Exercise 7: Stochasticity and Uncertainty

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

May 21, 2025

Goals:

- Understand the Ornstein-Uhlenbeck process, the meaning of its parameters, how to use it to describe environmental stochasticity in driving forces or its effect on parameters, and how to propagate this stochasticity to model outputs.
- Be able to consider environmental stochasticity and parameter uncertainty in modelling by propagating it through the simple lake plankton model using `ecosim`.

Task 1: Simulation of Ornstein-Uhlenbeck processes

Study the implementation of drawing samples from Ornstein-Uhlenbeck processes using the function `randou` of the R package `ecosim` and interpret the behaviour of the solutions.

```
# load required packages:

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

An Ornstein-Uhlenbeck process is defined by the mean, `mean`, asymptotic standard deviation, `sd`, and correlation time, `tau`. The code below can plot three realizations of different parameters.

Note that the third process is defined that the log of the plotted time series is an Ornstein-Uhlenbeck process. Still, the mean and the asymptotic standard deviation are defined at the non-log scale.

Experiment with different parameters values.

```
# Task 1:    ######################################################################

# Simulation of Ornstein-Uhlenbeck processes

# Draw from Ornstein-Uhlenbeck processes:

# Define time domain:

t <- 0:365
plot(numeric(0),numeric(0),xlim=c(0,365),ylim=c(0,20),xaxs="i",yaxs="i",
     xlab="t",ylab=expression(theta),main="Realizations of Ornstein-Uhlenbeck Processes")

mean <- 16
sd   <- 0.5
tau  <- 50
abline(h=mean,lty="dotted")
lines(randou(mean=mean,sd=sd,tau=tau,t=t),lty="solid",col="black")
```

```
lines(randou(mean=mean,sd=sd,tau=tau,t=t),lty="dashed",col="blue")
lines(randou(mean=mean,sd=sd,tau=tau,t=t),lty="dotdash",col="red")

mean <- 10
sd   <- 1
tau  <- 5
abline(h=mean,lty="dotted")
lines(randou(mean=mean,sd=sd,tau=tau,t=t),lty="solid",col="black")
lines(randou(mean=mean,sd=sd,tau=tau,t=t),lty="dashed",col="blue")
lines(randou(mean=mean,sd=sd,tau=tau,t=t),lty="dotdash",col="red")

mean <- 1
sd   <- 1
tau  <- 10
abline(h=mean,lty="dotted")
lines(randou(mean=mean,sd=sd,tau=tau,t=t,log=TRUE),lty="solid",col="black")
lines(randou(mean=mean,sd=sd,tau=tau,t=t,log=TRUE),lty="dashed",col="blue")
lines(randou(mean=mean,sd=sd,tau=tau,t=t,log=TRUE),lty="dotdash",col="red")
```

Task 2: Simulation of the lake plankton model with constant driving forces with consideration of environmental stochasticity

Study the implementation of the model that considers the effect of making four key parameters stochastic. How does this stochasticity affect the model outputs?

Note that the model definition is copied from exercise 2. We first define a named list of model parameters:

```
# Task 2:    ########################################################################

# Simulation of the lake plankton model with constant driving forces with
# consideration of environmental stochasticity
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


# definition of model parameters:

param.mean <- list(k.gro.ALG   = 0.5,      # 1/d
                   k.gro.ZOO   = 0.4,      # m3/gDM/d
                   k.death.ALG = 0.1,      # 1/d
                   k.death.ZOO = 0.05,     # 1/d
                   K.HPO4      = 0.002,    # gP/m3
                   Y.ZOO       = 0.2,      # gDM/gDM
                   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
                   C.ZOO.ini   = 0.1)      # gDM/m3
```

Next, we define the processes of growth and death of algae and zooplankton as objects of the class `process` of the package `ecosim`.

```
# definition of transformation processes

# growth of algae:

# 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

# growth of zooplankton:

gro.ZOO   <- new(Class  = "process",
                 name   = "Growth of zooplankton",
                 rate   = expression(k.gro.ZOO
                                     *C.ALG
                                     *C.ZOO),
                 stoich = list(C.ZOO  = expression(1),              # gDM/gDM
                               C.ALG  = expression(-1/Y.ZOO)))      # gP/gDM

# death of zooplankton:

death.ZOO <- new(Class  = "process",
                 name   = "Death of zooplankton",
                 rate   = expression(k.death.ZOO*C.ZOO),
                 stoich = list(C.ZOO  = 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
                              C.ZOO  = expression(C.ZOO.ini)),     # gDM/m3
      inflow           = expression(Q.in*86400),                   # m3/d
```

```
      inflow.conc      = list(C.HPO4 = expression(C.HPO4.in),
                              C.ALG  = 0,
                              C.ZOO  = 0),
      outflow          = expression(Q.in*86400),
      processes        = list(gro.ALG,death.ALG,gro.ZOO,death.ZOO))
```

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

First, we redo the simulations with constant parameters with absence or presence of zooplankton:

```
# Redo simulations without stochasticity and uncertainty:

system@param["C.ZOO.ini"] <- 0  # note that this reduces the model to model 91
res <- calcres(system)
plotres(res=res,colnames=list("C.HPO4",c("C.ALG", "C.ZOO")))

system@param <- param.mean
res <- calcres(system)
plotres(res=res,colnames=list("C.HPO4",c("C.ALG", "C.ZOO")))
```

To consider stochasticity in the parameters `k.gro.ALG`, `k.gro.ZOO`, `k.death.ALG`, `k.death.ZOO`, and `K.HPO4`, we first define a relative standard deviation of the parameters (for simplicity, we use the same for all parameters), a correlation time, tau, and the names of the parameters to be made stochastic:

```
# Define:

sd.rel  <- 0.1
tau      <- 5
names    <- c("k.gro.ALG","k.gro.ZOO","k.death.ALG","k.death.ZOO","K.HPO4")
```

We then define data structures to collect the results and loop over the sampling iterations. Within each iteration, we sample from the stochastic parameters, assign them to the model system, and perform simulations without and with zooplankton. In the first iteration, we plot the sampled parameters to illustrate their behavior and we print out the simulation time.

```
# Loop over Monte Carlo sampling and calculation of results:

sampsize.stoch <-     5 # number of samples

res.envsto  <- list()
res.envsto0 <- list()
par.def <- par(no.readonly=TRUE)
par(mfrow=c(2,3))
```

```
for ( i in 1:sampsize.stoch )
{
  # Make selected parameters stochastic in time:

  start.time <- proc.time()
  param <- param.mean
  for ( name in names )
  {
    mean <- param[[name]]
    param[[name]] <- randou(mean = mean,
                            sd   = sd.rel*mean,
                            tau  = tau,
                            t    = seq(0,365,by=1),
                            log  = TRUE)
    if ( i == 1 )
    {
      plot(param[[name]],type="l",xlab="t",ylab=name,ylim=c(0,2*mean))
      abline(h=mean,lty="dotted")
    }
  }
  system@param <- param
  res.envsto[[i]] <- calcres(system)
  system@param["C.ZOO.ini"] <- 0
  res.envsto0[[i]] <- calcres(system)
  if ( i == 1 )
  {
    print("computation time simple model, stochastic parameters")
    print(proc.time() - start.time)
  }
}
par(par.def)
```

Note that, due to the fluctuations in the parameter, the integration algorithm has to take much shorter steps than with constant parameters, which results in a strong increase in the simulation time.

Finally, we plot the results:

```
# Plot results:

plotres(res      = res.envsto0,
        colnames = list("C.HPO4",c("C.ALG","C.ZOO")))
plotres(res      = res.envsto,
        colnames = list("C.HPO4",c("C.ALG","C.ZOO")))
```

Task 3: Simulation of the lake plankton model with constant driving forces with consideration of parameter uncertainty

Study the implementation of the model that considers the effect of uncertainty of four key parameters of the model and the propagation of this uncertainty to model outputs.

The model is already defined from task 2 and we use the same parameters of the stochastic processes. For this reason, we can directly pass on to the loop over the Monte Carlo samples. The difference to task 2 is that within this loop, we draw constant parameters, rather than stochastic parameters to propagate their uncertainty through the model.

```
# Task 3:    ########################################################################

# Simulation of the lake plankton model with constant driving forces with
# consideration of parameter uncertainty
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

sampsize.unc <- 20 # number of samples

res.parunc  <- list()
res.parunc0 <- list()
for ( i in 1:sampsize.unc )
{
  # Make selected parameters uncertain:
  # -----------------------------------

  start.time <- proc.time()
  param <- param.mean
  for ( name in names )
  {
    mean <- param[[name]]
    param[[name]] <- randnorm(mean = mean,
                              sd   = sd.rel*mean,
                              log  = TRUE)
  }
  system@param <- param
  res.parunc[[i]] <- calcres(system)
  system@param["C.ZOO.ini"] <- 0
  res.parunc0[[i]] <- calcres(system)
  if ( i == 1 )
  {
    print("computation time simple model, uncertain parameters")
    print(proc.time() - start.time)
  }
}
```

Note the much shorter simulation time due to the use of constant parameters.

Finally, we plot again the results.

```
plotres(res      = res.parunc0,
        colnames = list("C.HPO4",c("C.ALG","C.ZOO")))
plotres(res      = res.parunc,
        colnames = list("C.HPO4",c("C.ALG","C.ZOO")))
```

Theory questions

1. What is the difference between environmental stochasticity and uncertainty about model parameters, both from a conceptual and from an implementation point of view?
2. What are the different characteristics of the three Ornstein-Uhlenbeck processes simulated in task 1? Can you guess the parameter values from the samples?
3. Why is it important to use transformed Ornstein-Uhlenbeck processes that lead to asymptotic lognormal distributions?