Exercise 3: General solution of stoichiometric equations

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

18.03.2020

Goals:

- Understand the concepts of stoichiometric calculations introduced in section 4.3.
- Be able to do stoichiometric calculations based on the general solution of stoichiometric equations (section 4.3.3) using the R package `stoichcalc`.
- Learn to be attentive to the results and to error messages and to correct the code accordingly (execute the code piece by piece and proceed only if the results are as expected and everything was correct).

Task 1: Simple stoichiometry using the `stoichcalc`-package

Study the implementation of the simple stoichiometry of algae growth as formulated in exercise 1 based on the `stoichcalc`-package. Run the implementation using the accompanying file `exercise_3.R`.

Task 2: Stoichiometry of a complex process model

Study the implementation of the stoichiometry of a process model of algae and zooplankton growth, respiration and death given below. Run the implementation based on the accompanying file `exercise_3.R`.

Task 3: Homework: Extend the process stoichiometry to sulfur

Extend the process model implemented in task 2 to the consideration of sulfur (S), assuming the same content of 0.3% of S in algae, zooplankton and particulate organic matter (decrease the carbon content by 0.3%) and the uptake and release of sulfur in the form of sulfate SO_4^{2-} (release may also be in the form of hydrogen sulfide, H_2S, which later is oxidized to sulfate; we aggregate these processes).

Questions

1. How do you find out, whether you need additional constraints to elemental mass balance and charge? What could be a drawback of the simplified approach to this question?
2. Where to get the required additional constraints from (if needed)?
3. Why do we add H^+, but not OH^- to the compounds considered for the calculation of stoichiometric coefficients?

Solution of Task 1

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

```
# load required packages:

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

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

Next we construct a composition matrix by defining substances as named vectors with the elemental content, build a list of all substances, and call the `stoichcalc` function `calc.comp.matrix` to compile the composition matrix. Note that you have to be very careful to choose units consistently across the model (we will feed the calculated stoichometric coefficients into processes in `ecosim` in the next exercise).

```
# Construct composition matrix:

HPO4   <- c(P      = 1)                  # gP/gHPO4-P
ALG    <- c(P      = 0.01)               # gP/gALG

subst.comp <- list(HPO4   = HPO4,
                   ALG    = ALG)

alpha.1 <- calc.comp.matrix(subst.comp)
```

```
## [1] "Composition matrix (1x2) successfully constructed:"
## [1] "el. const.: P"
## [1] "substances: HPO4,ALG"
```

```
print(alpha.1)
```

```
##   HPO4  ALG
## P    1 0.01
```

Alternatively, it would be possible to formulate the composition matrix directly (we will only need the composition matrix and the involved substances to calculate stoichiometric coefficients for a given process). The procedure shown above has the advantage that one only has to make specifications for elements that are contained in a substance and not to specify explicitly that the content of other elements is zero. This will then be done by the function `calc.comp.matrix`.

We then calculate process stoichiometries for growth and respiration of algae by providing the composition matrix, specifying a process name, providing the names of the substances involved in the process, and specifying which stoichiometric coefficient is set to a specific value (recall that process stoichiometry is only unique up to a multiplicative factor and that we make it unique by setting one of the coefficients to +1 or -1; this makes the process rate equal to plus or minus the conversion rate of the associated substance).

```
# Calculate stoichiometric coefficients of selected processes:

nu.gro.ALG <-
   calc.stoich.coef(alpha       = alpha.1,
                    name        = "growth.ALG",
                    subst       = c("HPO4","ALG"),
                    subst.norm  = "ALG",
                    nu.norm     = 1)
```

```
## [1] "Number of substances:                    2"
## [1] "Number of elementary constituents:       1"
## [1] "Number of constraints:                   0"
```

```
## [1] "Number of independent processes:           1"
## [1] "Stoichiometric coefficients successfully calculated."
```

```
nu.resp.ALG <-
   calc.stoich.coef(alpha       = alpha.1,
                    name        = "resp.ALG",
                    subst       = c("HPO4","ALG"),
                    subst.norm  = "ALG",
                    nu.norm     = -1)
```

```
## [1] "Number of substances:                      2"
## [1] "Number of elementary constituents:         1"
## [1] "Number of constraints:                     0"
## [1] "Number of independent processes:           1"
## [1] "Stoichiometric coefficients successfully calculated."
```

Note the message that the stoichiometric coefficients have been calculated successfully.

Finally, we bind the stoichiometric coefficients of the different processes together to the stoichiometric matrix:

```
nu.1 <- rbind(nu.gro.ALG,
              nu.resp.ALG)

print(nu.1)
```

```
##             HPO4 ALG
## growth.ALG -0.01   1
## resp.ALG    0.01  -1
```

Note that the stoichiometry of respiration is just the negative of that of primary production.

Solution of Task 2

For more complicated stoichiometries, it is worth to define composition parameters explicitly. This makes it much easier to adapte the composition later by changing the parameters and re-evaluating the stoichiometry. Note that the parameters are evaluate for calculating the stoichiometry and not carried-through as parameters. Changing the parameter without re-evaluating the stoichiometry does thus not have the desired effect.

We first define the mass fractions of oxygen, hydrogen, nitrogen and phosphorus for all organic substances:

```
# Set parameter values:

param <- list(a.O.ALG     = 0.50,         # gO/gALG
              a.H.ALG     = 0.07,         # gH/gALG
              a.N.ALG     = 0.06,         # gN/gALG
              a.P.ALG     = 0.005,        # gP/gALG
              a.O.ZOO     = 0.50,         # gO/gZOO
              a.H.ZOO     = 0.07,         # gH/gZOO
              a.N.ZOO     = 0.06,         # gN/gZOO
              a.P.ZOO     = 0.01,         # gP/gZOO
              a.O.POM     = 0.40,         # gO/gPOM
              a.H.POM     = 0.07,         # gH/gPOM
              a.N.POM     = 0.04,         # gN/gPOM
              a.P.POM     = 0.007,        # gP/gPOM
              Y.ZOO       = 0.2,          # gZOO/gALG
              f.e         = 0.4)          # gPOM/gALG
```

Then we calculate the carbon content as the difference to unity. It makes sense to do this for carbon as this is the larges mass fractions that does not bear the risk of becoming negative (if you would do it for phosphorus which is only about 1% of the mass or even less, one could easily get into negative values when modifying carbon that is about 50% of the mass).

```
# choose carbon fractions to guarantee that the fractions sum to unity:

param$a.C.ALG = 1-(param$a.O.ALG+param$a.H.ALG+param$a.N.ALG+param$a.P.ALG)
param$a.C.ZOO = 1-(param$a.O.ZOO+param$a.H.ZOO+param$a.N.ZOO+param$a.P.ZOO)
param$a.C.POM = 1-(param$a.O.POM+param$a.H.POM+param$a.N.POM+param$a.P.POM)
```

This completes the definition of the mass fractions.

Next we define the composition of all substances to be considered for the calculation of the stoichiometries of the growth, respiration and death processes. Note that we have to be careful with the units. The natural unit to measure the amount of inorganic chemical compounds is moles. This is not the case for organic compounds as they consist of a large number of different organic molecules and we just consider their mean composition. For this reason, we measure organic compounds as dry mass. As we have to consider conservation of charge as well as conservation of the elements, we need to add charge to the composition vectors:

```
# Construct composition matrix:

NH4    <- c(H      = 4,                 # molH/molNH4
            N      = 1,                 # molN/molNH4
            charge = 1)                 # chargeunits/molNH4
NO3    <- c(O      = 3,                 # molO/molNO3
            N      = 1,                 # molN/molNO3
            charge = -1)                # chargeunits/molNO3
HPO4   <- c(O      = 4,                 # molO/molHPO4
            H      = 1,                 # molH/molHPO4
            P      = 1,                 # molP/molHPO4
```

```
            charge = -2)                  # chargeunits/molHPO4
HCO3    <- c(C      = 1,                  # molC/molHCO3
             O      = 3,                  # molO/molHCO3
             H      = 1,                  # molH/molHCO3
             charge = -1)                 # chargeunits/molHCO3
O2      <- c(O      = 2)                  # molO/molO2
H       <- c(H      = 1,                  # molH/molH
             charge = 1)                  # chargeunits/molH
H2O     <- c(O      = 1,                  # molO/molH2O
             H      = 2)                  # molH/molH2O
ALG     <- c(C      = param$a.C.ALG/12,   # molC/gALG
             O      = param$a.O.ALG/16,   # molO/gALG
             H      = param$a.H.ALG,      # molH/gALG
             N      = param$a.N.ALG/14,   # molN/gALG
             P      = param$a.P.ALG/31)   # molP/gALG
ZOO     <- c(C      = param$a.C.ZOO/12,   # molC/gZOO
             O      = param$a.O.ZOO/16,   # molO/gZOO
             H      = param$a.H.ZOO,      # molH/gZOO
             N      = param$a.N.ZOO/14,   # molN/gZOO
             P      = param$a.P.ZOO/31)   # molP/gZOO
POM     <- c(C      = param$a.C.POM/12,   # molC/gPOM
             O      = param$a.O.POM/16,   # molO/gPOM
             H      = param$a.H.POM,      # molH/gPOM
             N      = param$a.N.POM/14,   # molN/gPOM
             P      = param$a.P.POM/31)   # molP/gPOM
```

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`:

```
subst.comp <- list(NH4    = NH4,
                   NO3    = NO3,
                   HPO4   = HPO4,
                   HCO3   = HCO3,
                   O2     = O2,
                   H      = H,
                   H2O    = H2O,
                   ALG    = ALG,
                   ZOO    = ZOO,
                   POM    = POM)

alpha.2 <- calc.comp.matrix(subst.comp)
```

```
## [1] "Composition matrix (6x10) successfully constructed:"
## [1] "el. const.: H,N,charge,O,P,C"
## [1] "substances: NH4,NO3,HPO4,HCO3,O2,H,H2O,ALG,ZOO,POM"
```

```
print(alpha.2)
```

```
##        NH4 NO3 HPO4 HCO3 O2 H H2O          ALG          ZOO          POM
## H        4   0    1    1  0 1   2 0.0700000000 0.0700000000 0.0700000000
## N        1   1    0    0  0 0   0 0.0042857143 0.0042857143 0.0028571429
## charge   1  -1   -2   -1  0 1   0 0.0000000000 0.0000000000 0.0000000000
## O        0   3    4    3  2 0   1 0.0312500000 0.0312500000 0.0250000000
## P        0   0    1    0  0 0   0 0.0001612903 0.0003225806 0.0002258065
## C        0   0    0    1  0 0   0 0.0304166667 0.0300000000 0.0402500000
```

Note that this function just complemented the definitions specified above for all substances by zeros for those elements that were not explicitly listed in the defintion.

Death converts living algae or zooplankton into dead particulate organic matter. The typical stoichiometry of this process would be -1 for algae or zooplankton and +1 for particulate organic matter. However, this only works if algae, zooplankton and dead particulate organic matter have the same composition. If the living organisms have not the same composition, we would need to introduce a state variable for dead organic particles with the corresponding composition for each group of living organisms. To avoid this increase in the number of state variables (in particular if there would be more than two organism groups to distinguish) and as most of the dead organic particles will be mineralized later anyway, we introduced a "yield" for death so that part of the organism becomes dead particulate organic matter and the remaining part is mineralized during the death process. The idea is to keep this latter fraction as small as possible while still being able to keep a reasonable stoichiometry. Our criteria are that we do not need an uptake of nitrogen or phosphorus for dying. This leads to the following expressions for the yield:

```
# choose yield of death to guarantee that no nutrients are required
# (oxygen content of POM was reduced to avoid need of oxygen):

param$Y.ALG.death = min(1,param$a.N.ALG/param$a.N.POM,param$a.P.ALG/param$a.P.POM)
param$Y.ZOO.death = min(1,param$a.N.ZOO/param$a.N.POM,param$a.P.ZOO/param$a.P.POM)

print(param$Y.ALG.death)
```

```
## [1] 0.7142857
```

```
print(param$Y.ZOO.death)
```

```
## [1] 1
```

If the yields would deviate strongly from unity, this approach would not make sense and we would have to switch to keeping multiple dead organic compounds of different composition in our model. The problem with algae is here that they contain less phosphorus than dead organic particles that represent a mixture of dead algae and dead zooplankton. It is typical in phosphorus-limited lakes during summer that the algae grow with a lower than usual phosphorus content. This can lead to the problem that the yield gets much lower than 1, which could be resolved by keeping dead algae and dead zooplanktion in the model.

Having defined the composition matrix, we can define the stoichiometries of the processes to be considered by the model. Each process requires the composition matrix alpha, it has a name, a list of substances to be considered, an indication of which coefficient should be chosen to make the stoichiometry unique, the value to which this coefficient is set (typically +1 or -1), and, optionally, the coefficients of additional constraints. These coefficients have to be provided in the form $\sum_j \nu_j \gamma_j = 0$. This means that e.g. the condition $\nu_{\mathrm{ZOO}} = -Y\nu_{ALG}$ indicating that one mass unit of algae is only converted into Y (= yield) mass units of zooplanktion has to be transformed into $\nu_{\mathrm{ZOO}} + Y\nu_{ALG} = 0$ to read the coefficients (see the first constraint below for the growth of zooplankton).

```
# Calculate stoichiometric coefficients of selected processes:

nu.gro.ALG.NH4 <-
   calc.stoich.coef(alpha       = alpha.2,
                    name        = "gro.ALG.NH4",
                    subst       = c("NH4","HPO4","HCO3","O2","H","H2O","ALG"),
                    subst.norm  = "ALG",
                    nu.norm     = 1)
```

```
## [1] "Number of substances:                    7"
## [1] "Number of elementary constituents:       6"
## [1] "Number of constraints:                   0"
## [1] "Number of independent processes:         1"
```

```
## [1] "Stoichiometric coefficients successfully calculated."
```

```
nu.gro.ALG.NO3 <-
   calc.stoich.coef(alpha       = alpha.2,
                    name        = "gro.ALG.NO3",
                    subst       = c("NO3","HPO4","HCO3","O2","H","H2O","ALG"),
                    subst.norm  = "ALG",
                    nu.norm     = 1)
```

```
## [1] "Number of substances:                    7"
## [1] "Number of elementary constituents:       6"
## [1] "Number of constraints:                   0"
## [1] "Number of independent processes:         1"
## [1] "Stoichiometric coefficients successfully calculated."
```

```
nu.resp.ALG <-
   calc.stoich.coef(alpha       = alpha.2,
                    name        = "resp.ALG",
                    subst       = c("NH4","HPO4","HCO3","O2","H","H2O","ALG"),
                    subst.norm  = "ALG",
                    nu.norm     = -1)
```

```
## [1] "Number of substances:                    7"
## [1] "Number of elementary constituents:       6"
## [1] "Number of constraints:                   0"
## [1] "Number of independent processes:         1"
## [1] "Stoichiometric coefficients successfully calculated."
```

```
nu.death.ALG <-
   calc.stoich.coef(alpha       = alpha.2,
                    name        = "death.ALG",
                    subst       = c("NH4","HPO4","HCO3","O2","H","H2O","ALG","POM"),
                    subst.norm  = "ALG",
                    nu.norm     = -1,
                    constraints = list(c("ALG" = param$Y.ALG.death,
                                         "POM" = 1)))
```

```
## [1] "Number of substances:                    8"
## [1] "Number of elementary constituents:       6"
## [1] "Number of constraints:                   1"
## [1] "Number of independent processes:         1"
## [1] "Stoichiometric coefficients successfully calculated."
```

```
nu.gro.ZOO <-
   calc.stoich.coef(alpha       = alpha.2,
                    name        = "groZOO",
                    subst       = c("NH4","HPO4","HCO3","O2","H","H2O","ALG","ZOO","POM"),
                    subst.norm  = "ZOO",
                    nu.norm     = 1,
                    constraints = list(c("ZOO" = 1,
                                         "ALG" = param$Y.ZOO),
                                       c("POM" = 1,
                                         "ALG" = param$f.e)))
```

```
## [1] "Number of substances:                    9"
## [1] "Number of elementary constituents:       6"
## [1] "Number of constraints:                   2"
```

```
## [1] "Number of independent processes:           1"
## [1] "Stoichiometric coefficients successfully calculated."
```

```
nu.resp.ZOO <-
   calc.stoich.coef(alpha       = alpha.2,
                    name        = "resp.ZOO",
                    subst       = c("NH4","HPO4","HCO3","O2","H","H2O","ZOO"),
                    subst.norm  = "ZOO",
                    nu.norm     = -1)
```

```
## [1] "Number of substances:                      7"
## [1] "Number of elementary constituents:         6"
## [1] "Number of constraints:                     0"
## [1] "Number of independent processes:           1"
## [1] "Stoichiometric coefficients successfully calculated."
```

```
nu.death.ZOO <-
   calc.stoich.coef(alpha       = alpha.2,
                    name        = "death.ZOO",
                    subst       = c("NH4","HPO4","HCO3","O2","H","H2O","ZOO","POM"),
                    subst.norm  = "ZOO",
                    nu.norm     = -1,
                    constraints = list(c("ZOO" = param$Y.ZOO.death,
                                         "POM" = 1)))
```

```
## [1] "Number of substances:                      8"
## [1] "Number of elementary constituents:         6"
## [1] "Number of constraints:                     1"
## [1] "Number of independent processes:           1"
## [1] "Stoichiometric coefficients successfully calculated."
```

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

```
nu.2 <- rbind(nu.gro.ALG.NH4,
              nu.gro.ALG.NO3,
              nu.resp.ALG,
              nu.death.ALG,
              nu.gro.ZOO,
              nu.resp.ZOO,
              nu.death.ZOO)

print(round(nu.2,3))
```

```
##                NH4    NO3 HPO4   HCO3     O2      H    H2O ALG ZOO   POM
## gro.ALG.NH4 -0.004  0.000    0 -0.030  0.029 -0.026  0.002   1   0 0.000
## gro.ALG.NO3  0.000 -0.004    0 -0.030  0.038 -0.035 -0.002   1   0 0.000
## resp.ALG     0.004  0.000    0  0.030 -0.029  0.026 -0.002  -1   0 0.000
## death.ALG    0.002  0.000    0  0.002  0.002 -0.001  0.005  -1   0 0.714
## groZOO       0.011  0.000    0  0.042 -0.031  0.030  0.011  -5   1 2.000
## resp.ZOO     0.004  0.000    0  0.030 -0.029  0.026 -0.002   0  -1 0.000
## death.ZOO    0.001  0.000    0 -0.010  0.014 -0.011  0.008   0  -1 1.000
```

This stoichiometric matrix should be checked in particular for the signs of the stoichiometric coefficients. As we chose the yields for death accordingly above, death of neither algae or zooplankton consumes phosphate or ammonium. However, we also have to check that it does not consume oxygen, which is not so easy to avoid. If it would consume oxygen, we would have to switch to the approach of using multiple fractions of dead organic matter (we do not have this problem if the compostion of dead organic matter is the same as

that of the dying organism).

Finally, we demonstrate the most frequently occurring errors. This was the correctly calculated stoichiometry for zooplankton growth (rounding of the output makes it more pleasant to read):

```
# Correct stoichiometry of growth of zooplankton:

round(calc.stoich.coef(alpha       = alpha.2,
                       name        = "groZOO",
                       subst       = c("NH4","HPO4","HCO3","O2","H","H2O","ALG","ZOO","POM"),
                       subst.norm  = "ZOO",
                       nu.norm     = 1,
                       constraints = list(c("ZOO" = 1,
                                            "ALG" = param$Y.ZOO),
                                          c("POM" = 1,
                                            "ALG" = param$f.e))),3)
```

```
## [1] "Number of substances:                     9"
## [1] "Number of elementary constituents:        6"
## [1] "Number of constraints:                    2"
## [1] "Number of independent processes:          1"
## [1] "Stoichiometric coefficients successfully calculated."
```

```
##          NH4 NO3 HPO4  HCO3     O2    H   H2O ALG ZOO POM
## groZOO 0.011   0    0 0.042 -0.031 0.03 0.011  -5   1   2
```

If we forget a constraint, the program cannot calculate a unique solution:

```
# Missing constraint error for the stoichiometry of growth of zooplankton:

round(calc.stoich.coef(alpha       = alpha.2,
                       name        = "groZOO",
                       subst       = c("NH4","HPO4","HCO3","O2","H","H2O","ALG","ZOO","POM"),
                       subst.norm  = "ZOO",
                       nu.norm     = 1,
                       constraints = list(c("ZOO" = 1,
                                            "ALG" = param$Y.ZOO))),3)
```

```
## [1] "Number of substances:                     9"
## [1] "Number of elementary constituents:        6"
## [1] "Number of constraints:                    1"
## [1] "Number of independent processes:          2"
## [1] "Number of required additional constraints: 1"
## [1] "Ratios are fixed between the following substance pairs:"
##      ZOO NH4 HPO4 HCO3 O2 H H2O ALG POM
## ZOO                              x
## NH4
## HPO4
## HCO3
## O2
## H
## H2O
## ALG  x
## POM
## [1] "Process not unique, 1 additional constraints needed."
```

```
## [1] NA
```

The same happen, if we include a substance that does not allow for a unique solution. In this case, including ammonia and nitrate does not allow the program to resolve in which form nitrogen should be released when respiring part of the algae:

```
# Too many substances error for the stoichiometry of growth of zooplankton:

round(calc.stoich.coef(alpha       = alpha.2,
                       name        = "groZOO",
                       subst       = c("NH4","NO3","HPO4","HCO3","O2","H","H2O","ALG","ZOO","POM"),
                       subst.norm  = "ZOO",
                       nu.norm     = 1,
                       constraints = list(c("ZOO" = 1,
                                            "ALG" = param$Y.ZOO),
                                          c("POM" = 1,
                                            "ALG" = param$f.e))),3)
```

```
## [1] "Number of substances:                      10"
## [1] "Number of elementary constituents:         6"
## [1] "Number of constraints:                     2"
## [1] "Number of independent processes:           2"
## [1] "Number of required additional constraints: 1"
## [1] "Ratios are fixed between the following substance pairs:"
##      ZOO NH4 NO3 HPO4 HCO3 O2 H H2O ALG POM
## ZOO              x    x             x   x
## NH4
## NO3
## HPO4 x                x             x   x
## HCO3 x           x                  x   x
## O2
## H
## H2O
## ALG  x           x    x                 x
## POM  x           x    x             x
## [1] "Process not unique, 1 additional constraints needed."
```

```
## [1] NA
```

The other kind of problem occurs if we forget a substance, in this example ammonia:

```
# Missing substance error for the stoichiometry of growth of zooplankton:

round(calc.stoich.coef(alpha       = alpha.2,
                       name        = "groZOO",
                       subst       = c("HPO4","HCO3","O2","H","H2O","ALG","ZOO","POM"),
                       subst.norm  = "ZOO",
                       nu.norm     = 1,
                       constraints = list(c("ZOO" = 1,
                                            "ALG" = param$Y.ZOO),
                                          c("POM" = 1,
                                            "ALG" = param$f.e))),3)
```

```
## [1] "Number of substances:                      8"
## [1] "Number of elementary constituents:         6"
## [1] "Number of constraints:                     2"
## [1] "Number of independent processes:           0"
## [1] "No process possible"
```

```
## [1] NA
```

There is no solution as there is no substance available for the release of the respired nitrogen.