Group project report

A template for a more structured writing of the group project report

Authors

April 2022

Abstract

You can include a short abstract here. It should summarize the overall aim and key findings of your project.

Contents

1 Introduction

The report starts with an introduction, which should explain the broader context of the project and define project aims. Ultimately, it should serve as a *motivation* for the work presented in the subsequent sections.

2 Methods

The Methods section describes the "how". It should explain

- the theoretical underpinning of the model (state variables, including units, conceptual diagram, chemical equations, if relevant, mass balance equations);

- key assumptions (rate expressions, boundary conditions);
- model implementation in R (model parameters & state variables, model function).

2.1 Equations

When writing chemical or mathematical equations, you can use LaTeX commands within the Rmd file. For example:

- Displayed chemical equation (without a label):

$$CH_2O + O_2 \rightarrow CO_2 + H_2O$$

- Displayed chemical equation (with a label):

$$CH_2O + O_2 \rightarrow CO_2 + H_2O \tag{1}$$

- Displayed mathematical equation (with a label):

$$\frac{\partial O_2}{\partial t} = Transport(O_2) + R_{aeration} - R_{miner} \tag{2}$$

  The label can then be used to reference Eq. 2 and Eq. 1 even if you change the order of equations later on.

2.2 Model implementation in R

In this section you will show, and explain, how you implemented the model in R. As an example, here we include the implementation of the BOD model (available in the `RTM_1D` template).

```r
require(ReacTran)

# units: time=days, space=meters, amount=moles, concentration=mol/m3

# model grid
Length <- 1000                                 # [m]
N      <- 100                                  # [-] number of boxes
Grid   <- setup.grid.1D(L = Length, N = N)     # grid of N equally-sized boxes

# initial conditions - state variables are defined in the middle of grid cells
O2      <- rep(0.1,   times = N)               # [mol/m3]
BOD     <- rep(0.001, times = N)               # [mol/m3]

# initial state of the system: a vector with all state variables (2*N)
state.ini   <- c(O2, BOD)

# names of the modeled state variables
SVnames <- c("O2", "BOD")

# model parameters
pars <- c(
  D          = 100,   # [m2/d]      dispersion coefficient (tidal mixing)
  v          = 10,    # [m/d]       advection velocity
  kDecay     = 0.05 , # [/d]        rate constant of BOD decay (first-order process)
  K.O2       = 0.001, # [mol/m3]    half-saturation O2 concentration for BOD decay
  inputBOD   = 10,    # [mol/m2/d]  BOD input rate upstream
  BODdown    = 0.1,   # [mol/m3]    BOD concentration downstream
  O2up       = 0.25,  # [mol/m3]    O2 concentration upstream
  satO2      = 0.3,   # [mol/m3]    saturation concentration of O2 (i.e., solubility)
  kAeration = 0.1     # [/d]        rate constant for air-water O2 exchange
)

# Model function
BOD1D <- function(t, state, parms) {  # state is a long vector, at time t
  with (as.list(parms),{

  # The vectors of the state variables O2 and BOD are
  # "extracted" from the LONG vector state passed to the function as input.
    O2  <- state[ (0*N+1) : (1*N) ]   # first N elements for O2
    BOD <- state[ (1*N+1) : (2*N) ]   # second N elements for BOD

  # Transport - tran.1D approximates the spatial derivatives
  # note: for O2: zero-gradient boundary downstream (default)
    tranO2  <- tran.1D(C    = O2,
                       C.up = O2up,         # imposed conc upstream,
                       D = D, v = v,        # dispersion, advection
                       dx = Grid)           # Grid

    tranBOD <- tran.1D(C       = BOD,
                       flux.up = inputBOD,  # imposed flux upstream
                       C.down  = BODdown,   # imposed conc downstream
                       D = D, v = v,        # dispersion, advection
                       dx = Grid)           # Grid

  # rate expressions [mol/m3/d] - values in the middle of grid cells
    Decay    <- kDecay * BOD * O2/(O2+K.O2) # BOD decay, limited by O2
    Aeration <- kAeration * (satO2-O2)      # air-water exchange of O2

  # Time-derivatives: dC/dt = transport + production - consumption [mol/m3/d]
    dO2.dt    <- tranO2$dC  + Aeration - Decay
    dBOD.dt   <- tranBOD$dC            - Decay

  # return vector of time-derivatives and ordinary variables as a list
    return(list(c(dO2.dt, dBOD.dt),  # time-derivatives
                                     # (the same order as state variables!!)
  # additional output:

    # process rates along the domain (1D vector)
    Decay           = Decay,                 # mol/m3/d
    Aeration        = Aeration,              # mol/m3/d

    # mean process rates (a number)
    MeanDecay       = mean(Decay),           # mol/m3/d
    MeanAeration    = mean(Aeration),        # mol/m3/d

    # rates integrated along the domain (for budgetting)
    TotalDecay      = sum(Decay*Grid$dx),    # mol/m2/d
    TotalAeration   = sum(Aeration*Grid$dx), # mol/m2/d

    # fluxes at domain boundaries (for budgetting)
    BODinflux = tranBOD$flux.up,      # BOD flux INTO the system upstream,     mol/m2/d
    BODefflux = tranBOD$flux.down,    # BOD flux OUT of the system downstream, mol/m2/d
    O2influx  = tranO2$flux.up,       # O2 flux INTO the system upstream,      mol/m2/d
    O2efflux  = tranO2$flux.down))    # O2 flux OUT of the system downstream,  mol/m2/d
  })
}
```

2.3 Modeled scenarios

In this section you can also include the *calculation* (not yet plotting) of model outputs for specific model scenarios.

For example, here we find a steady-state solution for the default model parameters.

```
std <- steady.1D(y = state.ini, parms = pars, func = BOD1D,
          positive = TRUE, names = SVnames, nspec = length(SVnames), dimens = N,
          atol = 1e-10, rtol = 1e-10)
```

In contrast, here we solve the model assuming no aeration.

```
p2 <- pars
p2["kAeration"] <- 0
std2 <- steady.1D(y = state.ini, parms = p2, func = BOD1D,
          positive = TRUE, names = SVnames, nspec = length(SVnames), dimens = N,
          atol = 1e-10, rtol = 1e-10)
```

We also generate the content of tables. We do not display them just yet, but we wait until the Results section.

```
toselect <- c("TotalDecay", "TotalAeration", "BODinflux", "BODefflux",
              "O2influx", "O2efflux")
BUDGET <- data.frame(default     = unlist(std[toselect]),
                     no_aeration = unlist(std2[toselect]))
```

This can go on until you clarify all methodological aspects covered in your project.

3 Results

The Results section should summarize your key findings supported by graphs and/or tables.

3.1 Including variables generated in Methods

When knitting this file independently of the `methods.Rmd` file, you need to ensure that the values of the variables generated by the R-code in the `methods.Rmd` file are known here. You can do this by including the following R-chunk (enclosed between the opening and closing ‘‘‘) somewhere at the beginning of the `results.Rmd` file:

```
{r, message=FALSE, echo=FALSE, eval=TRUE}
res <- knitr::knit_child('methods.Rmd', quiet = TRUE)
```

Once you have finished editing the `results.Rmd` file and are ready to knit the “master” file of your project, you can omit knitting of the `methods.Rmd` child here by using `eval=FALSE` in the above R-chunk:

```
{r, message=FALSE, echo=FALSE, eval=FALSE}
res <- knitr::knit_child('methods.Rmd', quiet = TRUE)
```

3.2 Plotting graphs

It is recommended to use *one* R-chunk for plotting one figure (plot). In this way, you can individually optimize the size of the figure and the corresponding figure caption.

Note that if you specify `fig.cap`, the figure will become a *floating* object and will have a reference number. How this floating object is placed within your final document can be prioritized in the yaml header of your Rmd file. Here, we use

```
    \usepackage{float}
    \floatplacement{figure}{ht}
```

which means that the first preference is to place the figure in the same place where the figure is generated (`h`, meaning *here*), while the second preference is to place it at the top of the next page (`t`, meaning *top*). These are LaTeX-specific tricks.

Note that there is no need to display the R-code that is used to generate the graphs, as this code is typically quite long but not overly informative. This is achieved by using the `echo=FALSE` flag in the corresponding R-chunk.

3.3 Reporting values in tables

Tables generated in the Methods section can be nicely typset using the `kable` function from the `knitr` package.

```r
knitr::kable(BUDGET, digits = 2, caption = "Budgets for the different model scenarios.")
```

Table 1: Budgets for the different model scenarios.

	default	no_aeration
TotalDecay	9.96	3.08
TotalAeration	9.87	0.00
BODinflux	10.00	10.00
BODefflux	0.04	6.92
O2influx	2.95	3.08
O2efflux	2.86	0.00

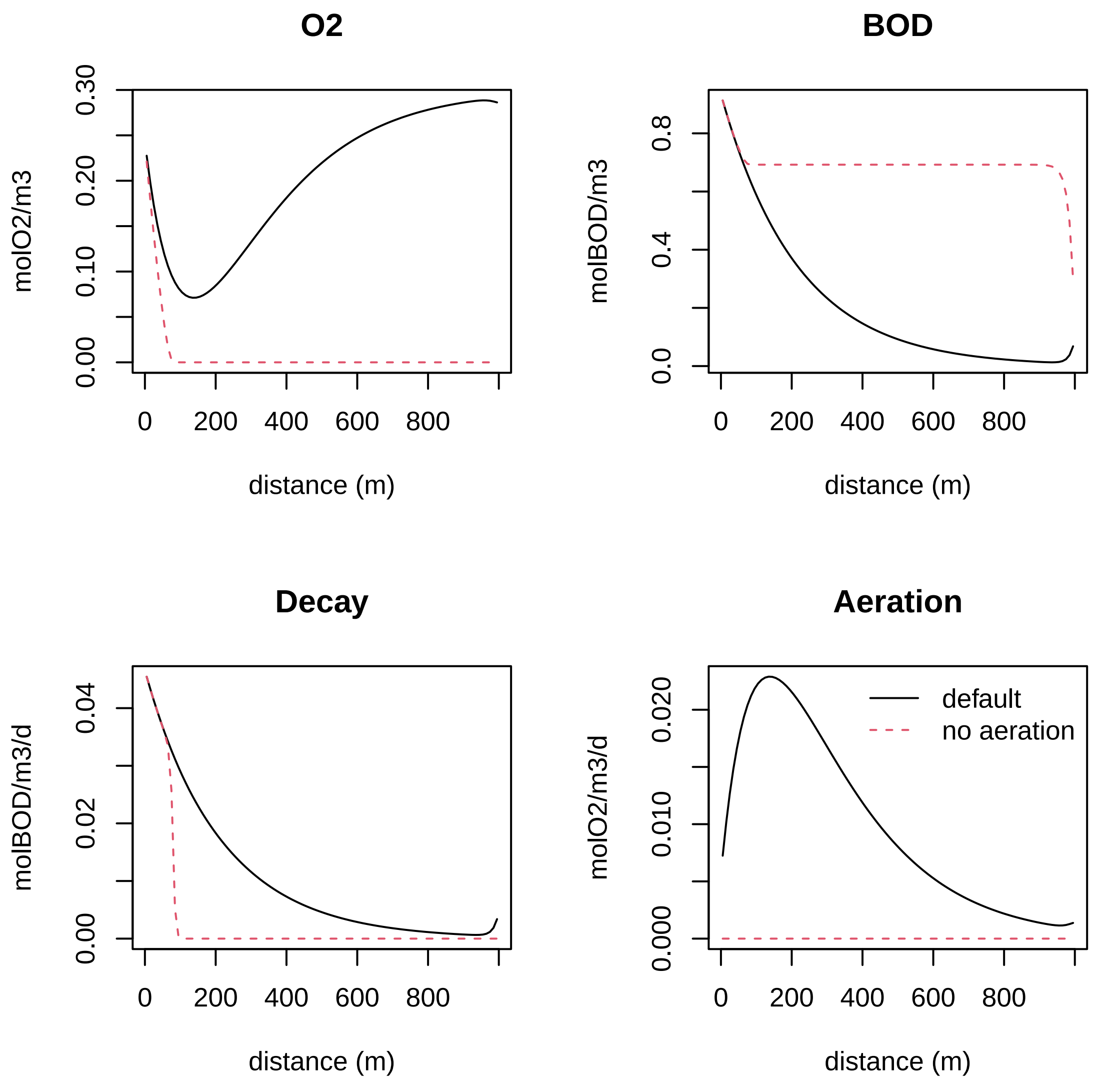

Figure 1: A comparison of model outputs for a default scenario and for a scenario without aeration.

4 Discussion

The Discussion section should tie the results into a coherent story. Specifically, it should explain what the results *mean*, clarify to what extent the project aims were reached, provide ideas for future work, etc.

5 Acknowledgements

It is always a good idea to acknowledge help of others.

6 Author's contributions

We also require that you list specific contributions of each author.

7 References

Lubos Polerecky, Dries Bonte, and Karline Soetaert (2021). RTM: learning environment for reaction-transport modelling in R. https://github.com/dynamic-R/RTM

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/.

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

Soetaert, Karline and Meysman, Filip, (2012). Reactive transport in aquatic ecosystems: Rapid model prototyping in the open source software R Environmental Modelling & Software, 32, 49-60.