Dissolution of spherical silica particles

Exercises Accompanying the Course Reaction Transport Modelling in the Hydrosphere

Lubos Polerecky & Karline Soetaert, Utrecht University

February 2021

Introduction

Silica is a common material in the natural environment, and its dissolution or precipitation is of significant interest. You will make a model of the dissolution and precipitation of silica. Using the model, you will then investigate the conditions under which dissolution or precipitation occurs.

As explained in class, dissolution of silica, $SiO_2(s)$, in water results in the formation of silicic acid, $Si(OH)_4(aq)$. This process can be written as the following reversible reaction:

$$\mathit{Reaction}\ 1: \quad SiO_2(s) + 2H_2O \leftrightarrow Si(OH)_4(aq)$$

This reaction is governed by the differential equation

$$\frac{dC_s}{dt} = -\frac{dC}{dt} = k_p \times \frac{A}{V} \times (C - C_{eq}), \qquad (1)$$

where C is the concentration of silicic acid (i.e., dissolved silica), and C_s is the concentration of solid silica (both in $mol\ Si\ m^{-3}$). A is the area of the reactive silica surface in a volume of water V, k_p is the rate constant of the precipitation reaction, and C_{eq} is the silica solubility.

Problem formulation

Diatoms are microscopic algae whose cell wall is made of silica. After the cell dies, diatom frustules sink from the surface waters towards the sea bottom. As the silica frustules encounter waters that are undersaturated in silicic acid, they will start to dissolve. You will develop a model to mimic this dissolution process, approximating the frustules as small spherical particles.

Our model system contains N spherical silica particles suspended in a volume V. We assume that the particles are identical (radius r) and that the particle density (N/V) remains constant. You may recall that the volume of a sphere with radius r is $(4/3)\pi r^3$, while its surface area is $4\pi r^2$.

Because the solid silica is in the form of many small spherical particles, the total concentration of solid silica, C_s, and the radius of each particle, r, are related according to:

$$C_s = \frac{N}{V} \times \frac{4\pi r^3}{3} \frac{\rho}{MW}, \qquad (2)$$

where MW and ρ is, respectively, the molar weight (60.08 $g\ mol^{-1}$) and the density (2196 $kg\ m^{-3}$) of amorphous silica.

Tasks

Task 1. Model development

Write a model for predicting the dissolution (or precipitation) of silica particles in water.

- What are the state variables? What are their units?
- Draw the conceptual diagram.
- What are the corresponding mass balance equations?
- What are the rate expressions?

Task 2. Model implementation

- Implement the model in R. Make sure to output also these two variables:
  - the radius of the silica particles, and
  - the total Si concentration.
- Assume $k_p = 1\ m\ yr^{-1}$ and $C_{eq} = 1\ mmol\ L^{-1}$. (Note: the value of the rate constant may not be correct, but it is not so important in this exercise.)
- The particle density is 8×10^6 particles m^{-3}.
- The initial radius of the spherical silica particles is $r_{ini} = 0.1\ mm$.
- For the first run, assume that the initial concentration of dissolved silica is 0.

Note: in your R-code you will need to calculate x^n, where n is a non-integer value. This at times gives numerical problems if $x < 0$. Many programming languages, including *R*, then return a *NaN*, i.e., "not a number". In an ideal world, and when dealing with concentrations, the value of x should *not* be negative. But, due to finite precision of the integration methods, very small negative numbers *may* arise. It is therfore prudent to set to 0 any negative values representing concentrations. That is, rather than writing *x^(1/2)*, write *max(0, x)^(1/2)*.

Task 3. Model application

- Run the model dynamically for 30 years; use function *ode* from the *deSolve* package to solve the model.
  - Explain the dynamics.
- Find the steady-state solution.
  - Use function *runsteady* from package *rootSolve* to find this solution.
  - What is the value of the sphere's radius at steady-state?

Task 4. Explore the model dynamics

Make a number of runs where you vary the initial concentrations of the silica particles or of the silicic acid.

- Which conditions correspond to the scenarios where the particles dissolve *partially* or *fully*?
- Under which initial conditions does *precipitation* of silica occur?
- Perform a sensitivy analysis to evaluate how the steady-state size of particles varies as a function of the particle density. Perform this analysis for the initial silicic acid concentration of zero.
  - Vary the particle density between 1×10^6 and 10×10^6 particles m^{-3} (create a sequence).
  - Estimate the steady-state for each of these densities, using *runsteady*, and store the resulting radius and dissolved Si concentration. (Hint: use a for loop.)

- Plot the steady-state particle radius and dissolved Si concentration as a function of the particle density.

Answer

Task 1. Model equations

From a mass-balance perspective it is logical to use the silicic acid concentration, C, and the solid phase silica concentration, C_s, as the model state variables, with units of $[mol\ Si\ m^{-3}]$.

The dissolution rate is expressed in terms of the surface of the spherical particles, so at each time step we need to estimate the surface area of the particles based on the solid silica concentration.

We first use Eq. 2 and calculate r as a function of C_s. We obtain

$$r = \left(C_s \times \frac{V}{N} \times \frac{MW}{\frac{4\pi\rho}{3}} \right)^{1/3}. \qquad (3)$$

By substituting this expression into Eq. 1 we obtain

$$\frac{dC}{dt} = -k_2 C_s^{2/3} \times (C - C_{eq}), \qquad (4)$$

$$\frac{dC_s}{dt} = k_2 C_s^{2/3} \times (C - C_{eq}), \qquad (5)$$

where the modified rate constant k_2 is calculated from the original rate constant k_p according to

$$k_2 = 4\pi k_p \times \left(\frac{N}{V} \right)^{1/3} \times \left(\frac{MW}{\frac{4}{3}\pi\rho} \right)^{2/3}. \qquad (6)$$

Note that the dissolution/precipitation rate does not linearly depend on the solid silica concentration, but on this quantity raised to the power of 2/3.

Task 2. Implementation in R

First, we define model parameters and initial conditions:

```
# parameters
parms <- c(
 NV    = 8e6      , # [number/m3], density of spherical silica particles (N/V)
 kp    = 1        , # [m/yr],      precipitation rate constant (assumed)
 MW    = 60.08e-3 , # [kg/mol],    silica molar weight
 rho   = 2196     , # [kg/m3],     silica density
 Ceq   = 1          # [mol/m3],    equilibrium concentration of silicic acid
)
```

The initial concentration of the solid phase silica can be estimated from the density of particles and their initial size (Eq. 2). As we will need to estimate the initial condition of solid silica several times from the parameters, we define a function (*Cs.ini.fun*):

```
Cs.ini.fun <- function(parms, r = 0.0001)
  with (as.list(parms), {
    Cs.ini <-  NV* 4/3*pi*r^3 * rho/MW
    return(Cs.ini)  # [mol/m3])
  })
```

In the first run, the dissolved Si concentration is 0, while the radius of the spheres is 0.1 mm.

```
C.ini  <- 0       # [mol/m3], initial concentration of silicic acid
r.ini  <- 0.1e-3  # [m],      initial radius of the spherical silica particles

# initial concentration of silica (assuming all spheres have the same radius)
Cs.ini    <- Cs.ini.fun(parms, r=r.ini)        # [mol/m3]

state.ini <- c(C=C.ini, Cs=Cs.ini)
```

In the model function, we return the total silica concentration, and the radius of the spheres, as output variables; the latter is calculated from Eq. 3.

```
DissolveSilica <- function(t, state, parms) {
  with (as.list(c(state, parms)), {

    k2  <- 4*pi*kp * NV^(1/3) * (MW/(4/3*pi*rho))^(2/3)    # modified rate constant
    Cs  <- max(0, Cs)   # to avoid that the power of a tiny negative number gives NaN

    # mass balance equations
    dC  <- -k2 * Cs^(2/3) * (C-Ceq)
    dCs <-  k2 * Cs^(2/3) * (C-Ceq)

    return(list(c(dC, dCs),
           r = ( Cs/NV * MW/(4/3*pi*rho) )^(1/3),  # Particle radius
           Ctot = C+Cs))                           # Total Si
  })
}
```

Task 3. Model solutions

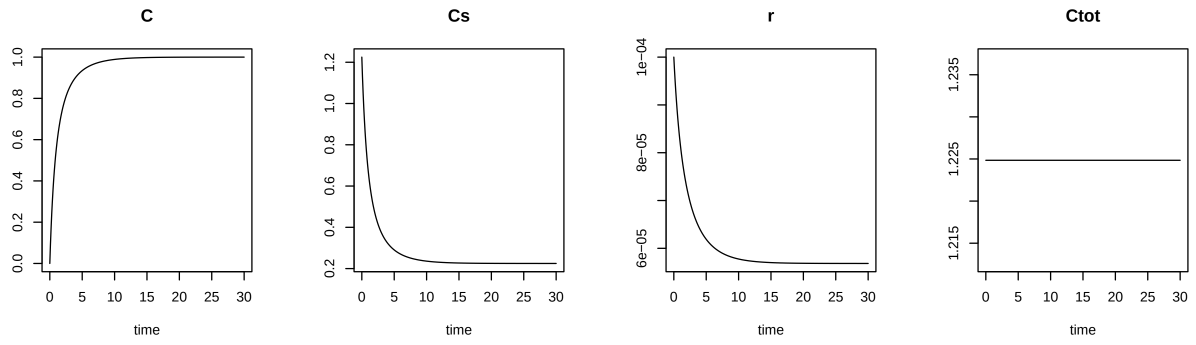
Scenario 1

Now we run the model using the above parameters and plot the results.

```
library(deSolve)
library(rootSolve)

times <- seq(from=0, to=30, by=0.1)  # time in years
out1  <- ode(y=state.ini, times=times, func=DissolveSilica, parms=parms)

plot(out1, mfrow=c(1,4))  # set mfrow to put all figures in one row (and 4 columns)
```



The last values are:

```
tail(out1, n = 1)
```

```
##         time         C        Cs            r     Ctot
## [301,]    30 0.9999836 0.2248611 5.683437e-05 1.224845
```

The steady-state solution:

```
std1  <- runsteady(y=state.ini, func=DissolveSilica, parms=parms)
std1$y
```

```
##         C        Cs
## 1.0000000 0.2248446
```

```
std1$r
```

```
## [1] 5.683299e-05
```

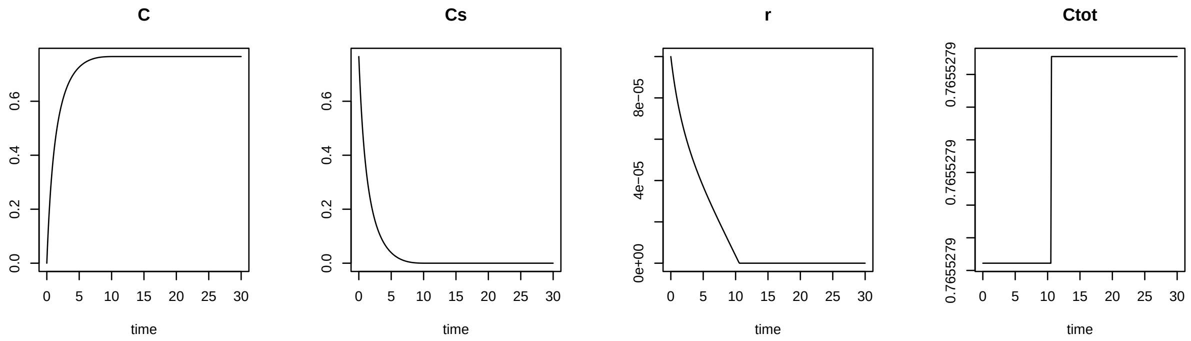
In this particular example the initial concentration of *total* silica in the water is high (i.e., above the silica solubility). Thus, as the particles dissolve, the concentration of silicic acid increases until it reaches solubility. At this point, the *dissolution stops because an equilibrium is reached*. In this scenario the particle radius decreased from 100 μm to about 57 μm. At steady-state, the particle radius is 56.833 μm.

Scenario 2

Now we decrease the particle density from 8×10^6 to 5×10^6 and run the model again.

```
parms2 <- parms
parms2["NV"] <- 5e6                             # [number/m3]
Cs.ini       <- Cs.ini.fun(parms2, r=r.ini)     # [mol/m3]
state2       <- c(C=C.ini, Cs=Cs.ini)

out2   <- ode(y=state2, times=times, func=DissolveSilica, parms=parms2,
              atol=1e-10, rtol=1e-10)    # atol and rtol to increase the precision
plot(out2, mfrow = c(1,4))
```



```
std2  <- runsteady(y=state2, func=DissolveSilica, parms=parms2)
std2$y
```

```
##             C            Cs
##  7.655283e-01 -3.781256e-07
```

```
std2$r
```

```
## [1] 0
```

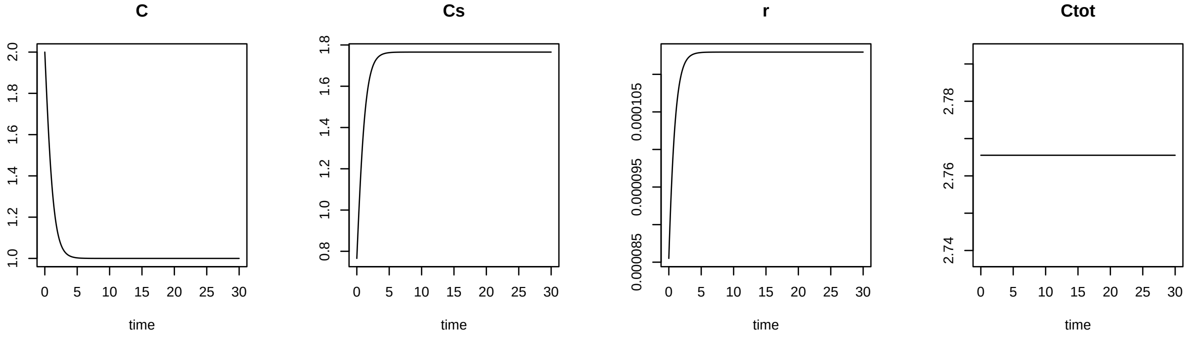
In this scenario the initial concentration of total silica in the water is low (below the silica solubility). Thus, the particles dissolve completely before the concentration of silicic acid reaches silica solubility. At this point, the *dissolution stops because there is no silica left to dissolve.* In this scenario the particle radius decreased from 100 μm to 0 μm.

Scenario 3

In the last scenario, we keep the same particle density but increase the initial silicic acid concentration above solubility. We expect that silica will precipitate.

```
C.ini  <- 2                        # mol/m3
state3 <- c(C=C.ini, Cs=Cs.ini)

out3   <- ode(y=state3, times=times, func=DissolveSilica, parms=parms,
              atol=1e-10, rtol=1e-10)
plot(out3, mfrow = c(1,4))
```



```
std3  <- runsteady(y=state3, func=DissolveSilica, parms=parms)
std3$y
```

```
##        C       Cs
## 1.000000 1.765528
```

```
std3$r
```

```
## [1] 0.0001129617
```

As expected, the size of the particles increases due to precipitation of silica from 100 μm to about 113 μm. This happens until an equilibrium is reached.

Task 4. Model sensitivity

We create a sequence of particle densities (*NV.seq*) for which we will estimate the equilibrium particle size and dissolved Si concentration at equilibrium.

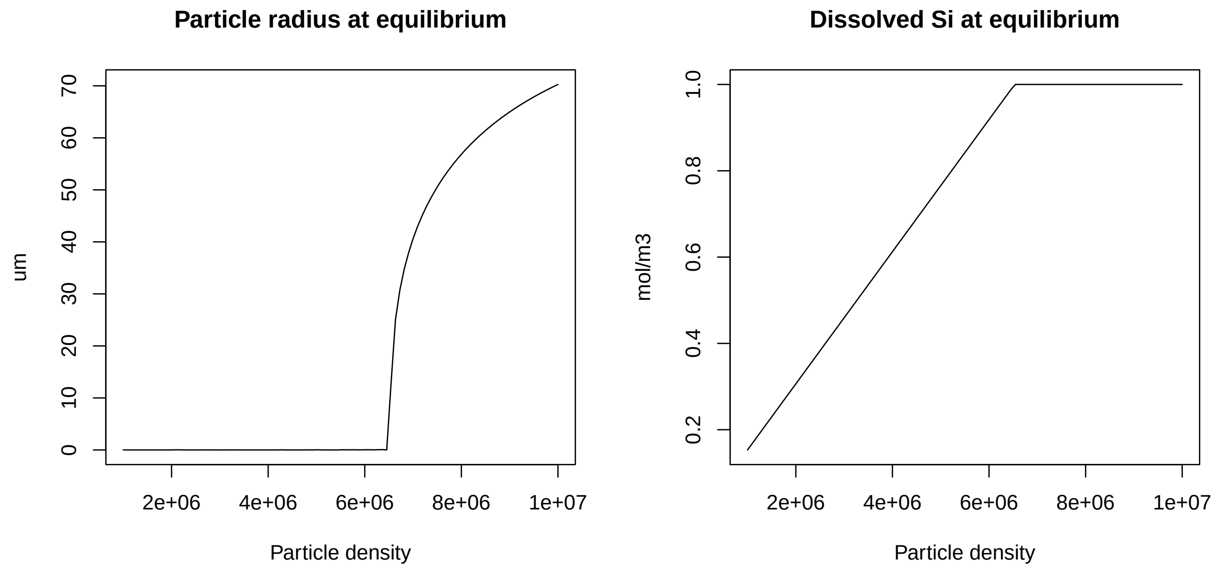
Then, for each of the elements in this sequence, the proper initial conditions are calculated (*state*), and the steady-state solution is estimated, using *runsteady*. From this, the equilibrium particle radius and dissolved Si concentration is extracted and stored in a vector. It is simplest to start with an empty (*NULL*) vector (r.out <- NULL), and then for each run concatenate the resulting radius to this vector.

```
# a sequence of particle densities
NV.seq <- seq(from=1e6, to=10e6, length.out=100)
```

```
r.out  <- NULL # will add steady-state "r" for each run
C.out  <- NULL # will add steady-state "C" for each run

for (nv in NV.seq) {
  parms2       <- parms          # a copy of default parameters
  parms2["NV"] <- nv             # new NV
  state <- c(C=0, Cs=Cs.ini.fun(parms2, r=r.ini))
  STD   <- runsteady(y=state, func=DissolveSilica, parms=parms2,
                     atol=1e-10, rtol=1e-10)
  r.out <- c(r.out, STD$r)
  C.out <- c(C.out, STD$y["C"])
}

par(mfrow = c(1,2))
plot(NV.seq, r.out*1e6, type="l", xlab="Particle density", ylab="um",
     main="Particle radius at equilibrium")
plot(NV.seq, C.out, type="l", xlab="Particle density", ylab="mol/m3",
     main="Dissolved Si at equilibrium")
```



References

R Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Karline Soetaert, Thomas Petzoldt, R. Woodrow Setzer (2010). Solving Differential Equations in R: Package deSolve. Journal of Statistical Software, 33(9), 1–25. URL http://www.jstatsoft.org/v33/i09/ DOI 10.18637/jss.v033.i09

Karline Soetaert (2009). rootSolve: Nonlinear root finding, equilibrium and steady-state analysis of ordinary differential equations. R-package version 1.6.