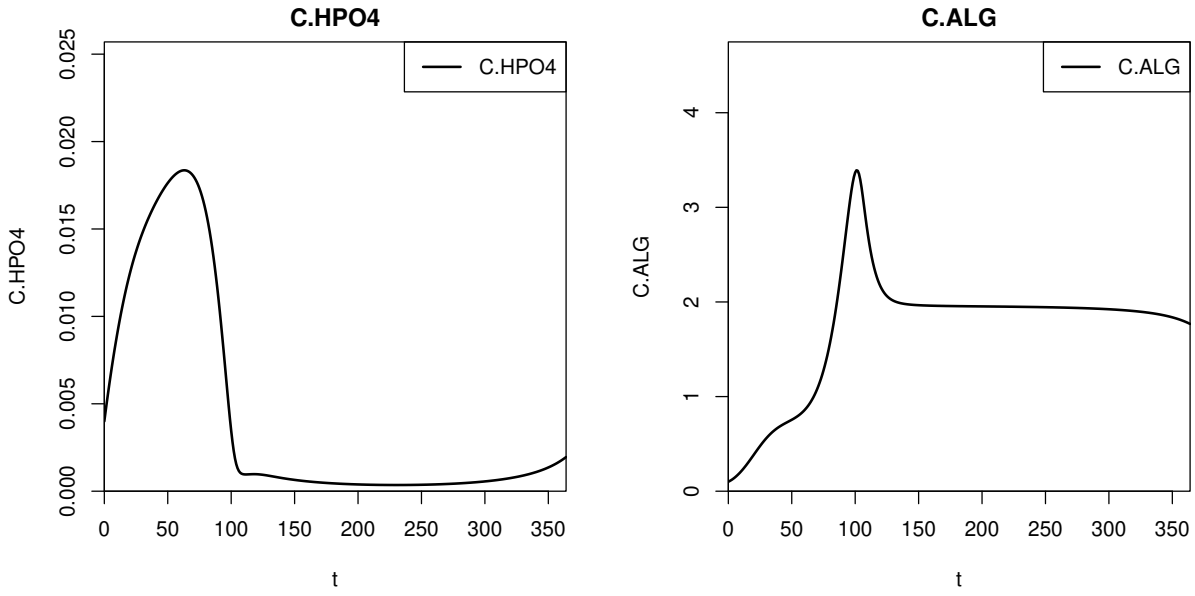
```r
param$k.gro.ALG <- 0.8

# replace parmeters in the system definition:

system.11.1.b@param <- param
```

Redo simulatons and plot results:

```r
# Redo simulations and plot results:

res.11.1.b <- calcres(system.11.1.b)

plotres(res=res.11.1.b, colnames=list("C.HPO4","C.ALG"))
```




```r
plotres(res=res.11.1.b[1:365,], colnames=list("C.HPO4","C.ALG"))
```




```
plotres(res      = res.11.1.b,    # plot to pdf file
        colnames = list("C.HPO4","C.ALG"),
        file     = "exercise_1_results_b1.pdf",
        width    = 10,
        height   = 5)
```

```
## pdf
##   2
```

```
# comparision of the two simulations:

plotres(res      = list(const=res.11.1.a.4y,dyn=res.11.1.b),
        colnames = list("C.HPO4","C.ALG"),
        file     = "exercise_1_results_ab.pdf",
        width    = 10,
        height   = 5)
```

```
## pdf
##   2
```