The FESDIA package - adding perturbations to the C, N, P, Fe and S cycle

Karline Soetaert

29-July-2020

FESDIAperturb

```
require(FESDIA)
```

The FESDIA package contains functions to generate diagenetic profiles, describing the cycles of C, N, O2, Fe, S and P. It is based on the OMEXDIA model (Soetaert et al., 1996a, b), extended with P, S, Fe dynamics.

The model describes seventeen state variables, in 100 layers:

- 2 fractions of organic carbon (FDET,SDET): fast and slow decaying, solid substance.
- Oxygen (O2), dissolved substance.
- Nitrate (NO3), dissolved substance.
- Nitrite (NO2), dissolved substance.
- Ammonia (NH3), dissolved substance.
- Dissolved inorganic carbon (DIC), dissolved substance
- Iron 2+ (Fe), dissolved substance
- Sulphide (H2S), dissolved substance
- Methane (CH4), dissolved substance
- Phosphate (PO4), dissolved substance
- Alkalinity (ALK), dissolved substance
- Iron hydroxides (FeOH3), solid substance
- Iron-bound P (FeP), P bound to iron oxides, solid substance
- Ca-bound P (CaP), apatite, solid substance
- Adsorbed P (Pads), solid substance

Time is expressed in days, and space is expressed in centimeters.

Concentrations of liquids and solids are expressed in [nmol/cm3 liquid] and [nmol/cm3 solid] respectively (Note: this is the same as [mmol/m3 liquid] and [mmol/m3 solid]).

Compared to the OMEXDIA model, FESDIA includes the following additions:

- simple phosphorus, iron and sulphur dynamics
- long-distance H2S and CH4 oxidation, e.g. by cable bacteria or worms associated with chemosynthetic bacteria
- allowing boundary conditions with water overlying sediment or exposure to the air.
- external conditions set either with time-variable forcings or as constant parameters
- bottom water conditions either imposed or dynamically modeled
- possibility to include sediment perturbation events
- vertical profiles of porosity, irrigation, bioturbation either set with parameters or inputted as data.

The model is implemented in fortran (for speed) and linked to R (for flexibility).

Main functions

The main functions allow to solve the model to steady state (*FESDIAsolve*), to run it dynamically (*FESDIAdyna*), or to add perturbations (*FESDIAperturb*) to dynamic simulations.

Steady-state solution, function FESDIAsolve

Function *FESDIAsolve* finds the steady-state solution of the FESDIA model. Its arguments are:

```
args(FESDIAsolve)
```

```
function (parms = list(), yini = NULL, gridtype = 1, Grid = NULL,
    porosity = NULL, bioturbation = NULL, irrigation = NULL,
    surface = NULL, diffusionfactor = NULL, dynamicbottomwater = FALSE,
    ratefactor = NULL, calcpH = FALSE, verbose = FALSE, method = NULL,
    times = c(0, 1e+06), ...)
NULL
```

here *parms* is a list with a subset of the FESDIA parameters (see main vignette for what they mean and their default values). If unspecified, then the default parameters are used.

Dynamic run, function FESDIAdyna

Function *FESDIAdyna* runs the model dynamically without perturbations. Its arguments are:

```
args(FESDIAdyna)
```

```
function (parms = list(), times = 0:365, spinup = NULL, yini = NULL,
    gridtype = 1, Grid = NULL, porosity = NULL, bioturbation = NULL,
    irrigation = NULL, surface = NULL, diffusionfactor = NULL,
    dynamicbottomwater = FALSE, CfluxForc = NULL, FeOH3fluxForc = NULL,
    CaPfluxForc = NULL, O2bwForc = NULL, NO3bwForc = NULL, NO2bwForc = NULL,
    NH3bwForc = NULL, FebwForc = NULL, H2SbwForc = NULL, SO4bwForc = NULL,
    CH4bwForc = NULL, PO4bwForc = NULL, DICbwForc = NULL, ALKbwForc = NULL,
    wForc = NULL, biotForc = NULL, irrForc = NULL, rFastForc = NULL,
    rSlowForc = NULL, pFastForc = NULL, MPBprodForc = NULL, gasfluxForc = NULL,
    HwaterForc = NULL, ratefactor = NULL, calcpH = FALSE, verbose = FALSE,
    ...)
NULL
```

Perturbation run, function FESDIAperturb

Function *FESDIAperturb* runs the FESDIA model for a specific time interval while adding perturbations at requested times.

Its arguments are:

```
args(FESDIAperturb)
```

```
function (parms = list(), times = 0:365, spinup = NULL, yini = NULL,
    gridtype = 1, Grid = NULL, porosity = NULL, bioturbation = NULL,
    irrigation = NULL, surface = NULL, diffusionfactor = NULL,
    dynamicbottomwater = FALSE, perturbType = "mix", perturbTimes = seq(from = 0,
        to = max(times), by = 365), perturbDepth = 5, concfac = 1,
```

```
    CfluxForc = NULL, FeOH3fluxForc = NULL, CaPfluxForc = NULL,
    O2bwForc = NULL, NO3bwForc = NULL, NO2bwForc = NULL, NH3bwForc = NULL,
    FebwForc = NULL, H2SbwForc = NULL, SO4bwForc = NULL, CH4bwForc = NULL,
    PO4bwForc = NULL, DICbwForc = NULL, ALKbwForc = NULL, wForc = NULL,
    biotForc = NULL, irrForc = NULL, rFastForc = NULL, rSlowForc = NULL,
    pFastForc = NULL, MPBprodForc = NULL, gasfluxForc = NULL,
    HwaterForc = NULL, ratefactor = NULL, verbose = FALSE, ...)
NULL
```

The functions to run the model dynamically also allow for several external conditions to be either constants or to vary in time. Thus, they can be set by a parameter or as a forcing function.

These conditions are:

- the flux of carbon, CaP and FeP (*Cflux, CaPflux, FeOH3flux*), forcings *CfluxForc, CaPfluxForc, FeOH3fluxForc*),
- the bottom water concentrations (*O2bw, NO2bw, NO3bw, NH3bw, H2Sbw, PO4bw, DICbw, ALKbw*), forcings *O2bwForc, NO3bwForc, NO2bwForc, NH3bwForc, ODUbwForc, PO4bwForc, DICbwForc, ALKbwForc*)
- the sedimentation, bioturbation and bio-irrigation rates (*w, biot, irr*), (*wForc, biotForc, irrForc*)
- the decay rates of organic matter (*rFast, rSlow*) and the fraction fast organic matter present in the flux (*pFast*), forcings (*rFastForc, rSlowForc, pFastForc*)
- the microphytobenthos production rate (*MPBprod*), (*MPBprodForc*)
- the air-sea exchange rate when exposed to the air (*gasflux*), (*gasfluxForc*)
- the height of the overlying water (*Hwater*), (*HwaterForc*), used only if *dynamicbottomwater* is *TRUE*.
- *ratefac* is a (time series or a constant) multiplication factor, that is multiplied with all biogeochemical rates. It can be used to impose temperature dependency.

These forcing functions are either prescribed as a list that either contains a data series (*list (data = ...)*) or as a list that specifies a periodic signal, defined by the amplitude (*amp*), *period*, *phase*, a coefficient that defines the strength of the periodic signal (*power*) and the minimum value (*min*) : the default settings are: *list(amp = 0, period = 365, phase = 0, pow = 1, min = 0)*. The mean value in the sine function is given by the corresponding parameter.

For instance, for the C flux, the seasonal signal would be defined as: $max(min, Cflux*(1+(amp*sin((times-phase)/period*2*pi))^pow)$.

Three types of perturbations are possible (argument *perturbType*):

- *mixing* straightens the profiles over a certain depth
- *erosion* removes part of the surficial sediment
- *deposition* adds sediment on top.

These perturbations are implemented as events, and need input of the perturbation times (*perturbTimes*), and the depth (*perturbDepth*). For deposition events, the factor of increase/decrease of the solid fraction concentration can also be inputted (*concfac*).

Accessory functions

The default values of the parameters, and their units can be interrogated:

```
P  <- FESDIAparms()
head(P)
```

```
##                 parms       units                description
## Cflux     4.566210e+02 nmolC/cm2/d total organic C deposition
## pFast     9.000000e-01           -   part FDET in carbon flux
```

```
## FeOH3flux 1.000000e+00  nmol/cm2/d   deposition rate of FeOH3
## CaPflux   0.000000e+00  nmol/cm2/d     deposition rate of CaP
## rFast     6.849315e-02          /d            decay rate FDET
## rSlow     1.369863e-04          /d            decay rate SDET
```

See the appendix of the main vignette for a complete list of the parameters.

Note: some parameters only apply if the bottom water concentration is modeled dynamically; they comprise the *dilution* of the bottom water (nudging to bottom water concentration), the height of the bottom water (*Hwater*), and the sinking rate of the solid constituents (C, FeP, FeOH3) (parameters *Cfall*, *FePfall* and *FeOH3fall*).

Budgets

Once the model is solved, it is possible to calculate budgets of the C, N, P, S, Fe and O2 cycle (*FESDIAbudgetC*, *FESDIAbudgetN*, *FESDIAbudgetP*, *FESDIAbudgetS*, *FESDIAbudgetFe*, *FESDIAbudgetO2*).

```
std <- FESDIAsolve()
print(FESDIAbudgetC(std))
```

```
## $Fluxes
##             FDET         SDET           DIC           CH4 CinCaP CaCO3 ARAG        Total
## surface 410.9589 4.566210e+01 -456.61647919 -4.379898e-08      0     0    0 4.525330e-03
## bottom    0.0000 1.168446e-71    0.00452533  0.000000e+00      0     0    0 4.525330e-03
## perturb   0.0000 0.000000e+00    0.00000000  0.000000e+00      0     0    0 0.000000e+00
## netin   410.9589 4.566210e+01 -456.62100452 -4.379898e-08      0     0    0 9.729705e-12
##
## $Rates
##             OxicMineralisation Denitrification IronReduction SulphateReduction Methanogenesis
## nmolC/cm2/d           426.9966        9.340614     0.7385786          18.91766      0.6275274
##             TotalMineralisation CH4oxidation CH4oxid.dist CH4oxidAOM MPBDICuptake MPBFDETproduction
## nmolC/cm2/d             456.621    0.0169016            0  0.2968621            0                 0
##             MPBResp CaPprecipitation CaPdissolution CaCO3dissolution ARAGdissolution
## nmolC/cm2/d       0                0              0                0               0
##             CaCO3production
## nmolC/cm2/d               0
##
## $Losses
## [1] 0.00452533
##
## $dC
##          Ext           DET           DIC           CaP           CH4           MPB         CaCO3
##  4.525373e-03 -1.136868e-13 -9.606538e-12  0.000000e+00 -4.379898e-08  0.000000e+00  0.000000e+00
##          ARAG        Burial           sum
##  0.000000e+00 -4.525330e-03  3.005582e-14
##
## $Delta
## [1] 9.750213e-12
##
## $Fluxmat
##             Ext     DET         DIC CaP       CH4 MPB CaCO3 ARAG       Burial
## Ext      0.0000 456.621   0.0000000   0 0.0000000   0     0    0 0.000000e+00
## DET      0.0000   0.000 456.3072408   0 0.3137637   0     0    0 1.168446e-71
## DIC    456.6165   0.000   0.0000000   0 0.0000000   0     0    0 4.525330e-03
## CaP      0.0000   0.000   0.0000000   0 0.0000000   0     0    0 0.000000e+00
```

```
## CH4      0.0000   0.000   0.3137637   0 0.0000000   0     0    0 0.000000e+00
## MPB      0.0000   0.000   0.0000000   0 0.0000000   0     0    0 0.000000e+00
## CaCO3    0.0000   0.000   0.0000000   0 0.0000000   0     0    0 0.000000e+00
## ARAG     0.0000   0.000   0.0000000   0 0.0000000   0     0    0 0.000000e+00
## Burial   0.0000   0.000   0.0000000   0 0.0000000   0     0    0 0.000000e+00
```

Perturbation runs

First the default run:

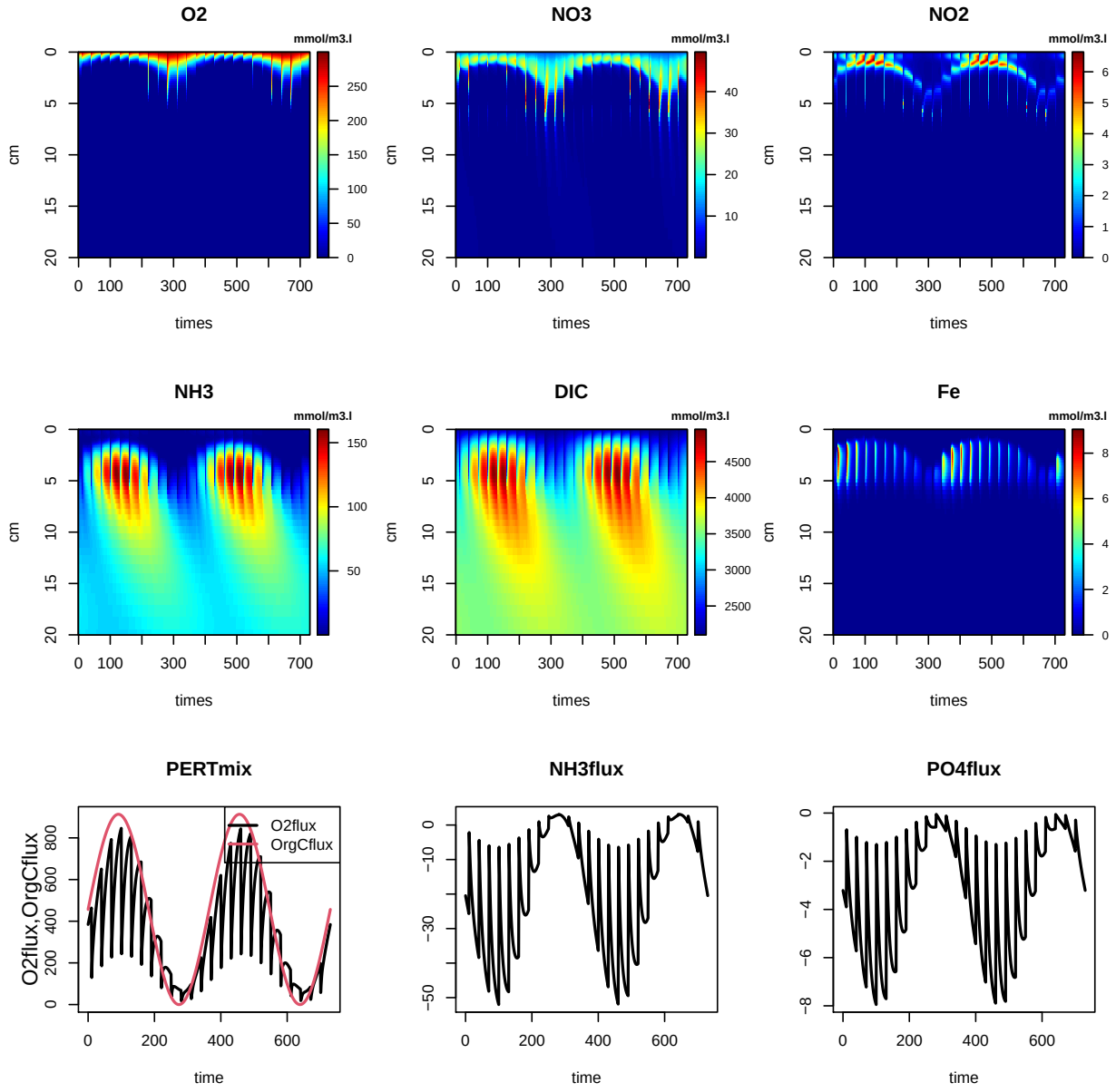
```
times <- 0:730
DIA <- FESDIAdyna(Cflux = list(amp = 1), spinup = 0:730,
        times = times, verbose = FALSE)
```

Mixing events

We add a mixing event every 30 days, and run this for 2 years, with two years spinup (spinup not shown).

```
times <- 0:730
PERTmix <- FESDIAperturb(Cflux = list(amp = 1), spinup = 0:730,
        perturbTimes = seq(from = 10, to = 730, by = 30),
        times = times, verbose = FALSE)

image2D(PERTmix,  ylim = c(20, 0), which = 3:8, mfrow = c(3,3))
matplot.0D(PERTmix, which = c("OrgCflux", "O2flux"), mfrow = NULL, lty = 1, lwd = 2)
plot(PERTmix, which = c("NH3flux", "PO4flux"), mfrow = NULL, lwd = 2)
```



The instantaneous fluxes

```
PertFlux <- attributes(PERTmix)$perturbFlux
knitr:::kable(rbind(PertFlux))
```

	time	FDET	SDET	O2	NO3	NO2	NH3	DIC	Fe	FeOH3	H2S
Fluxes	10	0	0	679.7370	-12.6373279	-1.947960	-23.273346	-20.87647	0	0	0
Fluxes.1	40	0	0	733.1636	-0.2112558	-2.679699	-85.783274	-31.61512	0	0	0
Fluxes.2	70	0	0	754.4140	8.5659172	-3.357257	-154.531740	-41.84477	0	0	0
Fluxes.3	100	0	0	760.0861	13.5443343	-3.729861	-211.072733	-50.15743	0	0	0
Fluxes.4	130	0	0	758.2285	12.8990082	-3.721666	-237.029739	-54.41141	0	0	0
Fluxes.5	160	0	0	748.3358	7.9017343	-3.417479	-224.229434	-53.50373	0	0	0
Fluxes.6	190	0	0	725.1087	1.6315593	-2.914725	-177.666207	-47.67358	0	0	0
Fluxes.7	220	0	0	685.2755	-3.1502851	-2.311459	-115.880720	-38.44232	0	0	0

	time	FDET	SDET	O2	NO3	NO2	NH3	DIC	Fe	FeOH3	H2S
Fluxes.8	250	0	0	632.5130	-6.2057415	-1.706605	-59.231169	-28.21860	0	0	0
Fluxes.9	280	0	0	578.6317	-8.7650363	-1.244119	-21.654987	-19.66978	0	0	0
Fluxes.10	310	0	0	553.3955	-11.8627596	-1.070991	-6.233332	-15.02665	0	0	0
Fluxes.11	340	0	0	595.9827	-12.3087187	-1.335831	-8.305869	-15.50232	0	0	0
Fluxes.12	370	0	0	672.0552	-8.6328415	-1.916416	-28.913629	-20.97532	0	0	0
Fluxes.13	400	0	0	727.1051	-1.6967751	-2.575924	-76.841792	-30.01893	0	0	0
Fluxes.14	430	0	0	752.3490	7.1700969	-3.252736	-144.306384	-40.27515	0	0	0
Fluxes.15	460	0	0	759.7429	13.1237058	-3.699139	-204.273441	-49.06870	0	0	0
Fluxes.16	490	0	0	759.0090	13.3369849	-3.744917	-236.080054	-54.10716	0	0	0
Fluxes.17	520	0	0	750.7360	8.9187411	-3.486810	-229.756745	-54.07907	0	0	0
Fluxes.18	550	0	0	730.1311	2.5792581	-3.008231	-187.722236	-48.99381	0	0	0
Fluxes.19	580	0	0	692.9809	-2.5191724	-2.421899	-126.892274	-40.17779	0	0	0
Fluxes.20	610	0	0	641.8309	-5.8316866	-1.806503	-68.073233	-29.93123	0	0	0
Fluxes.21	640	0	0	586.7828	-8.3755990	-1.322385	-26.659457	-20.92698	0	0	0
Fluxes.22	670	0	0	553.5421	-11.3328090	-1.091066	-7.539950	-15.51415	0	0	0
Fluxes.23	700	0	0	584.5759	-12.6556633	-1.260382	-7.070734	-15.10619	0	0	0

... compared to mean fluxes

```
rbind(perturbflux = (colSums(PertFlux)/730)[2:7],
      ordinary = summary(PERTmix)[4,c("O2flux","NO3flux", "NH3flux", "H2Sflux", "DICflux", "PO4flux")])
```

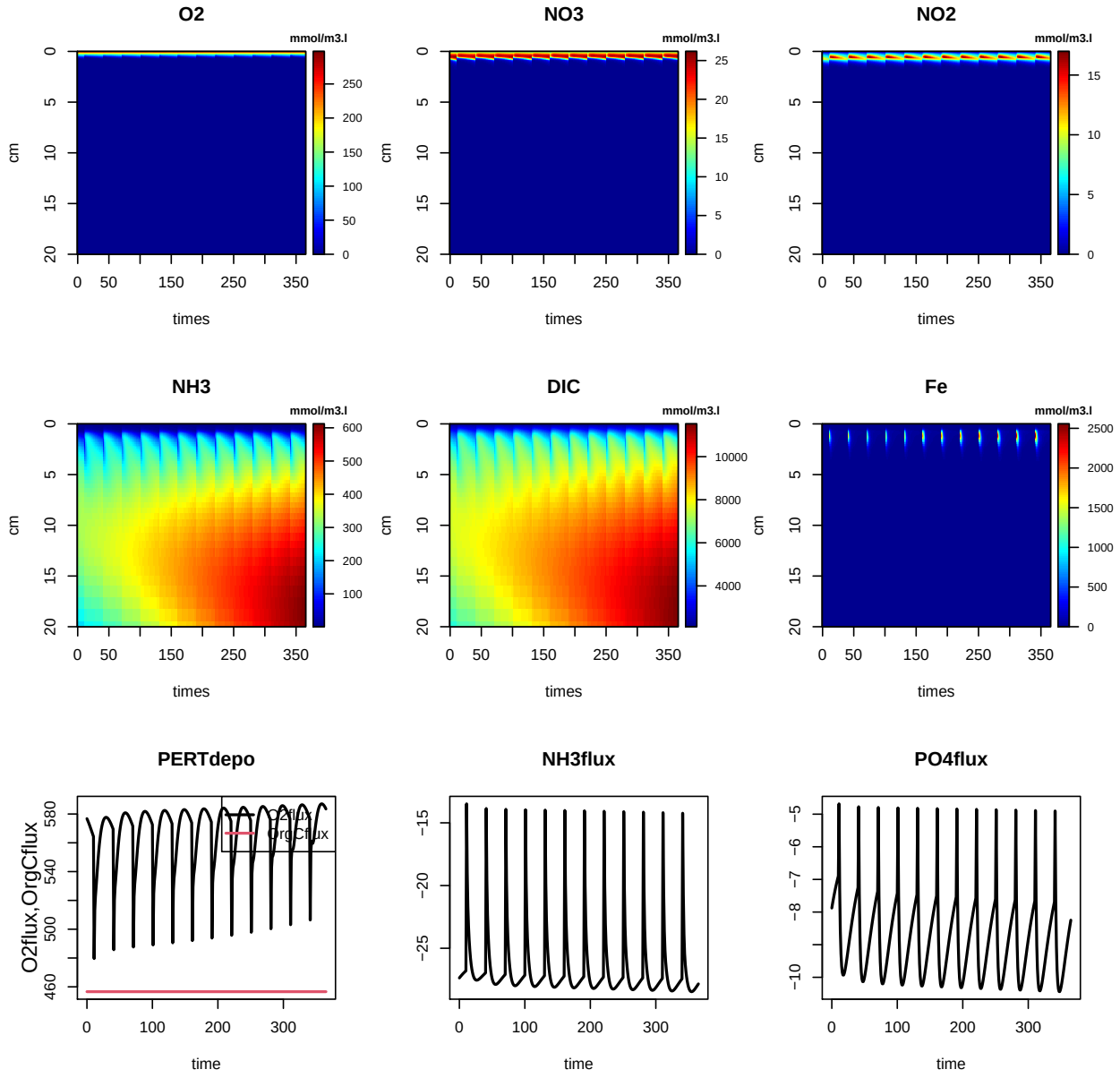
```
##                  O2flux       NO3flux   NH3flux      H2Sflux       DICflux   PO4flux
## perturbflux 9.344124e-16  3.189461e-13  22.48728  -0.02262237   -0.08085488 -3.656195
## ordinary    3.638565e+02 -1.474886e+01 -18.02227 -38.04723382 -354.42528682 -3.216043
```

deposition events - no loss in organics

We add a deposition event every month, and run this for 1 year, with one year spinup (spinup not shown). Here we keep the deposition flux constant.

In the first run, the deposited sediment has the same concentration of solids as that which is present (concfac = 1).

```
times <- 0:365
  PERTdepo <- FESDIAperturb(spinup = 0:365, times = times,
         perturbType = "deposit", perturbDepth = 1,
        perturbTimes = seq(from = 10, to = max(times), by = 30), verbose = FALSE)
image2D(PERTdepo, ylim = c(20, 0), which = 3:8, mfrow = c(3,3))
matplot.0D(PERTdepo, which = c("OrgCflux", "O2flux"), mfrow = NULL, lty = 1, lwd = 2)
plot(PERTdepo, which = c("NH3flux", "PO4flux"), mfrow = NULL, lwd = 2)
```



The instantaneous fluxes, compared to the other fluxes

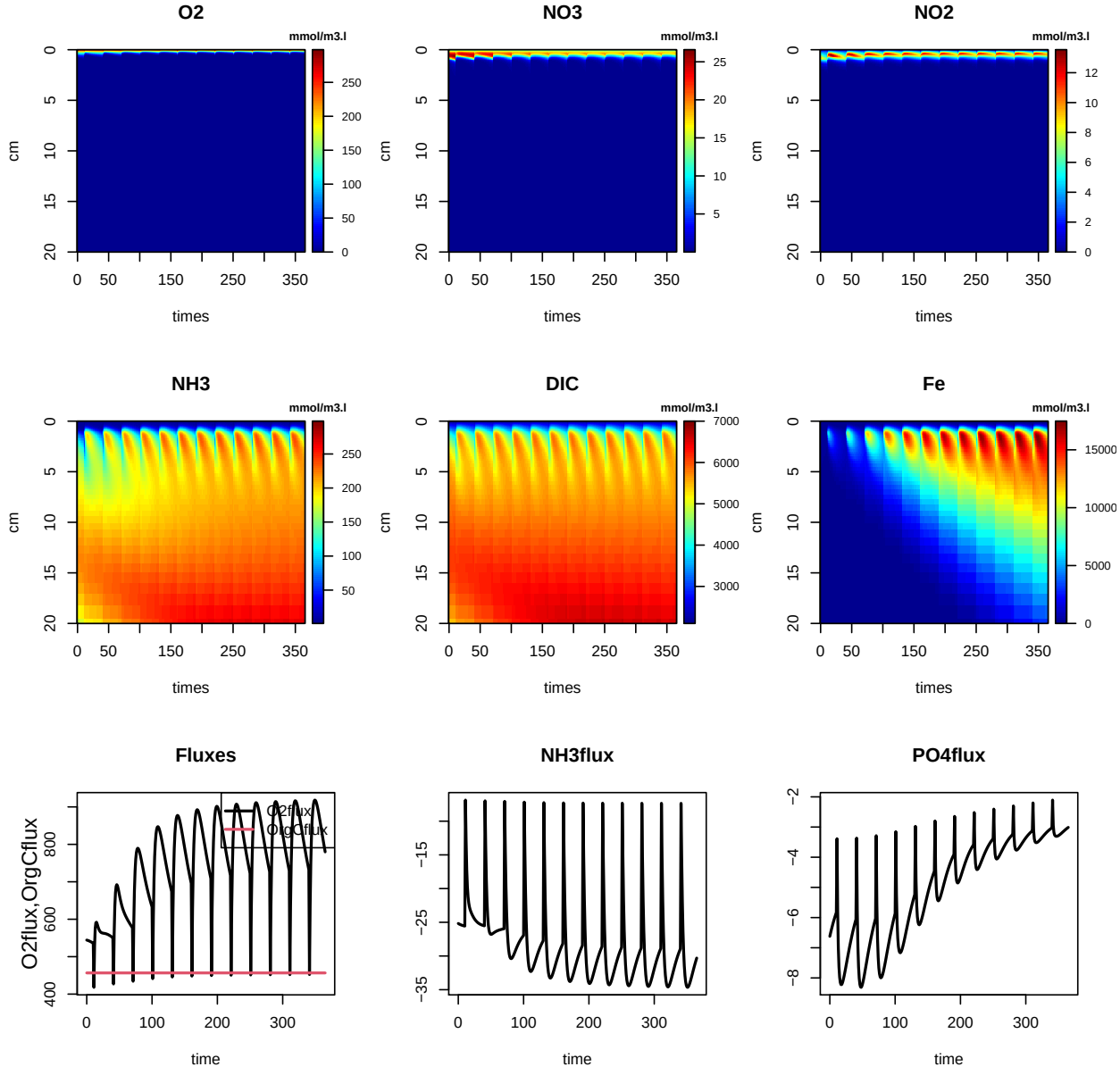
```
PertFlux <- attributes(PERTdepo)$perturbFlux
rbind(perturbflux = (colSums(PertFlux)/365)[2:7],
      ordinary = summary(PERTdepo)[4,c("O2flux","NO3flux", "NH3flux", "H2Sflux", "DICflux", "PO4flux")])
```

```
##              O2flux    NO3flux    NH3flux      H2Sflux       DICflux    PO4flux
## perturbflux 510.494 4105.94854   5.850209    0.1498788   -0.01411411 -0.4914708
## ordinary    568.757  -27.80058 -26.608684 -198.4962284 -956.87767455 -8.8391946
```

Deposition - loss in organics

The second run has half the concentration of solids as originally present (concfac = 0.5).

```
times <- 0:365
PERTdepo2 <- FESDIAperturb(spinup = 0:365, times = times,
        perturbType = "deposit", perturbDepth = 1, concfac = 0.5,
        perturbTimes = seq(from = 10, to = max(times), by = 30), verbose = FALSE)
image2D(PERTdepo2, ylim = c(20, 0), which = 3:8, mfrow = c(3,3))
matplot.0D(PERTdepo2, which = c("OrgCflux", "O2flux"), mfrow = NULL, lty = 1, lwd = 2, main = "Fluxes")
plot(PERTdepo2, which = c("NH3flux", "PO4flux"), mfrow = NULL, lwd = 2)
```



The instantaneous fluxes, compared to the other fluxes

```
PertFlux <- attributes(PERTdepo2)$perturbFlux
rbind(perturbflux = (colSums(PertFlux)/365)[2:7],
      ordinary = summary(PERTdepo2)[4,c("O2flux","NO3flux", "NH3flux", "H2Sflux", "DICflux", "PO4flux")])
```

```
##                O2flux    NO3flux    NH3flux    H2Sflux       DICflux    PO4flux
```

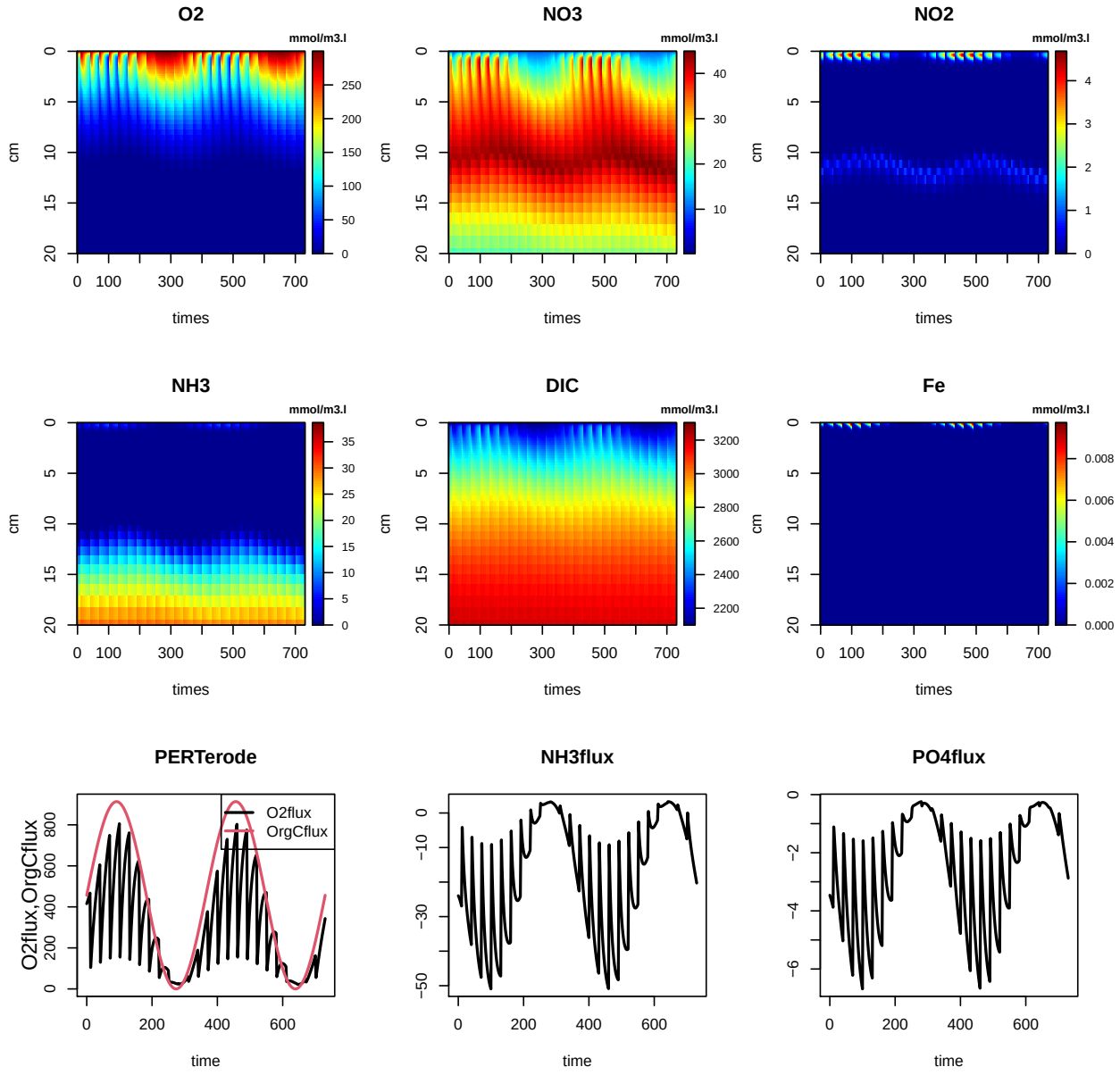
```
## perturbflux 392.0512 202.45340   5.916006  0.1565952   -0.01321592 -0.5905253
## ordinary    751.5441 -21.80914 -28.370272 -3.8049211 -766.40537319 -4.9577229
```

erosion

An erosion event every month removes a fraction of the sediment

```
times <- 0:365*2
PERTerode <- FESDIAperturb(Cflux = list(amp = 1), spinup = 0:365, times = times,
        perturbType = "erode", perturbDepth = 1,
        perturbTimes = seq(from = 10, to = max(times), by = 30), verbose = FALSE)

image2D(PERTerode, ylim = c(20, 0), which = 3:8, mfrow = c(3,3))
matplot.0D(PERTerode, which = c("OrgCflux", "O2flux"), mfrow = NULL, lty = 1, lwd = 2)
plot(PERTerode, which = c("NH3flux", "PO4flux"), mfrow = NULL, lwd = 2)
```



The instantaneous fluxes, compared to the other fluxes

```
PertFlux <- attributes(PERTerode)$perturbFlux
rbind(perturbflux = (colSums(PertFlux)/365)[2:7],
      ordinary = summary(PERTerode)[4,c("O2flux","NO3flux", "NH3flux", "H2Sflux", "DICflux", "PO4flux")]
```

```
##                 O2flux   NO3flux    NH3flux    H2Sflux       DICflux   PO4flux
## perturbflux -347.3248 -92.08120  -7.240208 -0.6722553   -0.06299142  1.265085
## ordinary     294.5556 -12.60642 -17.708106 -0.6749756 -264.77660181 -2.532542
```

```
cbind(FESDIAbudgetO2(DIA)$Fluxes, FESDIAbudgetO2(PERTerode)$Fluxes)
```

```
##                      O2            O2
## surface  4.955890e+02   2.943234e+02
## bottom  1.064018e-171  2.984607e-129
## perturb  0.000000e+00  -3.620104e+00
```

```
## netin    4.955890e+02   2.907033e+02
```

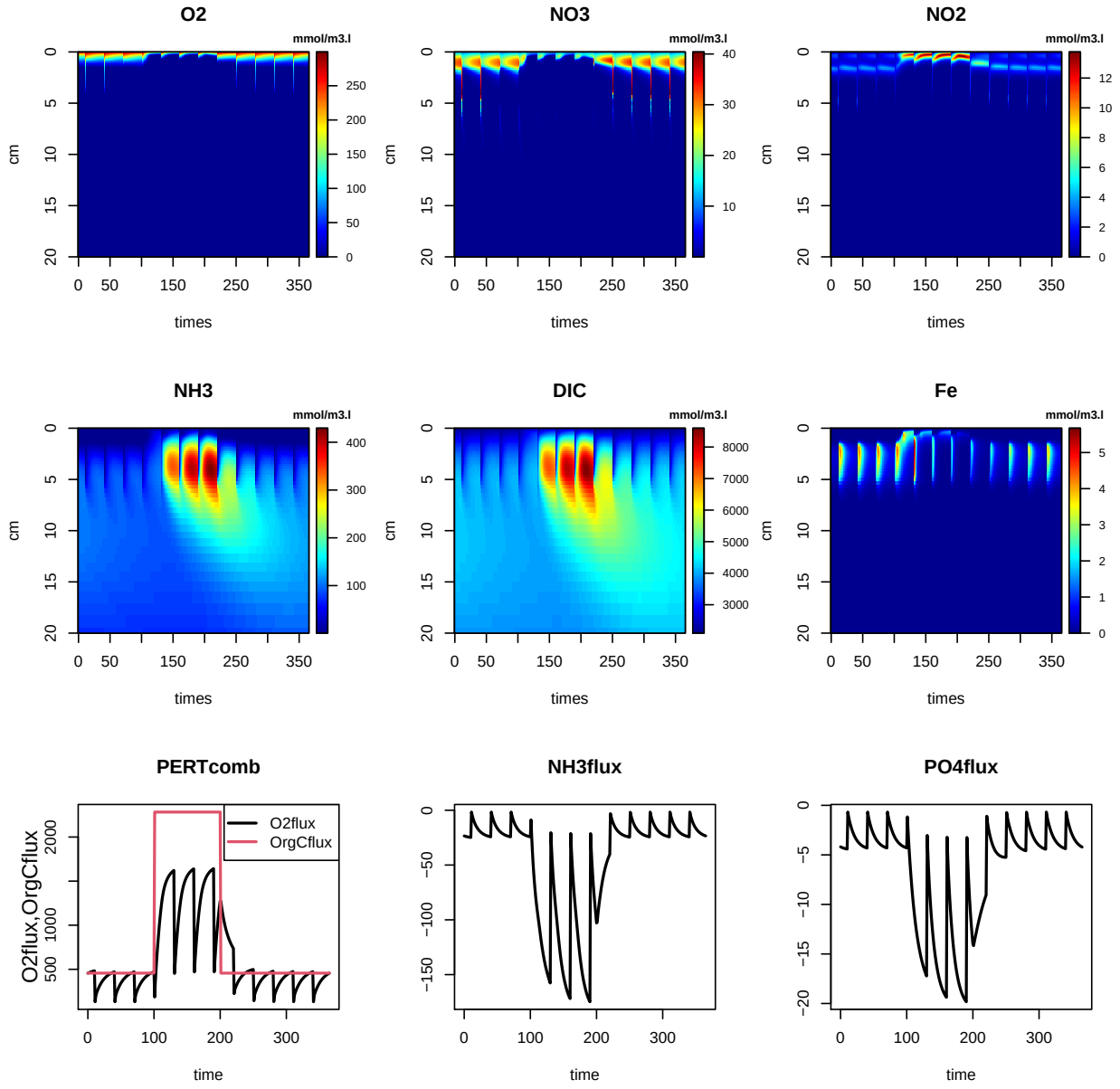
Mixing with variable forcings

We now combine a mixing event with a variable carbon deposition, imposed as a data series.

```
times <- 0:365
fluxforcdat <- data.frame(time = c(0, 100, 101, 200, 201, 365),
                          flux = c(20, 20, 100, 100, 20, 20)*1e5/12/365)
BASE <- FESDIAdyna(spinup = 0:365, times = times,
        Cflux = list(data = fluxforcdat))

PERTcomb <- FESDIAperturb(spinup = 0:365, times = times,
        Cflux = list(data = fluxforcdat),
        perturbTimes = seq(from = 10, to = max(times), by = 30))

image2D(PERTcomb, ylim = c(20, 0), which = 3:8, mfrow = c(3,3))
matplot.0D(PERTcomb, which = c("OrgCflux", "O2flux"), mfrow = NULL, lty = 1, lwd = 2)
plot(PERTcomb, which = c("NH3flux", "PO4flux"), mfrow = NULL, lwd = 2)
```



Mixing with explicitly modeled bottom water conditions

The simulation where bottomwater concentrations are also modeled is initiated with the steady-state conditions, while keeping the bottom water conditions constant.

```
std <- FESDIAsolve(dynamicbottomwater = FALSE, parms = list(Cflux = 20*1e5/12/365))
FESDIAbudgetO2(std, which = "Fluxes")
```

```
##                    O2
## surface  5.067625e+02
## bottom  3.701610e-131
## perturb  0.000000e+00
## netin    5.067625e+02
```

The initial conditions for the dynamic bottom water concentration run needs to have the bottom water concentrations in the first row.

The model is run for a couple of days, for mixing, erosion and deposition events.

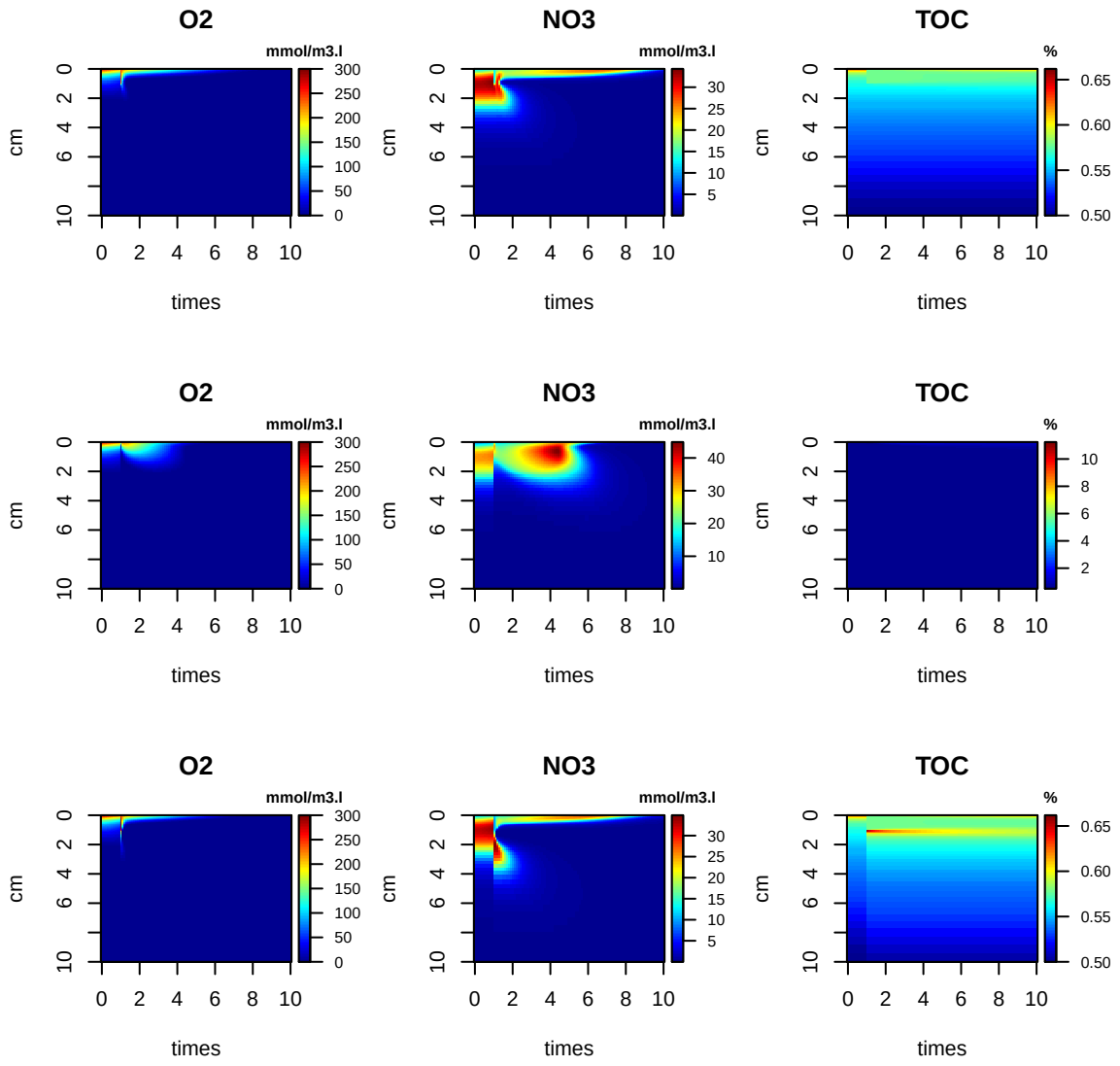
```
P <- FESDIAparms(std, as.vector = TRUE)[c("O2bw", "NO3bw", "NO2bw", "NH3bw", "DICbw", "Febw", "H2Sbw",
# order of state variables, FDET,SDET,O2,NO3,NO2,NH3,DIC,Fe,FeOH3,H2S,SO4,CH4,PO4,FeP,CaP,Pads,ALK
BW <- c(0, 0, P[c("O2bw","NO3bw","NO2bw","NH3bw","DICbw","Febw")], 0, P[c("H2Sbw","SO4bw","CH4bw","PO4bw
times = seq(0, 10, length.out = 100)
dyn  <- FESDIAdyna  (dynamicbottomwater = TRUE, yini = rbind(BW, std$y),
        parms = list(Cflux = 20*1e5/12/365), times = times)
dyn1 <- FESDIAperturb(dynamicbottomwater = TRUE, yini = rbind(BW, std$y),
                      perturbType = "mix", perturbTimes = 1, perturbDepth = 1,
        parms = list(Cflux = 20*1e5/12/365), times = times)
dyn2 <- FESDIAperturb(dynamicbottomwater = TRUE, yini = rbind(BW, std$y),
                      perturbType = "erode", perturbTimes = 1, perturbDepth = 1,
        parms = list(Cflux = 20*1e5/12/365), times = times)
dyn3 <- FESDIAperturb(dynamicbottomwater = TRUE, yini = rbind(BW, std$y),
                      perturbType = "deposit", perturbTimes = 1, perturbDepth = 1,
        parms = list(Cflux = 20*1e5/12/365), times = times)
```

```
image2D(dyn1, which = c("O2", "NO3", "TOC"), ylim = c(10,0), mfrow = c(3,3))
image2D(dyn2, which = c("O2", "NO3", "TOC"), ylim = c(10,0), mfrow = NULL)
```

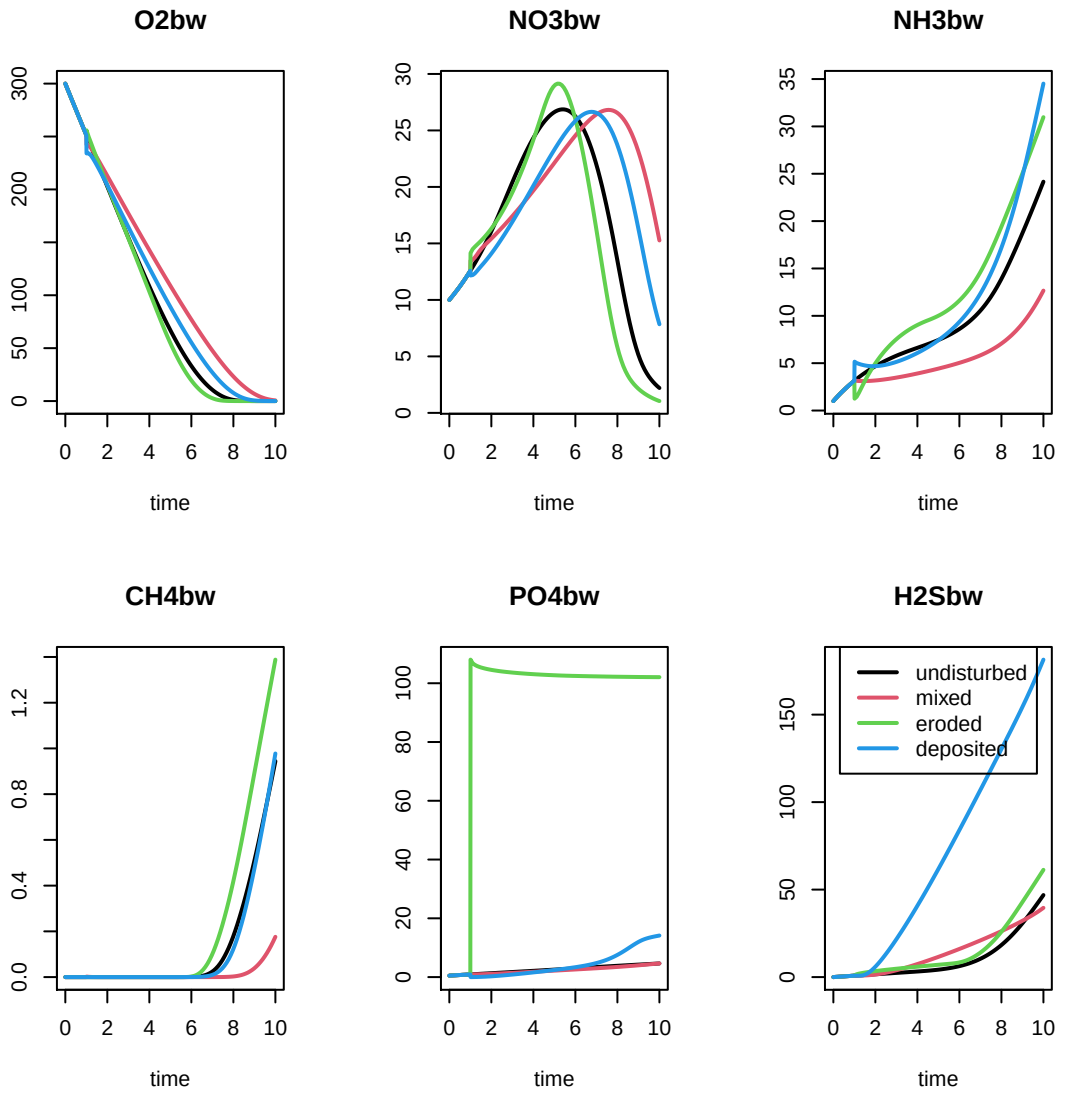
```
## Warning in rep(dots, length.out = n): 'x' is NULL so the result will be NULL
```

```
image2D(dyn3, which = c("O2", "NO3", "TOC"), ylim = c(10,0), mfrow = NULL)
```

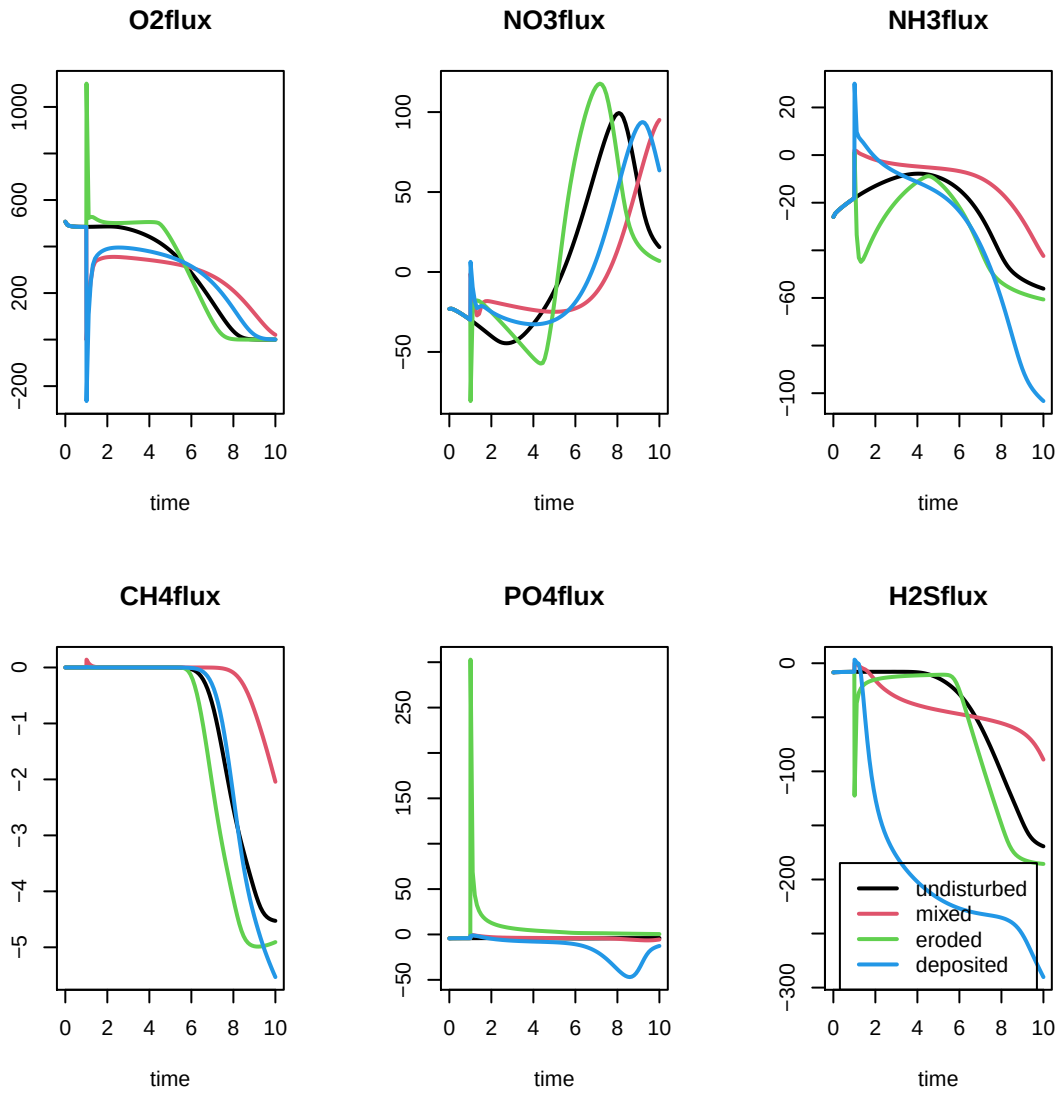
```
## Warning in rep(dots, length.out = n): 'x' is NULL so the result will be NULL
```



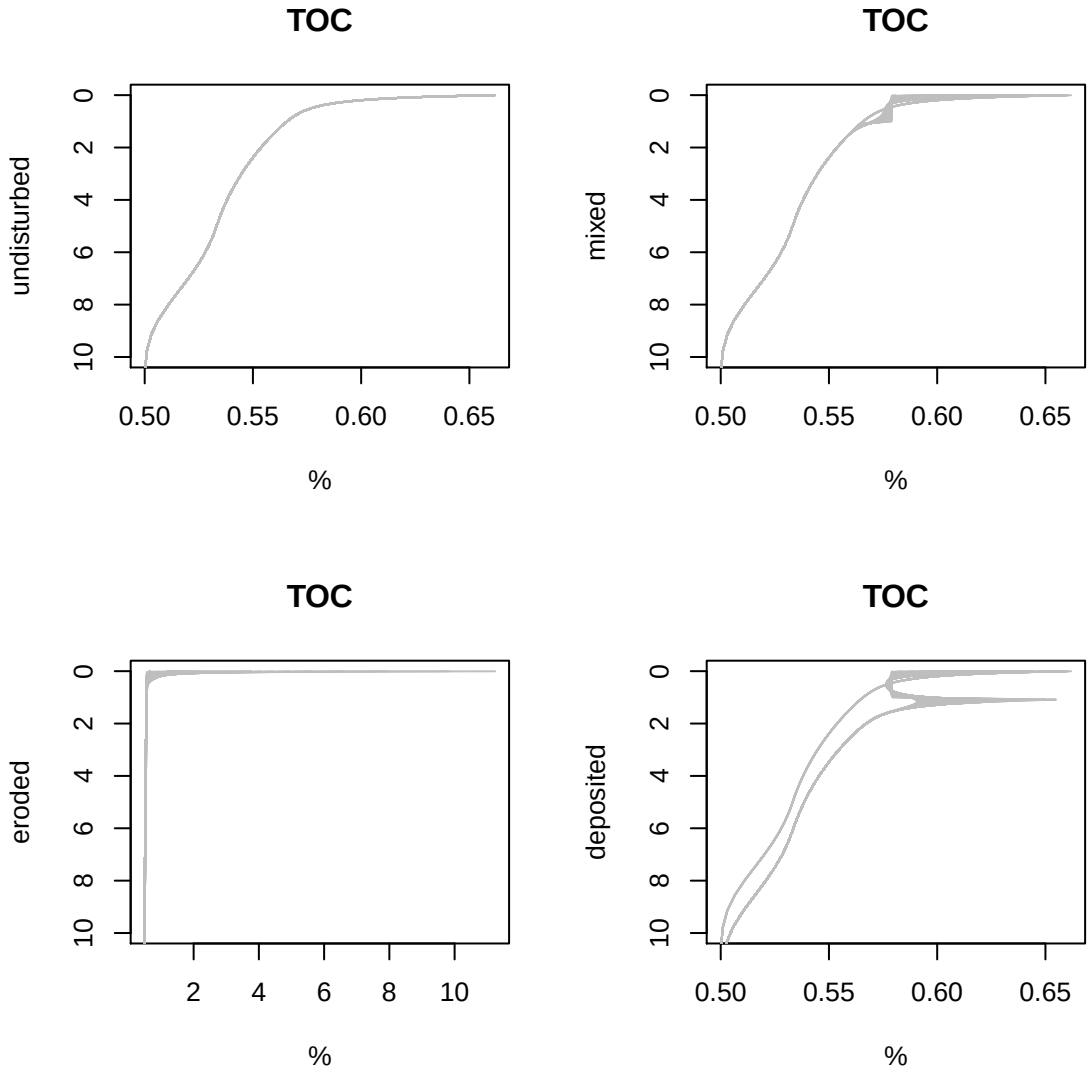

```
plot(dyn, dyn1, dyn2, dyn3, which = c("O2bw","NO3bw","NH3bw","CH4bw","PO4bw","H2Sbw"),
     type = "l", lwd = 2, lty = 1)
legend("top", col =1:4, lty = 1, lwd = 2, legend = c("undisturbed", "mixed", "eroded", "deposited"))
```




```
plot(dyn, dyn1, dyn2, dyn3, which = c("O2flux","NO3flux","NH3flux","CH4flux","PO4flux","H2Sflux"),
     type = "l", lwd = 2, lty = 1)
legend("bottom", col =1:4, lty = 1, lwd = 2, legend = c("undisturbed", "mixed", "eroded", "deposited"))
```




```
par(mfrow = c(2,2))
matplot1D(dyn, which = "TOC", type = "l",
          col = "grey", ylim = c(10,0), mfrow = NULL, ylab = "undisturbed")
matplot1D(dyn1, which = "TOC", type = "l",
          col = "grey", ylim = c(10,0), mfrow = NULL, ylab = "mixed")
matplot1D(dyn2, which = "TOC", type = "l",
          col = "grey", ylim = c(10,0), mfrow = NULL, ylab = "eroded")
matplot1D(dyn3, which = "TOC",  type = "l",
          col = "grey", ylim = c(10,0), mfrow = NULL, ylab = "deposited")
```



References

Soetaert K, PMJ Herman and JJ Middelburg, 1996a. A model of early diagenetic processes from the shelf to abyssal depths. Geochimica Cosmochimica Acta, 60(6):1019-1040.

Soetaert K, PMJ Herman and JJ Middelburg, 1996b. Dynamic response of deep-sea sediments to seasonal variation: a model. Limnol. Oceanogr. 41(8): 1651-1668.