Auxiliary Material Submission for "REMAP: A Reaction Transport model for isotope ratio calculations in porous media": User Guide

Boris M. Chernyavsky and Ulrich G. Wortmann
Geobiology Isotope Laboratory
Dept. of Geology, University of Toronto
22 Russell Street, Toronto, ON
Canada M5S 3B1
chernyavsky@geology.utoronto.ca

August 1, 2006

1 Installation

To use REMAP, you need a working Octave or Matlab installation. Octave can be obtained without charge from `http://www.octave.org`, and is available for all major operating systems. Our code has been tested on Octave 2.9.4, but should also run on earlier versions if you install the octave-forge package (`http://octave.sourceforge.net/`). Code performance relies heavily on the availability of optimized machine specific libraries. These are typically provided for Matlab, but have to be installed manually for octave. Make sure that you install an optimized BLAS library on your system (see e.g., `http://math-atlas.sourceforge.net/` or the goto-blas library `http://www.tacc.utexas.edu/resources/software/`, before installing octave. We also did some limited testing on Matlab, which suggest that the current version is fully compatible with Matlab version 7.2.
REMAP consists of a program file called remap.m and several other m-files containing subroutines. These files have to be either in the search path of your Octave/Matlab installation, or in the local directory where you intend to perform your modeling.

2 Usage

After starting an Octave/Matlab session, simply type

Octave:1>remap

and REMAP will ask for the input file name containing your model description. If you are on a Unix system using Octave, you can alternatively make the REMAP code directly executable by adding

```
#!/usr/local/bin/octave -qfi
```

as the first line of remap.m (adjust the path as needed), and issue the following command within a terminal window

chmod +x remap.m

These modifications allow you to run REMAP directly from a terminal window by issuing

./remap.m input.file

A similar procedure may be available for Matlab users.
The results of the simulation are written as comma separated values to an output file. Additionally, a separate file is created containing the auxiliary variables used in the computation, e.g., diffusion coefficients, the reaction terms, etc. The latter data is meant for debugging and analysis purposes, and usually written to remap_properties.csv.

3 Input file structure

REMAP knows 2 types of input files, one describing the model, and one holding data (e.g., porosity data etc.). The data file format consists of a required header line, followed by a single comma separated depth, value pair per line, e.g.,

```
# Depth, Porosity
1, 0.6
2, 0.5
3, 0.4
```

The data resolution is independent of the model resolution and REMAP will automatically interpolate or sub-sample the data file.

The format of the input file containing the model description allows for an arbitrary amount of comments and empty lines, but requires that all variable are given in the correct order, and that statements begin at the first column of each line. Comments must be preceded by the # sign at the beginning of the line. We provide several examples files (joergensen.rmp, site_1130_chloride.rmp, site_1130_sulfate.rmp) highlighting different aspects of the program capabilities. The above files contain short comments about each variable, and we provide below a detailed description of each parameter.

3.1 Input variables

REMAP requires the following input variables to be present. If the model requires less parameters, you can set the unneeded variables to zero, but they have to be present. The input files has several different data formats, the most basic one being just a number or a sequence of numbers. Unlike the data file format, no commas are used to separate numbers.
Several fields can contain either a constant, a function definition, or file-name as argument. For these fields, the user has to supply an identifier describing the type of input followed by the input. The identifiers are C=constant, E=expression, and F=file-name. The following list contains all input variables in the same order as they have to present in the input file, followed by an example of their actual value. The program does no sanity check on these variables and the onus is on the user to specify a physical meaningful model. I.e., you may get strange results if you use a Dirichlet condition on one species, and Neumann condition on another species, etc.

1. The first line of the input file must be the version number which is currently 0.1
   `0.1`

2. The next line contains the name of the file where all output data is written to. All file names should be alphanumeric without spaces or special characters.
   `my_model_out.csv`

3. Number of grid points to be used for the model. This number is independent from the resolution of the input data. The code will automatically re-sample input data to the chosen grid resolution. See the accompanying manuscript on how grid resolution may affect your results.
   `1000`

4. Offset of the first model grid point from seawater/sediment interface [m]. This option can be used in association with exponential functions used to describe model properties, to avoid a singularity near zero, or if your experimental data starts at some nonzero depth.
   `0`

5. Isotopic ratio of the reference material used in the δ calculations. To suppress isotopic calculations, set this to 0. E.g. for CDTV use
   `45004.5e-6`

6. Percentage of product phase which precipitates as a solid phase [0-1]. Note, that this calculation uses the approach of Joergensen 1979, GCA 43:363ff, which is not universally applicable. Setting this to 0 will disable the calculation of a precipitating phase and it's isotopes.
   `0`

7. The substrate boundary conditions ([mol/m3] = [mM]) are given as a pair of 4 numbers i.e., `[upper_value lower_type lower_value lower_gradient]`. The boundary type for the lower boundary can be either 1 to select a Dirichlet (concentration) boundary condition, 2 for a Neumann (gradient) boundary condition, or 3 for 'free' or 'floating' boundary. In the later case, you have to provide `lower_bc_value` and `lower_bc_gradient`. Note that a floating boundary can only be specified for the substrate, but not for the product, or isotopes, and that floating boundaries can not be used in time dependent calculations. Note that all four numbers must be present, even if some of them are unknown or unnecessary (e.g., you don't need both value and gradient for Dirichlet boundary condition). In this case, use zero for unknown/unneeded variables. See the free boundary example on how to set up a floating boundary.
   `28 1 5 0`

8. Boundary conditions for isotopic calculations of the substrate in terms of isotopic ratio [permil].
   `[upper_value lower_type lower_value lower_gradient]`. The boundary type for the lower boundary can be either 1 to select a Dirichlet (concentration) boundary condition, or 2 for a Neumann (gradient) boundary condition.
   `21 1 40 0`

9. Product boundary conditions [mol/m3] = [mM]
   `[upper_value lower_type lower_value lower_gradient]`. The

boundary type for the lower boundary can be either 1 to select a Dirichlet (concentration) boundary condition, or 2 for a Neumann (gradient) boundary condition.
`0 1 10 0`

10. Boundary conditions for isotopic calculations of the product in terms of isotopic ratio [permil]
`[upper_value lower_type lower_value lower_gradient]`. The boundary type for the lower boundary can be either 1 to select a Dirichlet (concentration) boundary condition, or 2 for a Neumann (gradient) boundary condition.
`0 1 0 0`

11. Boundary conditions for the precipitating phase [mol/m3] = [mM] `[upper_value lower_type lower_value lower_gradient]`. The boundary type for the lower boundary can be either 1 to select a Dirichlet (concentration) boundary condition, or 2 for a Neumann (gradient) boundary condition.
`0 1 0 0`

12. Boundary conditions for the isotopic ratio calculation in the precipitating phase in terms of isotopic ratio [permil] `[upper_value lower_type lower_value lower_gradient]`. The boundary type for the lower boundary can be either 1 to select a Dirichlet (concentration) boundary condition, or 2 for a Neumann (gradient) boundary condition.
`0 1 0 0`

13. Duration of the simulation run [s]. If set to `Inf` the program will perform a steady state simulation. Alternatively, it can be set to a finite time, e.g., to see how a systems changes within a given time frame. In this case you will need to provide in the same line the name of the file containing the initial condition for the system (e.g., the output from a steady state simulation). The file should be in the same format as the remap output file (see above). You can also provide optional expressions describing the change of the a variables over time. The allowed time dependent variables are `alpha` and `omega`, and you can modify the boundary conditions in a similar way. The latter case requires however that the user writes an equation for the corresponding internal variables, which are stored in the array `c.bc(x,y)` where the subscripts `x` and `y` denote the species and boundary type respectively. The possible values of `x` are 1=substrate, 2=substrate isotopes, 3=product,

4=product isotopes, 5=precipitate, and 6=precipitate isotopes. The possible values of `y` are 1 to 4 corresponding to the 4 fields defining a boundary condition. Note that unlike the spatially variable parameters (see below), you must provide a full equation, rather than only the right hand side of it, and that the time dependent solver requires either a Dirichlet or Neumann condition. To set a time dependent change of the fractionation factor, one could write

```
1e12 initial_conditions.csv alpha=alpha*(1+time/1e10)
```

14. Porosity data [0-1]. Data can be given as a numerical constant (preceded by identifier *C*), user supplied optional data file (preceded by identifier *F*), or as an explicit equation (preceded by identifier *E*). If you use the data file option, the data file must have a header as first line, followed by a list of comma separated values, of depth, porosity pairs. The ODP_1130 case provides an example how to read porosity data. Note that the current code only handles steady state compaction, and that your porosity file must satisfy this condition. Because of this, we assume that the last entry in the porosity file is at the same time the porosity for which no further compaction occurs. A detailed discussion on this can be found in Berner 1980, "Early Diagenesis: A Theoretical Approach", Princeton University Press.

    You may also specify an explicit function, for details see the explanation on substrate consumption below. Note that the code will not perform sanity checks for the values derived from a user provided function. Check the output of the properties.csv file to see how your function was interpreted.
    ```
    C 0.6
    ```

15. Temperature data [C], provided in a numerical constant, user data file or explicit equation form and preceded by an identifier *C, F,* or *E*. Data file format requirements are discussed under porosity.
    ```
    F my_porosity_file.csv
    ```

16. Biologically mediated substrate consumption [mol/m3/s]. Set this to zero the compute a conservative species. In this case no product phase will be computed and the code terminates after substrate calculation is completed. Data can be provided in the form of a constant, data file, or an explicit equation. If you use an explicit expression, you will need to provide the right hand side of the equation only. The user

may build functions using Octave/Matlab syntax from any program internal variable. However, as most functions are depth dependent, we provide a variable named Z (i.e., depth) which facilitates simple user defined function definitions. To set up e.g., a depth dependent function for bacterial activity, one could write:

```
E 1.025e-9 * (Z/1.5113 + 0.1)^-2.1
```

If you use a Monod type limiter, the limiter will be applied on top of whatever function you specify in this location.

17. Fractionation factor [permil], which can be either a numerical constant, a data file, or an explicit equation form preceded by the right identifier (*C, F,* or *E*). Note that if you use a Monod modifier (by setting Monod limiter for fractionation factor non equal to zero), it will be applied on top of whatever function you specify in this location.
`1.025`

18. Sedimentation rate [m/s]. Note that the sedimentation rate must be always positive. The optional second value corresponds to the porosity value at the depth where compaction is assumed to be completed. This value is used to calculate the upward velocity of the fluid phase as result of compaction of the solid phase. If no value is specified, porosity value at maximum depth will be used (note that this option imply that porosity values were supplied from a file). Note that the current code only handles steady state compaction, and that your porosity file must satisfy this condition (see Berner 1980, "Early Diagenesis: A Theoretical Approach", Princeton University Press, for discussion).
`9.8935e-12`

19. Externally imposed flow field if present [m/s]. This value is negative for an upward directed flow, and positive for a downward directed flow.
`0`

20. Substrate diffusion parameter(s). If you provide a single number it will be interpreted as a fixed (depth independent) diffusion coefficient [m^2/s]. If you provide two numbers, and a temperature and porosity files (see above) the program assumes that these are the regression parameters [mo,m1] and calculates the diffusion coefficient as

$$D = \frac{(m_0 + m_1 t)10^{-10} m^2 s^{-1}}{1 - \ln(\varphi^2)}, \tag{1}$$

where φ denotes porosity and t temperature. See B.P. Boudreau, 1996, "Diagenetic Models and their Implementation", Springer, New York, for further discussion.
`4.88 0.232`

21. Product diffusion parameter(s), see above.
`10.4 0.273`

22. Monod type limiter function [mM] reflecting the substrate dependence of biological processes (e.g., metabolic activity or fractionation factor), see accompanying manuscript for discussion. Note that a non-zero limiter will be applied on top of whatever function or data you specify for the reduction term or for α.
`0 0`

The following parameters are recommended to be used only by advanced users, but need to be present.

23. γ and β coefficients, controlling the numerical scheme. The default values for two-layer (semi)implicit function is $\beta = 1$ and $\gamma = 0$. You can modify them to change the scheme into explicit $\beta = 0$ and $\gamma = 0$, three layer (semi)implicit $\beta = 1$ and $\gamma = \frac{1}{2}$ or Crank-Nicholson $\beta = \frac{1}{2}$ and $\gamma = 0$.
`0 1`

24. Accuracy limits [ϵ_{grad}, ϵ_c, ϵ_{cd}, ϵ_z]. ϵ_{grad} defines the accuracy required to satisfy the boundary condition (for Neumann and free lower boundaries) (note that this value must have the correct sign) [mM/m], ϵ_c is the convergence accuracy for concentration calculations [mM], ϵ_{cd} is the convergence accuracy for the isotope ratios, based on δ, and ϵ_z [m] is the accuracy used to locate the extinction depth for free boundary problems.
`1e-6 1e-6 1e-6 1e-6`

25. Maximum number of iterations allowed to achieve convergence during iterative solution and to find the location of the extinction depth.
If you model does not converge, increase this number by an order of magnitude, or modify the time step.
`1e4 1e4`

26. Initial time step [s]. If your model does not converge, try changing the time step. 3e10 s is approximately 1 ky.
`3e10`

27. Initial step [m] for a free boundary location iteration (should be set to zero for fixed boundary).
    `0`

28. Fiadeiro hybrid scheme utilization (0 (default) - if not used, 1 if used)
    `0`

29. Name of the file where additional information about the internal state of the program will be stored.
    `model_properties.csv`

4 License