Exercise 5: Benthic Population Model of a River

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

May 8, 2024

```
# to display chunk code along with its results
knitr::opts_chunk$set(echo = TRUE)
```

Goals:

- Understand the concepts and process formulations of the coupling of benthic population dynamics with nutrient and dissolved oxygen dynamics in the water column as described in section 11.6.
- Understand the implementation of this model based on the R packages `stoichcalc` and `ecosim`.
- Understand the behaviour (time courses of the state variables and overall mass fluxes of this model).
- Perform some simple sensitivity analyses to deepen the understanding of the behaviour of the model.

Task 1: Model formulation

Study and try to understand the model formulation given in section 11.6. Note that it may be useful to study first the ''intermediately complex" model described in section 11.5.

Task 2: Model implementation

Study, complete and run the implementation of the model given below.

Load the package `ecosim` (and install if it is not yet installed).

```
# load required packages:

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

Note that the packages `deSolve`, to numerially solve systems of ordinary differential equations, and `stoichcalc`, to calculate process stoichiometry were loaded automatically.

Define parameter values for stoichiometry, kinetics, river geometry input and environmental conditions:

```
# Definition of parameters:

param <- list(
  # stoichiometric parameters
  alpha.O.ALG     = 0.50,         # gO/gALG
  alpha.H.ALG     = 0.07,         # gH/gALG
  alpha.N.ALG     = 0.06,         # gN/gALG
  alpha.P.ALG     = 0.01,         # gP/gALG
```

```
  alpha.O.BAC     = 0.30,         # gO/gBAC
  alpha.H.BAC     = 0.08,         # gH/gBAC
  alpha.N.BAC     = 0.10,         # gN/gBAC
  alpha.P.BAC     = 0.02,         # gP/gBAC
  alpha.O.POM     = 0.26,         # gO/gPOM
  alpha.H.POM     = 0.07,         # gH/gPOM
  alpha.N.POM     = 0.04,         # gN/gPOM
  alpha.P.POM     = 0.01,         # gP/gPOM
  alpha.O.DOM     = 0.26,         # gO/gDOM
  alpha.H.DOM     = 0.07,         # gH/gDOM
  alpha.N.DOM     = 0.04,         # gN/gDOM
  alpha.P.DOM     = 0.01,         # gP/gDOM
  Y.N1            = 0.13,         # gN1/gNH4-N
  Y.N2            = 0.03,         # gN2/gNO2-N
  Y.HET           = 0.6,          # gHET/gDOM
  Y.hyd           = 1.0,          # gDOM/gPOM

  # kinetic parameters
  k.gro.ALG       = 1.5,          # 1/d
  k.gro.HET       = 1.5,          # 1/d
  k.gro.N1        = 0.8,          # 1/d
  k.gro.N2        = 1.1,          # 1/d
  k.resp.ALG      = 0.10,         # 1/d
  k.resp.HET      = 0.20,         # 1/d
  k.resp.N1       = 0.10,         # 1/d
  k.resp.N2       = 0.10,         # 1/d
  k.death.ALG     = 0.10,         # 1/d
  k.death.HET     = 0.20,         # 1/d
  k.death.N1      = 0.10,         # 1/d
  k.death.N2      = 0.10,         # 1/d
  k.hyd.POM       = 0.5,          # 1/d
  K.HPO4.ALG      = 0.01,         # gP/m3
  K.HPO4.HET      = 0.01,         # gP/m3
  K.HPO4.nitri    = 0.02,         # gP/m3
  K.N.ALG         = 0.04,         # gN/m3
  K.N.HET         = 0.04,         # gN/m3
  p.NH4.ALG       = 5,            # -
  p.NH4.HET       = 5,            # -
  K.NH4.nitri     = 0.5,          # gN/m3
  K.NO2.nitri     = 0.5,          # gN/m3
  K.O2.nitri      = 0.5,          # gO/m3
  K.O2.ALG        = 0.5,          # gO/m3
  K.O2.HET        = 0.5,          # gO/m3
  K.DOM.HET       = 0.5,          # gDOM/m3
  beta.ALG        = 0.046,        # 1/degC
  beta.HET        = 0.046,        # 1/degC
  beta.N1         = 0.098,        # 1/degC
  beta.N2         = 0.069,        # 1/degC
  beta.hyd        = 0.046,        # 1/degC
  K.I             = 200,          # W/m2
  lambda          = 0.1,          # 1/m
  K2.O2           = 10,           # 1/d
  K.shadow.ALG    = 100,              # gDM/m2
```

```
  K.limit.HET     = 10,             # gDM/m2
  K.limit.nitri   = 5,            # gDM/m2

  # River geometry
  L               = 2000,         # m
  w               = 20,           # m
  h               = 0.5,          # m

  # Input and initial conditions
  Q.in            = 4,            # m3/s
  D.ALG.ini       = 50,           # gDM/m2
  D.HET.ini       = 20,           # gDM/m2
  D.N1.ini        = 2,            # gDM/m2
  D.N2.ini        = 1,            # gDM/m2
  D.POM.ini       = 50,           # gDM/m2
  C.HPO4.ini      = 0.4,          # gP/m3
  C.NH4.ini       = 0.4,          # gN/m3
  C.NO3.ini       = 4,            # gN/m3
  C.NO2.ini       = 0,            # gN/m3
  C.O2.ini        = 10,           # gO/m3
  C.DOM.ini       = 3,            # gDOM/m3
  C.HPO4.in       = 0.4,          # gP/m3
  C.NH4.in        = 0.4,          # gN/m3
  C.NO3.in        = 4,            # gN/m3
  C.O2.in         = 10,           # gO/m3
  C.DOM.in        = 3,            # gDOM/m3

  # environmental conditions
  T0              = 20,           # degC
  T.min           = 15,           # degC
  T.max           = 25,           # degC
  I0.max          = 600,          # W/m2
  t.max.I         = 0.5,          # d
  t.max.T         = 0.6,          # d
  p               = 101325)       # Pa
```

This set of parameters contains the elemental mass fractions of nitrogen, phosphorus, oxygen and hydrogen. The largest mass fraction is then calculated as the difference to unity:

```
# choose carbon fractions to guarantee that the fractions sum to unity:
param$alpha.C.ALG <- 1 - (param$alpha.O.ALG + param$alpha.H.ALG +
                            param$alpha.N.ALG + param$alpha.P.ALG)
param$alpha.C.BAC <- 1 - (param$alpha.O.BAC + param$alpha.H.BAC +
                            param$alpha.N.BAC + param$alpha.P.BAC)
param$alpha.C.POM <- 1 - (param$alpha.O.POM + param$alpha.H.POM +
                            param$alpha.N.POM + param$alpha.P.POM)
param$alpha.C.DOM <- 1 - (param$alpha.O.DOM + param$alpha.H.DOM +
                            param$alpha.N.DOM + param$alpha.P.DOM)
```

We define the ''yields" of algae and bacteria death processes as in exercise 4:

```
# choose yield of death to guarantee that no nutrients are required
# (oxygen content of POM was reduced to avoid need of oxygen):
param$Y.ALG.death <-  min(1,
                          param$alpha.N.ALG/param$alpha.N.POM,
                          param$alpha.P.ALG/param$alpha.P.POM,
                          param$alpha.C.ALG/param$alpha.C.POM)
param$Y.BAC.death <-  min(1,
                          param$alpha.N.BAC/param$alpha.N.POM,
                          param$alpha.P.BAC/param$alpha.P.POM,
                          param$alpha.C.BAC/param$alpha.C.POM)
print(param$Y.ALG.death)
print(param$Y.BAC.death)
```

Once all substances are defined, we can construct a composition matrix by passing the list of substances to the function `calc.comp.matrix` of the package `stoichcalc`:

```
# Construction of composition matrix:

NH4    <- c(H      = 4*1/14,             # gH/gNH4-N
            N      = 1,                  # gN/gNH4-N
            charge = 1/14)               # chargeunits/gNH4-N
NO2    <- c(O      = 2*16/14,            # gO/gNO2-N
            N      = 1,                  # gN/gNO2-N
            charge = -1/14)              # chargeunits/gNO2-N
NO3    <- c(O      = 3*16/14,            # gO/gNO3-N
            N      = 1,                  # gN/gNO3-N
            charge = -1/14)              # chargeunits/gNO3-N
HPO4   <- c(O      = 4*16/31,            # gO/gHPO4-P
            H      = 1*1/31,             # gH/gHPO4-P
            P      = 1,                  # gP/gHPO4-P
            charge = -2/31)              # chargeunits/gHPO4-P
HCO3   <- c(C      = 1,                  # gC/gHCO3-C
            O      = 3*16/12,            # gO/gHCO3-C
            H      = 1*1/12,             # gH/gHCO3-C
            charge = -1/12)              # chargeunits/gHCO3-C
O2     <- c(O      = 1)                  # gO/gO2-O
H      <- c(H      = 1,                  # gH/molH
            charge = 1)                  # chargeunits/molH
H2O    <- c(O      = 1*16,               # gO/molH2O
            H      = 2*1)                # gH/molH2O

ALG    <- c(C      = param$alpha.C.ALG,     # gC/gALG
            O      = param$alpha.O.ALG,     # gO/gALG
            H      = param$alpha.H.ALG,     # gH/gALG
            N      = param$alpha.N.ALG,     # gN/gALG
            P      = param$alpha.P.ALG)     # gP/gALG
HET    <- c(C      = param$alpha.C.BAC,     # gC/gBAC
            O      = param$alpha.O.BAC,     # gO/gBAC
            H      = param$alpha.H.BAC,     # gH/gBAC
            N      = param$alpha.N.BAC,     # gN/gBAC
            P      = param$alpha.P.BAC)     # gP/gBAC
N1     <- c(C      = param$alpha.C.BAC,     # gC/gBAC
            O      = param$alpha.O.BAC,     # gO/gBAC
```

```
            H       = param$alpha.H.BAC,     # gH/gBAC
            N       = param$alpha.N.BAC,     # gN/gBAC
            P       = param$alpha.P.BAC)     # gP/gBAC
N2      <- c(C      = param$alpha.C.BAC,     # gC/gBAC
            O       = param$alpha.O.BAC,     # gO/gBAC
            H       = param$alpha.H.BAC,     # gH/gBAC
            N       = param$alpha.N.BAC,     # gN/gBAC
            P       = param$alpha.P.BAC)     # gP/gBAC
POM     <- c(C      = param$alpha.C.POM,     # gC/gPOM
            O       = param$alpha.O.POM,     # gO/gPOM
            H       = param$alpha.H.POM,     # gH/gPOM
            N       = param$alpha.N.POM,     # gN/gPOM
            P       = param$alpha.P.POM)     # gP/gPOM
DOM     <- c(C      = param$alpha.C.DOM,     # gC/gDOM
            O       = param$alpha.O.DOM,     # gO/gDOM
            H       = param$alpha.H.DOM,     # gH/gDOM
            N       = param$alpha.N.DOM,     # gN/gDOM
            P       = param$alpha.P.DOM)     # gP/gDOM

subst.comp <- list(C.NH4    = NH4,
                   C.NO2    = NO2,
                   C.NO3    = NO3,
                   C.HPO4   = HPO4,
                   C.HCO3   = HCO3,
                   C.O2     = O2,
                   C.DOM    = DOM,
                   C.H      = H,
                   C.H2O    = H2O,
                   D.ALG    = ALG,
                   D.HET    = HET,
                   D.N1     = N1,
                   D.N2     = N2,
                   D.POM    = POM)

alpha <- calc.comp.matrix(subst.comp)
print(round(alpha,3))
```

Having defined the composition matrix and the stoichiometric parameters, we can define the stoichiometries of the processes to be considered by the model:

```
# Derivation of Process Stoichiometry:

# Growth of algae on ammonium:

nu.gro.ALG.NH4 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "gro.ALG.NH4",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.ALG"),
                   subst.norm  = "D.ALG",
                   nu.norm     = 1)
```

```
# Growth of algae on nitrate:

nu.gro.ALG.NO3 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "gro.ALG.NO3",
                   subst       = c("C.NO3","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.ALG"),
                   subst.norm  = "D.ALG",
                   nu.norm     = 1)

# Respiration of algae:

nu.resp.ALG <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "resp.ALG",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.ALG"),
                   subst.norm  = "D.ALG",
                   nu.norm     = -1)

# Death of algae:

nu.death.ALG <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "death.ALG",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.ALG","D.POM"),
                   subst.norm  = "D.ALG",
                   nu.norm     = -1,
                   constraints = list(c("D.ALG" = param$Y.ALG.death,
                                        "D.POM" = 1)))

# Growth of heterotrophic bacteria using ammonium:

nu.gro.HET.NH4 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "gro.HET.NH4",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.DOM","C.H","C.H2O","D.HET"),
                   subst.norm  = "D.HET",
                   nu.norm     = 1,
                   constraints = list(c("C.DOM" = param$Y.HET,
                                        "D.HET"  = 1)))

# Growth of heterotrophic bacteria using nitrate:

nu.gro.HET.NO3 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "gro.HET.NO3",
                   subst       = c("C.NO3","C.HPO4","C.HCO3","C.O2",
                                   "C.DOM","C.H","C.H2O","D.HET"),
                   subst.norm  = "D.HET",
                   nu.norm     = 1,
```

```
                  constraints = list(c("C.DOM" = param$Y.HET,
                                       "D.HET"  = 1)))

# Respiration of heterotrophic bacteria:

nu.resp.HET <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "resp.HET",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.HET"),
                   subst.norm  = "D.HET",
                   nu.norm     = -1)

# Death of heterotrophic bacteria:

nu.death.HET <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "death.HET",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.HET","D.POM"),
                   subst.norm  = "D.HET",
                   nu.norm     = -1,
                   constraints = list(c("D.HET" = param$Y.BAC.death,
                                        "D.POM" = 1)))

# Growth of first stage nitrifiers:

nu.gro.N1 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "gro.N1",
                   subst       = c("C.NH4","C.NO2","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.N1"),
                   subst.norm  = "D.N1",
                   nu.norm     = 1,
                   constraints = list(c("C.NH4" = param$Y.N1,
                                        "D.N1"  = 1)))

# Respiration of first stage nitrifiers:

nu.resp.N1 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "resp.N1",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.N1"),
                   subst.norm  = "D.N1",
                   nu.norm     = -1)

# Death of first stage nitrifiers:

nu.death.N1 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "death.N1",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
```

```
                                    "C.H","C.H2O","D.N1","D.POM"),
                  subst.norm  = "D.N1",
                  nu.norm     = -1,
                  constraints = list(c("D.N1"  = param$Y.BAC.death,
                                       "D.POM" = 1)))

# Growth of second stage nitrifiers:

nu.gro.N2 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "gro.N2",
                   subst       = c("C.NO2","C.NO3","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.N2"),
                   subst.norm  = "D.N2",
                   nu.norm     = 1,
                   constraints = list(c("C.NO2" = param$Y.N2,
                                        "D.N2"  = 1)))

# Respiration of second stage nitrifiers:

nu.resp.N2 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "resp.N2",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.N2"),
                   subst.norm  = "D.N2",
                   nu.norm     = -1)

# Death of second stage nitrifiers:

nu.death.N2 <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "death.N2",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.H","C.H2O","D.N2","D.POM"),
                   subst.norm  = "D.N2",
                   nu.norm     = -1,
                   constraints = list(c("D.N2"  = param$Y.BAC.death,
                                        "D.POM" = 1)))

# Hydrolysis of organic particles:

nu.hyd.POM <-
  calc.stoich.coef(alpha       = alpha,
                   name        = "hyd.POM",
                   subst       = c("C.NH4","C.HPO4","C.HCO3","C.O2",
                                   "C.DOM","C.H","C.H2O","D.POM"),
                   subst.norm  = "D.POM",
                   nu.norm     = -1,
                   constraints = list(c("D.POM" = param$Y.hyd,
                                        "C.DOM"  = 1)))
```

Finally, we bind the stoichiometries together for the full stoichiometric matrix:

```
# Combine process stoichiometries to stoichiometric matrix:

nu <- rbind(nu.gro.ALG.NH4,
            nu.gro.ALG.NO3,
            nu.resp.ALG,
            nu.death.ALG,
            nu.gro.HET.NH4,
            nu.gro.HET.NO3,
            nu.resp.HET,
            nu.death.HET,
            nu.gro.N1,
            nu.resp.N1,
            nu.death.N1,
            nu.gro.N2,
            nu.resp.N2,
            nu.death.N2,
            nu.hyd.POM)
print(round(nu,3))

# write.table(nu,file="river2_nu.dat",sep="\t",col.names=NA)
```

Now, we can proceed with the definition of transformation processes: **Complete** the missing equations for the expression of the rates based on Table 11.15.

```
# Definition of transformation processes:

# Growth of algae on ammonium:

gro.ALG.NH4 <-
  new(Class  = "process",
      name   = "gro.ALG.NH4",
      rate   = expression(k.gro.ALG
                          *exp(beta.ALG*(T-T0))
                          *I0*exp(-lambda*h)/(K.I+I0*exp(-lambda*h))
                          *min(C.HPO4/(K.HPO4.ALG+C.HPO4),
                               (C.NH4+C.NO3)/(K.N.ALG+C.NH4+C.NO3))
                          *(p.NH4.ALG*C.NH4/(p.NH4.ALG*C.NH4+C.NO3))
                          *D.ALG*K.shadow.ALG/(K.shadow.ALG+D.ALG)),
      stoich = as.list(nu["gro.ALG.NH4",]),
      pervol = F)

# Growth of algae on nitrate:

gro.ALG.NO3 <-
  new(Class  = "process",
      name   = "gro.ALG.NO3",
      rate   = expression(k.gro.ALG
                          *exp(beta.ALG*(T-T0))
                          *I0*exp(-lambda*h)/(K.I+I0*exp(-lambda*h))
                          *min(C.HPO4/(K.HPO4.ALG+C.HPO4),
                               (C.NH4+C.NO3)/(K.N.ALG+C.NH4+C.NO3))
                          *(C.NO3/(p.NH4.ALG*C.NH4+C.NO3))
                          *D.ALG*K.shadow.ALG/(K.shadow.ALG+D.ALG)),
```

```
      stoich = as.list(nu["gro.ALG.NO3",]),
      pervol = F)

# Respiration of algae:

resp.ALG <-
  new(Class  = "process",
      name   = "resp.ALG",
      rate   = expression(k.resp.ALG*exp(beta.ALG*(T-T0))
                          *C.O2/(K.O2.ALG+C.O2)*D.ALG),
      stoich = as.list(nu["resp.ALG",]),
      pervol = F)

# Death of algae:

death.ALG <-
  new(Class = "process",
      name   = "death.ALG",
      rate   = expression(k.death.ALG*D.ALG),
      stoich = as.list(nu["death.ALG",]),
      pervol = F)

# Growth of heterotrophic bacteria with ammonium:

gro.HET.NH4 <-
  new(Class  = "process",
      name   = "gro.HET.NH4",
      rate   = expression(k.gro.HET
                          *exp(beta.HET*(T-T0))
                          *min(C.DOM/(K.DOM.HET+C.DOM),
                               C.O2/(K.O2.HET+C.O2),
                               C.HPO4/(K.HPO4.HET+C.HPO4),
                               (C.NH4+C.NO3)/(K.N.HET+C.NH4+C.NO3))
                          *(p.NH4.HET*C.NH4/(p.NH4.HET*C.NH4+C.NO3))
                          *D.HET*K.limit.HET/(K.limit.HET+D.HET)),
      stoich = as.list(nu["gro.HET.NH4",]),
      pervol = F)

# Growth of heterotrophic bacteria with nitrate:

gro.HET.NO3 <-
  new(Class  = "process",
      name   = "gro.HET.NO3",
      rate   = expression(k.gro.HET
                          *exp(beta.HET*(T-T0))
                          *min(C.DOM/(K.DOM.HET+C.DOM),
                               C.O2/(K.O2.HET+C.O2),
                               C.HPO4/(K.HPO4.HET+C.HPO4),
                               (C.NH4+C.NO3)/(K.N.HET+C.NH4+C.NO3))
                          *(C.NO3/(p.NH4.HET*C.NH4+C.NO3))
                          *D.HET*K.limit.HET/(K.limit.HET+D.HET)),
      stoich = as.list(nu["gro.HET.NO3",]),
      pervol = F)
```

```
# Respiration of heterotrophic bacteria: TO BE COMPLETED

resp.HET <-
  new(Class  = "process",
      name   = "resp.HET",
      rate   = expression(...),
      stoich = as.list(nu["resp.HET",]),
      pervol = F)

# Death of heterotrophic bacteria:

death.HET <-
  new(Class = "process",
      name   = "death.HET",
      rate   = expression(k.death.HET*D.HET),
      stoich = as.list(nu["death.HET",]),
      pervol = F)

# Growth of first stage nitrifiers:

gro.N1 <-
  new(Class  = "process",
      name   = "gro.N1",
      rate   = expression(k.gro.N1
                          *exp(beta.N1*(T-T0))
                          *min(C.HPO4/(K.HPO4.nitri+C.HPO4),
                               C.NH4/(K.NH4.nitri+C.NH4),
                               C.O2/(K.O2.nitri+C.O2))
                          *D.N1*K.limit.nitri/(K.limit.nitri+D.N1)),
      stoich = as.list(nu["gro.N1",]),
      pervol = F)

# Respiration of first stage nitrifiers:

resp.N1 <-
  new(Class  = "process",
      name   = "resp.N1",
      rate   = expression(k.resp.N1*exp(beta.N1*(T-T0))
                          *C.O2/(K.O2.nitri+C.O2)*D.N1),
      stoich = as.list(nu["resp.N1",]),
      pervol = F)

# Death of first stage nitrifiers:

death.N1 <-
  new(Class = "process",
      name   = "death.N1",
      rate   = expression(k.death.N1*D.N1),
      stoich = as.list(nu["death.N1",]),
      pervol = F)

# Growth of second stage nitrifiers:
```

```
gro.N2 <-
  new(Class  = "process",
      name   = "gro.N2",
      rate   = expression(k.gro.N2
                          *exp(beta.N2*(T-T0))
                          *min(C.HPO4/(K.HPO4.nitri+C.HPO4),
                               C.NO2/(K.NO2.nitri+C.NO2),
                               C.O2/(K.O2.nitri+C.O2))
                          *D.N2*K.limit.nitri/(K.limit.nitri+D.N2)),
      stoich = as.list(nu["gro.N2",]),
      pervol = F)

# Respiration of second stage nitrifiers:

resp.N2 <-
  new(Class  = "process",
      name   = "resp.N2",
      rate   = expression(k.resp.N2*exp(beta.N2*(T-T0))
                          *C.O2/(K.O2.nitri+C.O2)*D.N2),
      stoich = as.list(nu["resp.N2",]),
      pervol = F)

# Death of second stage nitrifiers:

death.N2 <-
  new(Class = "process",
      name   = "death.N2",
      rate   = expression(k.death.N2*D.N2),
      stoich = as.list(nu["death.N2",]),
      pervol = F)

# Hydrolysis of POM: TO BE COMPLETED

hyd.POM <-
  new(Class  = "process",
      name   = "nu.hyd.POM",
      rate   = expression(...),
      stoich = as.list(nu["hyd.POM",]),
      pervol = F)
```

We now define the seasonally varying light intensity and temperature and, as a consequence of temperature variation, the seasonally varying saturation concentration of dissolved oxygen:

```
# Definition of environmental conditions:

cond <- list(I0       = expression(I0.max*0.5*(sign(cos(2*pi/1.0*(t-t.max.I)))+1)
                                   *cos(2*pi/1.0*(t-t.max.I))^2),
             T        = expression( 0.5*(T.min+T.max)
                                    +0.5*(T.max-T.min)
                                    *cos(2*pi/1.0*(t-t.max.T))),
             C.O2.sat = expression(exp(7.7117-1.31403*log(T+45.93))
                                   *p/101325))
```

We then define three reactors to approximate advective transport and roughly resolve longitudinal concen-

tration gradients:

```
# Definition of reactor:

# River Sections:

reach1 <-
  new(Class            = "reactor",
      name             = "R1",
      volume.ini       = expression(L*w*h),
      area             = expression(L*w),
      conc.pervol.ini  = list(C.HPO4 = expression(C.HPO4.ini),  # gP/m3
                              C.NH4  = expression(C.NH4.ini),   # gN/m3
                              C.NO2  = expression(C.NO2.ini),   # gN/m3
                              C.NO3  = expression(C.NO3.ini),   # gN/m3
                              C.O2   = expression(C.O2.ini),    # gN/m3
                              C.DOM  = expression(C.DOM.ini)),  # gDOM/m3
      conc.perarea.ini = list(D.ALG  = expression(D.ALG.ini),   # gALG/m2
                              D.HET  = expression(D.HET.ini),   # gHET/m2
                              D.N1   = expression(D.N1.ini),    # gN1/m2
                              D.N2   = expression(D.N2.ini),    # gN2/m2
                              D.POM  = expression(D.POM.ini)),  # gPOM/m2
      input            = list(C.O2   = expression(K2.O2*L*w*h
                                                  *(C.O2.sat-C.O2))), # gas exchange
      inflow           = expression(Q.in*86400),                # m3/d
      inflow.conc      = list(C.HPO4 = expression(C.HPO4.in),
                              C.NH4  = expression(C.NH4.in),
                              C.NO3  = expression(C.NO3.in),
                              C.O2   = expression(C.O2.sat),
                              C.DOM  = expression(C.DOM.in)),
      processes        = list(gro.ALG.NH4,gro.ALG.NO3,resp.ALG,death.ALG,
                              gro.HET.NH4,gro.HET.NO3,resp.HET,death.HET,
                              gro.N1,resp.N1,death.N1,gro.N2,resp.N2,death.N2,
                              hyd.POM))

reach2 <-
  new(Class            = "reactor",
      name             = "R2",
      volume.ini       = expression(L*w*h),
      area             = expression(L*w),
      conc.pervol.ini  = list(C.HPO4 = expression(C.HPO4.ini),  # gP/m3
                              C.NH4  = expression(C.NH4.ini),   # gN/m3
                              C.NO2  = expression(C.NO2.ini),   # gN/m3
                              C.NO3  = expression(C.NO3.ini),   # gN/m3
                              C.O2   = expression(C.O2.ini),    # gN/m3
                              C.DOM  = expression(C.DOM.ini)),  # gDOM/m3
      conc.perarea.ini = list(D.ALG  = expression(D.ALG.ini),   # gALG/m2
                              D.HET  = expression(D.HET.ini),   # gHET/m2
                              D.N1   = expression(D.N1.ini),    # gN1/m2
                              D.N2   = expression(D.N2.ini),    # gN2/m2
                              D.POM  = expression(D.POM.ini)),  # gPOM/m2
      input            = list(C.O2   = expression(K2.O2*L*w*h
                                                  *(C.O2.sat-C.O2))), # gas exchange
      processes        = list(gro.ALG.NH4,gro.ALG.NO3,resp.ALG,death.ALG,
```

```
                              gro.HET.NH4,gro.HET.NO3,resp.HET,death.HET,
                              gro.N1,resp.N1,death.N1,gro.N2,resp.N2,death.N2,
                              hyd.POM))

reach3 <-
  new(Class            = "reactor",
      name             = "R3",
      volume.ini       = expression(L*w*h),
      area             = expression(L*w),
      conc.pervol.ini  = list(C.HPO4 = expression(C.HPO4.ini),  # gP/m3
                              C.NH4  = expression(C.NH4.ini),   # gN/m3
                              C.NO2  = expression(C.NO2.ini),   # gN/m3
                              C.NO3  = expression(C.NO3.ini),   # gN/m3
                              C.O2   = expression(C.O2.ini),    # gN/m3
                              C.DOM  = expression(C.DOM.ini)),  # gDOM/m3
      conc.perarea.ini = list(D.ALG  = expression(D.ALG.ini),   # gALG/m2
                              D.HET  = expression(D.HET.ini),   # gHET/m2
                              D.N1   = expression(D.N1.ini),    # gN1/m2
                              D.N2   = expression(D.N2.ini),    # gN2/m2
                              D.POM  = expression(D.POM.ini)),  # gPOM/m2
      input            = list(C.O2   = expression(K2.O2*L*w*h
                                                  *(C.O2.sat-C.O2))), # gas ex.
      outflow          = expression(Q.in*86400),
      processes        = list(gro.ALG.NH4,gro.ALG.NO3,resp.ALG,death.ALG,
                              gro.HET.NH4,gro.HET.NO3,resp.HET,death.HET,
                              gro.N1,resp.N1,death.N1,gro.N2,resp.N2,death.N2,
                              hyd.POM))
```

We then define advective links to combine the three reactors: **Complete** the missing definition of the link from reach 2 to reach 3 based on the code below and chapter 16 of the manuscript.

```
# Definition of links:

# Link from reach 1 to reach 2:

reach1.reach2 <-
  new(Class      = "link",
      name       = "R1R2",
      from       = "R1",
      to         = "R2",
      flow       = expression(Q.in*86400))

# Link from reach 2 to reach 3: TO BE COMPLETED

reach2.reach3 <- ...
```

The definition of the model is concluded with the system definition that contains the reactors, links, environmental conditions, parameters and output time points for simulations:

```
# Definition of system:

# River system:
```

```
system.11.6 <- new(Class    = "system",
                   name     = "River",
                   reactors = list(reach1,reach2,reach3),
                   links    = list(reach1.reach2,reach2.reach3),
                   cond     = cond,
                   param    = param,
                   t.out    = seq(0,3,by=0.02))      #t.out=seq(0,10,by=0.02),
```

Task 3: Model results

Peform a simulation of the model and try to understand the time courses of the state variables and the overall mass fluxes of phosphorus and nitrogen compounds.

Complete the code below to start the simulation.

```
# Perform simulation:

res.11.6 <- calcres(...) # TO BE COMPLETED
```

We then plot the results to the screen and write them also with different options to files. Please note that you have to adjust the size of the pdf files with the `width` and `height` arguments to have reasonable sizes of the legend and the descriptions of the axes. You may also want to plot multiple lines into a single plot by specifying the argument `colnames` of the function `plotres` (you may type `?plotres` to get a description of the arguments of this function).

```
# Plot results:
#plotres(res.11.6)              # default plots

# plot to screen:

plotres(res.11.6,colnames=list(c("C.NH4.R1", "C.NH4.R2", "C.NH4.R3"),
                               c("C.NO2.R1", "C.NO2.R2", "C.NO2.R3"),
                               c("C.NO3.R1", "C.NO3.R2", "C.NO3.R3"),
                               c("C.HPO4.R1","C.HPO4.R2","C.HPO4.R3"),
                               c("C.O2.R1",  "C.O2.R2",  "C.O2.R3"),
                               c("C.DOM.R1", "C.DOM.R2", "C.DOM.R3"),
                               c("D.ALG.R1", "D.ALG.R2", "D.ALG.R3"),
                               c("D.HET.R1", "D.HET.R2", "D.HET.R3"),
                               c("D.N1.R1",  "D.N1.R2",  "D.N1.R3"),
                               c("D.N2.R1",  "D.N2.R2",  "D.N2.R3"),
                               c("D.POM.R1", "D.POM.R2", "D.POM.R3")))

# plot to file:

plotres(res.11.6,colnames=list(c("C.NH4.R1", "C.NH4.R2", "C.NH4.R3"),
                               c("C.NO2.R1", "C.NO2.R2", "C.NO2.R3"),
                               c("C.NO3.R1", "C.NO3.R2", "C.NO3.R3"),
                               c("C.HPO4.R1","C.HPO4.R2","C.HPO4.R3"),
                               c("C.O2.R1",  "C.O2.R2",  "C.O2.R3"),
                               c("C.DOM.R1", "C.DOM.R2", "C.DOM.R3"),
                               c("D.ALG.R1", "D.ALG.R2", "D.ALG.R3"),
                               c("D.HET.R1", "D.HET.R2", "D.HET.R3"),
                               c("D.N1.R1",  "D.N1.R2",  "D.N1.R3"),
```

```
                              c("D.N2.R1",  "D.N2.R2",  "D.N2.R3"),
                              c("D.POM.R1", "D.POM.R2", "D.POM.R3")),
       file     = "exercise_5_results.pdf",
       width    = 12,
       height   = 9)
```

Task 4: Homework: Model sensitivity

What happens in the model if a herbicide enters the water and inhibits the growth of algae? What happens if a bactericide enters the water and inhibits the growth of a specific bacteria group (heterotrophic bacteria, step one nitrifiers, N1, or step two nitrifiers, N2)? Answer these questions by modifying the related kinetic parameters and interpret the changes in the simulation results.

Theory questions

1. What is the spatial arrangement of the reactors in the river model compared to the previous lake model? How is the advective flow implemented?
2. What is the difference between planktonic primary production (modelled in the previous lake models) and benthic primary production (modelled in this river model)?
3. If you have to develop a river water quality model, how do you decide if you model nitrification as a one step process or a two step process (as in section 8.6) or if you model the nitrifying bacteria explicitly (as in section 8.8.2)?
4. What do you have to consider when deciding about the number and length of the boxes to discretize a river in a model like this one? Hint: Have a look at transport processes described in section 6.1.2.4.
5. What is the relative importance of advective transport and transformation processes for concentrations in the reactors?